Rhine-Waal University of Applied Sciences

Applied Research Project
Mechanical Engineering, M.Sc.

Controller Design for a Mobile Robot

Giridhar Vitta Bukka
20456

supervised by

Prof. Dr. Ronny Hartanto

Contents

List of Figures

[1] 1

List of Tables

Abstract

In an attempt to contribute to the Field Robot Project at Rhine-Waal University of Applied Sciences and also as an Applied Research Project, submitted in partial fulfillment of the requirement for the Degree of Masters of Science, Mechanical engineering at Rhine-Waal University of Applied Sciences, a Prototype of the Mobile Robot with Skid-Steer type mechanism is assembled and an algorithm for a Low-Level controller is designed to achieve pose regulation and trajectory tracking problem. Mathematical model of the mobile robot is created and a control strategy as discussed in [13], is used to develop the algorithm for implementation on the prototype of the mobile robot. This report discusses the approach taken, challenges faced and results obtained in realizing the goal of designing a controller for a mobile robot prototype of skid-steering type mechanism while understanding the controller design process for non-linear Dynamic systems.

Chapter 1

Introduction

The prominence of autonomous mobile robots is rapidly increasing. Their potential to handle routine and boring jobs for human beings is already being realized. To push their potential further various events like Field Robot Event are being promoted. At Field Robot Event, two of the main tasks to be achieved were related to navigation, where the mobile robot had to autonomously navigate itself along the curved rows of maize plants in a realistic field scenario with obstacles $<$ 25mm size above ground level. To achieve these tasks a robust low-level controller capable of tracking the trajectory and pose regulation are necessary. With an emphasis on these tasks controller design process in this project is explored.

This type of mobile robots is used widely because of its simple drive system and is considered as all-terrain vehicle because of its robustness. However, due to the complex wheel-ground interactions and the kinematic constraints, it is a challenge to understand the kinematics and dynamics of such a robotic platform [19]. Mobile robots of such drive mechanism during its normal operation are prone to slip and skid. This brings complexity in estimating the change in position over time. Many research papers provide various approaches to tackle the odometry problem for a mobile robot of skid-steering type mechanism. As the time frame for the research in this project was limited in-depth consideration of the Odometry is neglected and assumed to be known.

Path planning and motion control of mobile robots are important aspects of controller design. Usually considered as tasks of a high-level controller and determined by the mobile robot perception. This research project addresses regulation problem and can also be easily extended to the trajectory tracking problem for the prototype built. To facilitate the controller design, the physical system is mathematically described using first principles. Control algorithm is developed to achieve the tasks set point and trajectory tracking as a unified control problem, considering the kinematics, dynamics and actuator limitation of the mobile robot prototype. Simulation tools like Matlab and Simulink are used to simulate the algorithm, plot some interesting graphs among which are convergence plots and path of mobile robot.

Various Mathematical and Control theory tools and methods like Coordinate Transformation, Non-holonomic Constraint optimization with Lagrange multiplier, Parameter estimation using Curve fitting, Regression analysis, control law design using Lyapunov stability criteria with Backstepping control framework, Numerical methods for embedded system implementation, Non-stationary differential equations and their solutions, non-linear dynamic system analysis and Error dynamics applicable for open-loop error system and closed-loop error system analysis have been introduced and explained at a length proportional to their relevance in a systematic flow.

Chapter 2

Requirements study

2.1 Mechanical System Requirements

- The Prototype frame weight to not exceed 15kg
- Rigid structural design to carry mechanical, electrical and electromechanical components on the frame
- The distance between ground to base of the mobile robot needed to be more than 30mm to avoid the contact with obstacles of size 25mm

2.2 Controller Requirements

Setpoint control and Trajectory tracking control with pose regulation are the controller's main requirements.

- Setpoint is a fixed constant point in the inertial frame where the mobile robot will be traversing. The controller should be able to achieve setpoint goal with pose regulation. Choice of the setpoint cannot be arbitrary and can only be chosen based on the controller design.
- Trajectory is a geometric path with an associated timing law or in other words time parameterized reference. The controller should be able to force the mobile robot to follow the Trajectory. Choice of trajectories cannot be arbitrary and can only be chosen based on the controller design.
- Pose regulation is the ability of the controller to regulate the orientation of the mobile robot as it reaches the goal position to the desired orientation.

2.3 Mobile Robot System Requirements

- Max speed of the mobile robot 2.5m/s

- To be capable of achieving the setpoint and trajectory tracking task with pose regulation on the concrete floor
- Capable of having a payload of 5kg to carry perception sensors and onboard embedded system

Chapter 3

Mechanical design and assembly

3.1 Selection of the system components

As the author has taken up the project already started by previous Field robot team members, the conscious choice of few components was not possible. However, a detailed explanation is given below for the validity and usability of already procured motors and gearboxes to ensure the suitability of the procured parts for the current requirements.

3.1.1 Drive system components

BLDC motors Electric motors especially BLDC motors do not produce high torques needed to run a Field mobile robot which is usually heavier and their weight range falls above 10kg in general. The prototype built in this project weighs 12Kg including all mechanical and electro-mechanical components. However, BLDC motors have high speeds of rotation proportional to the voltage supplied. The torques achievable by such motor is enough to drive the gearbox input shaft which can be a tradeoff for high speeds of the BLDC motors to reasonable torque values. The table 3.1 gives an idea of the capability of the motor speeds and torques.
Quantity : 2 $\times$Orbit 25-16 from Plettenberg Elektromotoren GmbH

Table 3.1: BLDC motor Orbit 25-16 from Plettenberg Elektromotoren capability.

Rpm 1/min	**Torque** Nm	**Voltage** V	**Current** A
10411	0.68	21.4	40

Gearbox As mentioned in [7, p. 200] gearbox, in general, brings some disadvantages like increased cost, added weight, added friction and mechanical noise. The increased cost and the added weight is worth considering as the gearbox efficiently multiplies the motor

torque 12 times which is needed for the mobile robot optimal motion. This particular gearbox due to its structural design is very smooth and silent. The table 3.2 shows capability of the gearbox.
2 X PLG60 Dunkermotoren

Table 3.2: Gearbox PLG60 from Dunkermotoren capability.

Efficiency	**Reduction ratio**	**ContinuousTorque** Nm	**EmergencystopTorque** Nm
0.81	12:1	21.4	28

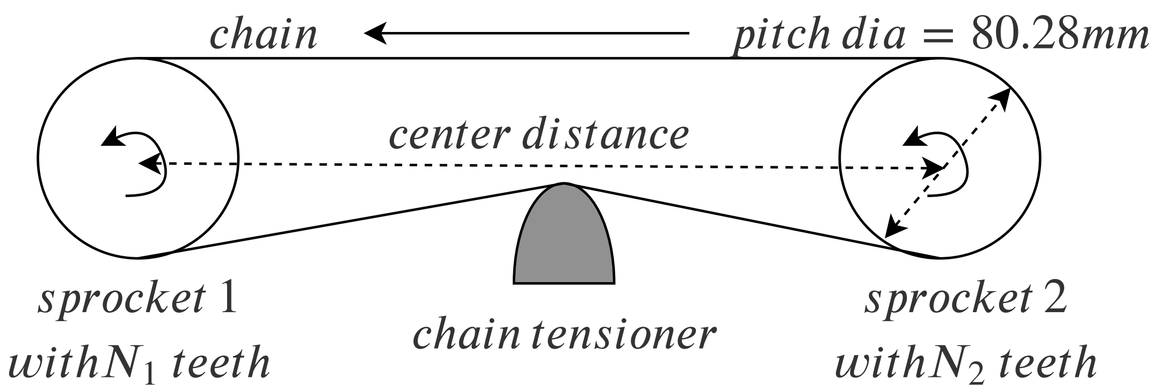


Figure 3.1: Free body diagram of sprocket chain drive system.

Sprocket and Chain In Skid-steer type mechanism mobile robots, wheels on the each side of the vehicle ideally should rotate at the same speed. To achieve the same speed of rotation between wheels on each side of the vehicle a mechanical connecting link with a combination of chain and sprocket is introduced. Care has been taken to avoid slag of the chain between two sprockets by using chain tensioner as shown in fig 3.1, hence contributing to minimal lag of rotation between the two wheels.
4 × Low carbon steel sprocket

$$C = \frac{centerdistance}{chainpitch},$$

$$L = \frac{N_1 + N_2}{2} + 2C + \frac{\frac{N_1 - N_2}{2\pi}}{C}, \qquad [16]$$

$$L : Numberofchainlinks \qquad ,$$

$$N_1 : Numberofteethonsprocket1 = 42,$$

$$N_2 : Numberofteethonsprocket2 = 42,$$

$$centerdistance = 295mm,$$

$$chainpitch = 6mm,$$

$$C = \frac{295mm}{6mm},$$
$$L = \frac{42+42}{2} + (2 \times \frac{295mm}{6mm}) + \frac{\frac{42-42}{2\pi}}{\frac{295mm}{6mm}} = 140.33 \cong 140 Links,$$

$Chainlengthforonesideofvehicle = 140 \times 6 = 858mm = 0.858m.$
Chain on both sides of the vehicle is needed hence total length of $0.858m \times 2 = 1.716m + allowanceforchaintensioner = 1.8m$ is bought from MÄDLER GmbH.

Wheels For the prototype, wheels from a toy bike were chosen by the previous team. According to the mobile robotics jargon the wheels are of *fixed wheel* type as shown here

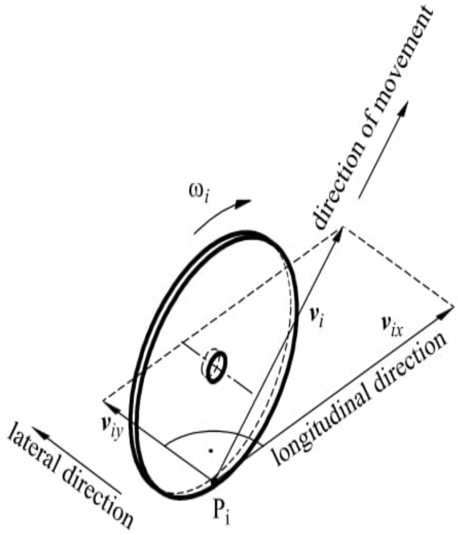


Figure 3.2: Fixed wheel [13]

The material of the wheel in contact with the surface on which mobile robot will be traversing is rubber. The surface on which the mobile robot will be traversing is considered as dry concrete. Coefficient of static friction and kinetic friction for such combination of the material surfaces i.e. between rubber and concrete are taken as given in [15, p. 257] as 1 and 0.7 respectively for reference.

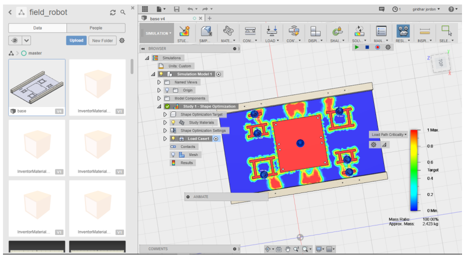
ESC (Electronic Speed Controller) A prebuilt circuit used to control the speed of the BLDC motor. ESC's are available for both BLDC and DC motors. BLDC motors can be controlled with two types of ESC's sensorless and with sensor ESCs. In this project, sensorless ESC's are procured. For specifications of the ESC please refer table 3.3

Table 3.3: Esc specifications.

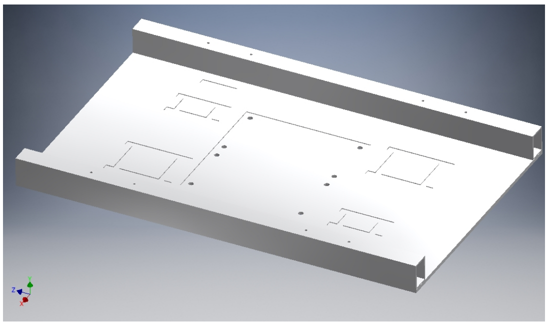
Continuous max current A	**cell count**	**BEC voltage current** V A	**Type of ESC**
60	up to 6s LIPO	5.5 3	sensorless

3.1.2 Design of Chasis and Assembly

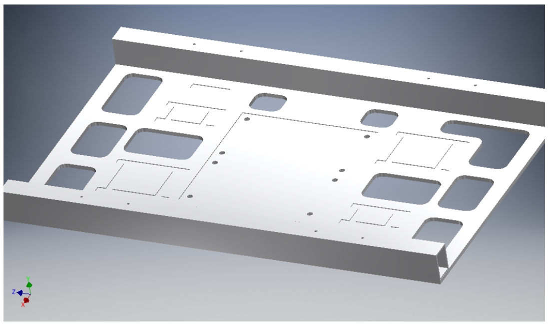
Chassis for the mobile robot Rigid and structurally optimized chassis is required to accommodate all the components on the mobile robot in place. Material such as Aluminum which is Light, affordable yet rigid enough to suffice the necessity for the current project is chosen to fabricate the chassis of the mobile robot.



Designed chassis as in fig 3.4a is analyzed for the structural integrity by conducting static stress analysis, see 3.3. The shape optimization technique has been used to identify over-designed portions of the chassis frame and eventually leading to weight reduction of the frame as shown in fig 3.4b.



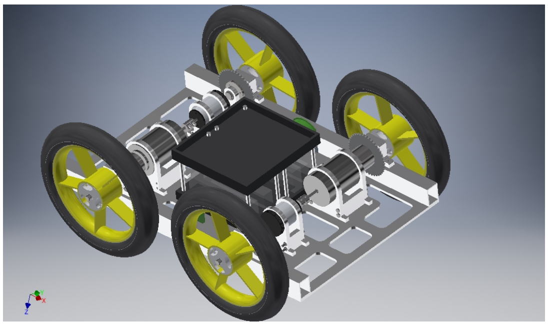
(a) before Shape Optimization



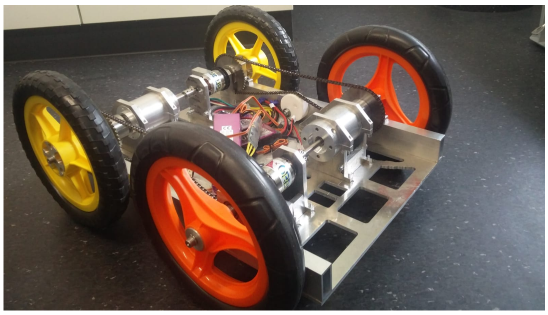
(b) Optimized shape

Figure 3.4: Mobile robot chassis

Assembly All the individual components are fabricated in the university workshop. Fabricated individual components are assembled and the prototype functionally ready to be programmed is made available, see 3.5b.



(a) CAD model



(b) Prototype built

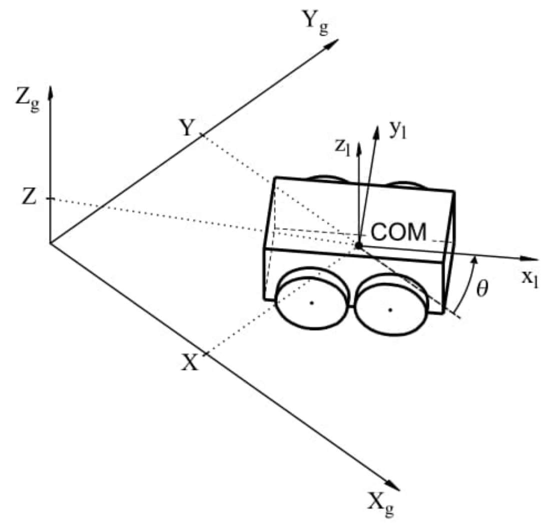
Figure 3.5: Mobile robot assembly

Chapter 4

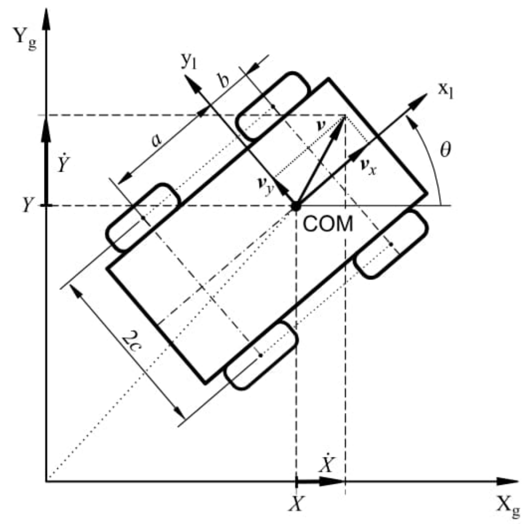
Theory and Methods Relevant for Design of a Low-Level Controller for Mobile Robot

4.1 Reference Frames

Inertial frame or Fixed world reference frame or Global coordinate frame Mobile robot's position and orientation in 2D space are referred to as its pose. Mobile robot's pose is described in relation to a fixed frame of reference also called Inertial frame denoted by (x_g, y_g, θ) as shown in 4.1a. A left-handed coordinate system convention is used throughout the report.



(a) Global coordinate frame (x_g, y_g, θ)



(b) Local coordinate frame (x_l, y_l, θ)

Figure 4.1: Reference Frames [13]

Robot frame or Local coordinate frame This frame is attached to a point on the mobile robot. This frame moves along with the mobile robot and is denoted by (x_l, y_l, θ) as shown in 4.1b. In this report using, global coordinate frame (x_g, y_g, θ) and local coordinate frame (x_l, y_l, θ), construction of the kinematic model for skid-steer mobile robot describing the motion of the robot in the inertial frame as function of its own geometry and wheels behavior is made with the help of transformations between the local and global coordinate frames.

Conventions often used in robotics for simplification and better control for moving objects in space are Fernet-Frame convention Denavit–Hartenberg convention

4.2 Coordinate Transformation or State Transformations and its Importance

Dynamic and kinematic models described using the first principles are transformed into state-space forms. The structure of the state-space models can often be recast into special descriptions referred to as *canonical forms* [18, p. 59]. Canonical forms are of special interest as they bring simplicity to the design procedure. There are a variety of canonical forms studied and applied in the field such as *chained form*, power form etc.

In other words original model equations are transformed into canonical forms with coordinate transformation. In this research report kinematic error model of the skid-steer mobile robot is allowed through coordinate transformation resulting in nice geometric interpretation: z_1 as orientation error and z_2 as length of projection of the vector $[\tilde{X}\,\tilde{Y}]^T$ as mentioned in [13], where $\tilde{X}$ and $\tilde{Y}$ are error components.

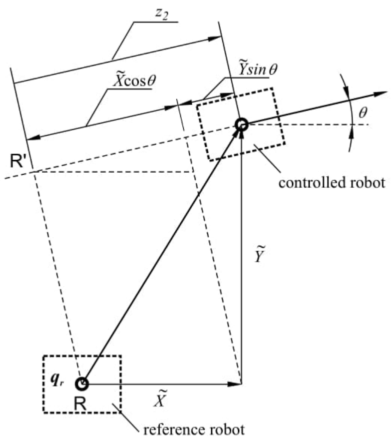


Figure 4.2: coordinate transformation-canonical form [13]

$$z_1 = \tilde{\theta},$$

$$z_2 = \tilde{X}cos\theta + \tilde{Y}sin\theta,$$

with canonocal form,

$$\dot{z}_1 = u_1,$$

$$\dot{z}_2 = u_2,$$

$$\dot{w} = z_2u_1 - z_1u_2,$$

Choice of the description for two new coordinates z_1 and z_2 was made based on [8, p. 9] and the representation in *chained form* was made based on [4, p. 182] where $\dot{w}$ was defined as $z_2u_1 - z_1u_2$.

A simple coordinate transformation also used often in mobile robotics path planning for simplicity is Cartesian to polar coordinate transformation.

4.3 Linear vs Non-linear system

Most processes in the real world can be considered as dynamic systems which can be described mathematically. These real-life processes from mathematical descriptive perspective can be seen as linear and non-linear processes based on their input-output mapping. A few important properties that highlight the complex nature of non-linear system response as opposed to linear system response are mentioned here 4.1 which are referred from [2, p. 5].

Table 4.1: Linear system Vs Non-linear system

Linear system	**Non-linear system**
Mathematical representation	
$\dot{x} = Ax + Bu$	$\dot{x} = f(x, u)$
Equilibrium	
• Unforced system (u=0) equilibrium point is unique if A is non-singular	• Unforced system (u=0) has 1 or multiple equilibrium points
Stability	
• Stability about equilibrium point is independent on initial conditions, forcing functions,the concepts of local or global behavior • system is stable if all eigenvalues of A have negative real parts	• Stability about equilibrium point is dependent on initial conditions, forcing functions, the concepts of u local or global behavior • Exhibit limit cycles which are closed, unique trajectories or orbits • There equilibrium manifolds may be attractive or repulsive
Forced Response	
• Satisfy the property of superposition $x(u_1(t)+u_2(t)) = x(u_1(t))+x(u_2(t))$ • Satisfy the property of homogeneity $x(\alpha u(t)) = \alpha x(u(t)))$	• Do not satisfy the property of superposition • Do not satisfy the property of homogeneity

4.4 Mathematical description of the Dynamic System

4.4.1 Modeling

"The temporal behavior of systems, such as e.g. technical systems from the areas of electrical engineering, mechanical engineering, and process engineering, as well as non-technical systems from areas as diverse as biology, medicine, chemistry, physics, economics, to name a few, can uniformly be described by mathematical models"[11]. The process of creating these mathematical models is called Modeling. Theoretical modeling and Experimental modeling are two types of modeling approaches.

Theoretical modeling

Plant modeling Mathematical description of the kinematics and dynamics of various components needed to design a controller is made. The main components considered here are, the drive system consisting of BLDC motors, gearbox coupled with power transmitting components and mobile robot frame with its wheels. This description is also called as plant modeling and is done based on first principles and available prior information. The fig 5.2 shows the possible approaches to obtain mathematical descriptions.

4.4.2 Mathematical description of the mobile robot

Dynamic model for skid-steer type mobile robot for this research project is based on [13] [9].

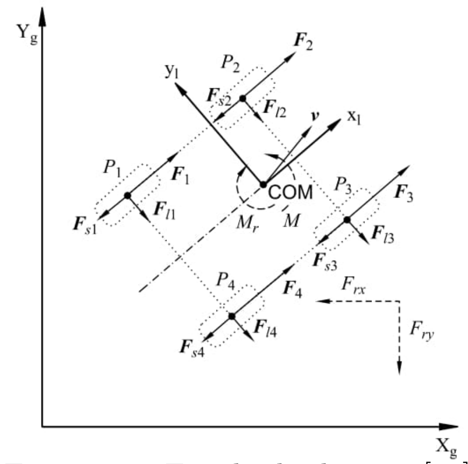


Figure 4.3: Free body diagram[13]

The free body diagram of the field robot in fig 4.3 shows the active and resistive forces which play a important role in actual motion of the mobile robot.

Reactive Forces In fig 4.4 reactive and active forces acting on wheel are shown. In this section the method of dynamic model description for skid-steer type mobile robot from [13] is recalled.

Normal Forces They result from the self weight of the mobile robot also called Normal force N_i as shown in fig 4.4.

$$\sum_{k=1}^{4} N_i = mg,$$

$$N_1 = N_4 = \frac{b}{2(a+b)}mg,$$
$$N_2 = N_3 = \frac{a}{2(a+b)}mg,$$

m is mass of the vehicle,
g acceleration due to gravity,
a, b, c are dimensions of the chassis as shown in fig 4.1b.

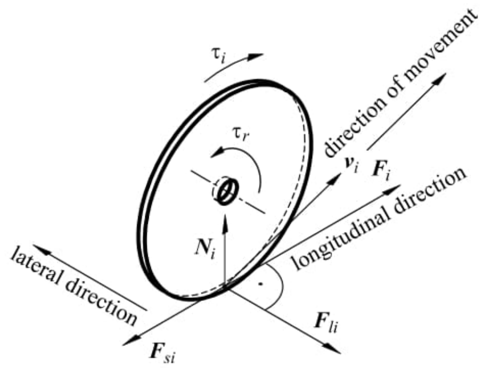


Figure 4.4: Forces acting,on wheel[13]

Friction forces *Longitudinal reactive force* F_{si} result from the rolling resistant moment τ_{ri} in response to active torque on wheel and ground surface interaction as shown in fig 4.4.
Lateral reactive force F_{li} result from the lateral skidding of the vehicle due to active torque on wheel and ground surface interaction as shown in fig 4.4.
based on coulomb friction force, F_{si} and F_{li} are written in [13] as,

$$F_{si} = \mu_{sci} mg\widehat{sgn}(v_{xi}),$$
$$F_{li} = \mu_{lci} mg\widehat{sgn}(v_{yi}),$$

where μ_{sci}, μ_{lci} are coefficients of friction in longitudinal and lateral directions respectively.

Active Forces result from the torque τ_i generated by actuators and are denoted by F_i as shown in fig 4.4. it is known know that,

$$F_i = \frac{\tau_i}{r},$$

active forces when expressed in inertial frame,

$$F_x = \cos\theta \sum_{k=1}^{4} F_i,$$

$$F_y = \sin\theta \sum_{k=1}^{4} F_i,$$

net torque resulting from the combination of active forces in inertial frame,

$$
\begin{aligned}
M &= c(\sum_{k=1}^{4} F_i), \\
M &= c(-F_1 - F_2 + F_3 + F_4),
\end{aligned}
$$

vector $\mathbf{F}$ with active forces, $\mathbf{F} = [F_x \ F_y \ M]^T$,
where r is the radius of the wheel.

Resistive Forces They are the forces which cause dissipation of energy [13].

$$
\begin{aligned}
F_{rx}(\dot{q}) &= \cos\theta \sum_{k=1}^{4} F_{si}(v_{xi}) - \sin\theta \sum_{k=1}^{4} F_{li}(v_{yi}), \\
F_{ry}(\dot{q}) &= \sin\theta \sum_{k=1}^{4} F_{si}(v_{xi}) + \cos\theta \sum_{k=1}^{4} F_{li}(v_{yi}),
\end{aligned}
$$

resistant moment M_r around center of mass,

$$
M_r(\dot{q}) = -a \sum_{i=1,4} F_{li}(v_{yi}) + b \sum_{i=2,3} F_{li}(v_{yi}) + c \left[- \sum_{i=1,2} F_{si}(v_{xi}) + \sum_{k=3,4} F_{si}(v_{xi}) \right].
$$

resistive forces vector, $R(\dot{q}) = [F_{rx}(\dot{q}) \ F_{ry}(\dot{q}) \ M_r(\dot{q})]^T$.

Inertial Forces To derive dynamic equations for a dynamic system *Newton's method* and *Lagrange's method* are commonly used. When dealing with dynamic systems with constraints *Lagrange's method* produces simple system equations, compared to *Newton's method* where every reactive force caused due to constraints needs to be modeled explicitly leading to a cumbersome set of equations.
To derive equations of motion using *Lagrange's method*, the Lagrangian, L is defined as., $L(q, \dot{q}) = T(q, \dot{q}) - V(q, \dot{q})$,
where T is Kinetic energy and V is Potential energy of the system both written in generalized coordinates.
then, equations of motion for a mechanical system is given by.,

$$
\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \Upsilon, \qquad (4.4.2.1)
$$

where Υ is external force acting on generalized coordinates.
The method followed here is recalled from [13], for a skid-steer mobile robot, potential

energy $V(q,\dot{q}) = 0$,as only planar motion is considered.
$L(q,\dot{q}) = T(q,\dot{q})$,
taking partial derivative of $L(q,\dot{q})$ which is equal to $T(q,\dot{q})$, Kinetic energy and then its time derivative, following 4.4.2.1, the following is obtained,

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{q}} = \begin{bmatrix} m\ddot{X} \\ m\ddot{Y} \\ I\ddot{\theta} \end{bmatrix} = M\ddot{q}$$

$$where M = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix} .$$

Equilibrium of forces Now by considering newton's laws of motion and equilibrium of forces a balance equation is obtained as follows,
$mass \times acceleration = net\, force\, acting\, on\, the\, body$,
$net\, force\, acting\, on\, the\, body = \sum F_x$, in x direction of the inertial frame,

$$M(q)\ddot{q} + R(\dot{q}) = \mathbf{F}(q). \tag{4.4.2.2}$$

4.4.3 Mathematical description of the Drive model

The drive model consists of BLDC motors, Gearbox, and a simple power transmission mechanism between two wheels on each side. In this section the method of drive model description for skid-steer type mobile robot from [13] is recalled and modify it according to the planned drive model of the prototype. Two BLDC motors, each on one side of the prototype, via a gearbox, actuate the wheels on each side which are linked with sprocket-chain as shown in fig 5.3.

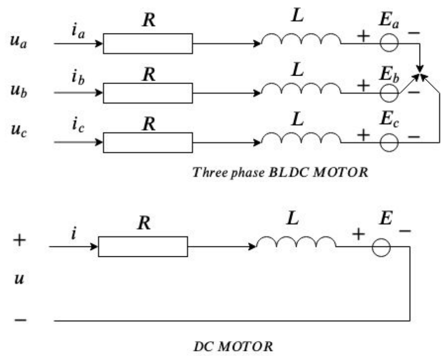


Figure 4.5: Circuit diagram of BLDC and
dc motor for comparison

Dynamics of the actuator BLDC motor and DC motors have similarities as shown in fig 4.5. Equations describing BLDC motor dynamics are,

$$u_a = L\frac{d}{dt}i_a + Ri_a + k_e\omega_m,$$
$$u_b = L\frac{d}{dt}i_b + Ri_b + k_e\omega_m,$$
$$u_c = L\frac{d}{dt}i_c + Ri_c + k_e\omega_m,$$
$$\tau_m = k_i(i_a + i_b + i_c).$$

Where $u_a, u_b, u_c$are the voltage difference in each phase respectively,
$i_a, i_b, i_c$are the current values in the phase respectively,
L is the inductance of the armature coil,
R is the resistance of the armature coil,
k_e is voltage constant of the motor,

k_i is current constant of the motor,

An electronic speed controller (ESC) controls the speed of the BLDC sensorless motor, and it does it by shifting voltage between three-phase lines (as shown in fig 4.5) within the BLDC motor by taking a Pulse width modulation(PWM) signal as input alongside a voltage source. Hence when an ESC is used the dynamic equations of BLDC motor can be simplified to a simple DC motor. Therefore equations given below are used in further development of the algorithm,
$u_a = L\frac{d}{dt}i_a + Ri_a + k_e\omega_m,$
$\tau_m = k_i i_a,$
Another important relationship between voltage constant and current constant parameters in case of BLDC motor is $k_i = k_e * \sqrt{3}$.

BLDC motor in combination with Gearbox Since the BLDC motor by itself cannot produce enough torque needed to suffice the purpose, a gearbox to step up the torque is used.
hence torque out of the shaft going into the wheel is calculated as,

$$\tau_i = Nk_i i_i,$$
$$\omega_i = \frac{\omega_m}{N},$$

where τ_i is torque at the wheel i,
ω_i is angular speed of wheel i,
N is the value from the gear ratio $1 : N$ of the gearbox.

Dynamic model of the whole drive system Control signal at voltage level is defined as,

$V_d = [V_{d1}\ V_{d2}]^T,$
where V_{d1}, V_{d2} are the voltage values needed for motors on left and right side of the prototype,
since the two wheels on each side of the prototype are mechanically coupled,

$$\begin{bmatrix}\omega_L\\ \omega_R\end{bmatrix} = \frac{1}{N} \begin{bmatrix}\omega_{m1}\\ \omega_{m2}\end{bmatrix},$$

where ω_L and ω_R are the angular speeds of wheels on left side and right side of the wheels respectively.
the final voltage current equations considering the whole drive model are,

$$V = L\frac{d}{dt}i + Ri + k_e N\omega_w,$$
$$\tau = \frac{k_i i N}{2},$$

as the torque from one motor and gearbox combination is being distributed between two wheels on one side of the mobile robot prototype.

4.5 Control system Design for Mobile Robot

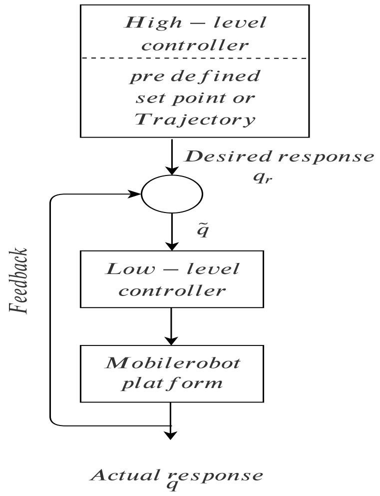


Figure 4.6: Contol System Architecture

"Robot control deals with the problem of determining the forces and torques that must be developed by the robotic actuators for the robot to go at a desired position, track the desired trajectory, and, in general, to perform some task with desired performance requirements",[17]. Considering the kinematic and dynamic behavior of the mobile robot, and the control objectives such as pose tracking and trajectory tracking, a suitable control strategy has to be chosen. Control strategy comprises multiple control laws that are designed keeping in mind achievability of the task, stability of the controller and the system being controlled. The fig 4.6 gives an overview of the whole system flow, including High-Level and Low-Level controllers being implemented on a mobile robot. In following sections, few topics which help understand control system design are introduced.

4.5.1 Control Strategy

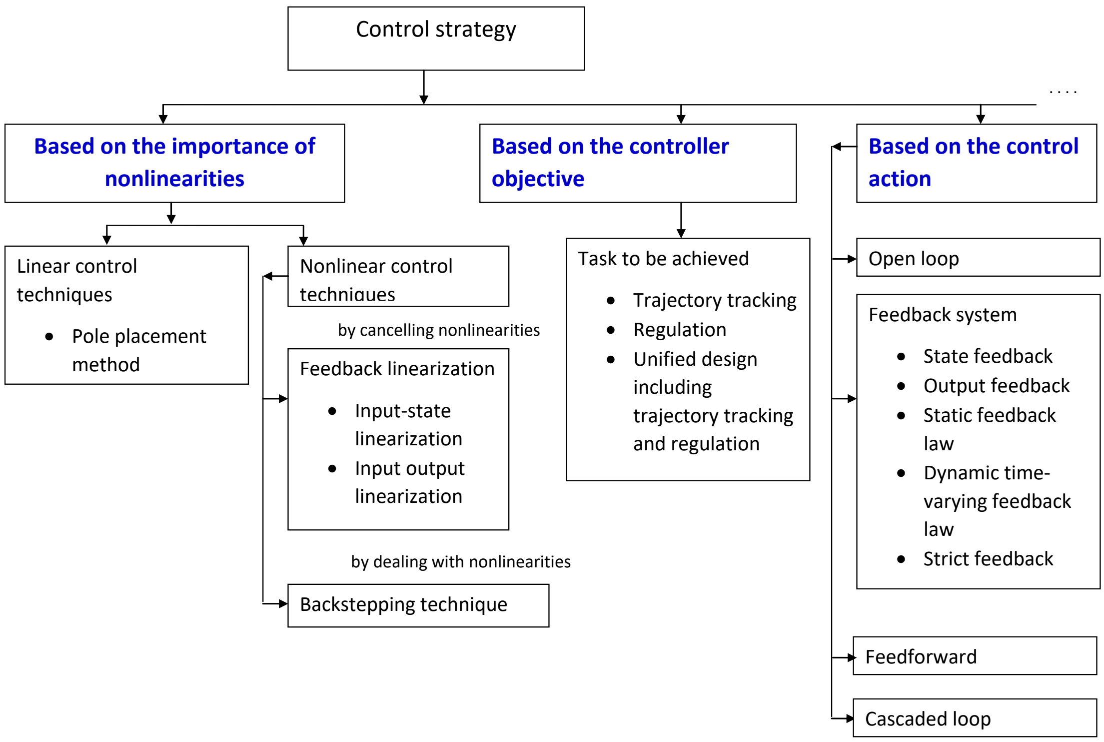


Figure 4.7: Control strategy possibilities.

The fig 4.7 outlines few key points which can help to decide a particular control strategy. The diagram is a result of the author's understanding and research and may not conform to any established standard.

4.5.2 Controller Stability Criteria

Depending on the system being autonomous or non-autonomous the stability study is made based on either *Routh-Hurwitz* or *Lyapunov's* stability criteria respectively. [17]. An Autonomous system is not explicitly time dependent, with form $\dot{x} = f(x, u)$ where as a non-autonomous system is explicitly time dependent and is of form, $\dot{x} = f(x(t), u)$.
In case of nonlinear, non-autonomous system like a skid-steer type mobile robot, *Lyapunov's stability criteria* is widely used. In context of controller design for a dynamic system based on *Lyapunov's stability criteria* the following stability properties at equilibrium state $x = 0$ are defined in [17, p 145] as,

Lyapunov-Stable, if the free system $\dot{x} = A(t)$ at equilibrium state $x = 0$ is stable in Lyapunov sense, if for every initial time t_0 and every real number $\varepsilon > 0$, there exists some number $\delta > 0$ as small as desired, that depends on t_0 and ε, such that: if $||x_0|| < \delta$, then $||x(t)|| < \varepsilon$ for all $t \geq t_0$, where $||.||$ denotes the norm of the vector x, that is, $||x|| = (x_1^2 + x_2^2 + ... + x_n^2)^{\frac{1}{2}}$ [17].

Global stability, when the system stability does not depend on initial condition x_0.

Local stability, when the system stability depends on initial condition x_0.

Asymptotically stable, when system is Lyapunov-Stable and for every t_0 and x_0 sufficiently close to $x = 0$, the condition $x(t) \to 0$, for $t \to \infty$ holds.

Uniform Lyapunov-Stable, if the parameter δ does not depend on t_0.

Uniform Asymptotically stable, if the system is uniformly lyapunov stable, and for all t_0 and for arbitarily large ρ, the relation $||x_0|| < \rho$ implies $x(t) \to 0$ for $t \to \infty$.

Theorem "The linear system $\dot{x} = A(t)$ is *uniformly asymptotically stable,* if and only if there exist two constant parameters k_1 and k_2 such that:$||\Phi(t, t_0)|| \leq k_1 e^{-k_2(t-t_0)}$ for all t_0 and all $t \geq t_0$ [17]. This theorem is used in designing a unified kinematic controller for trajectory tracking and regulation in [13].

Unstable, if for some real number $\varepsilon > 0$, some $t_1 > t_0$ and any real number δ arbitrarily small, there always exists an initial state $||x_0|| < \delta$ such that $||x(t)|| > \varepsilon$ for $t \geq t_1$.

figure 4.8 geometrically illustrates the concepts of *Lyapunov stability, Asymptotic stability, and instability.*

4.5.3 Error dynamics

Error dynamics equations describe the evolution of the error $\tilde{q}$ which is defined as $\tilde{q} = q_d - q$ for the control system, where$q$ and q_d are actual position vector and desired position vector of the mobile robot on a plane where $q^T = [xy\theta]$. As transformation of error coordinates for which global diffeomorfism that preserves origin was possible in the current project controller design based on [13], the error dynamics in terms of transformed error coordinates $Z = \left[w\, z_1\, z_2^T\right]$, obtained from $Z = P(\theta, \dot{\theta})\tilde{q}$ is analyzed.

[1]please refer [12, p 5]

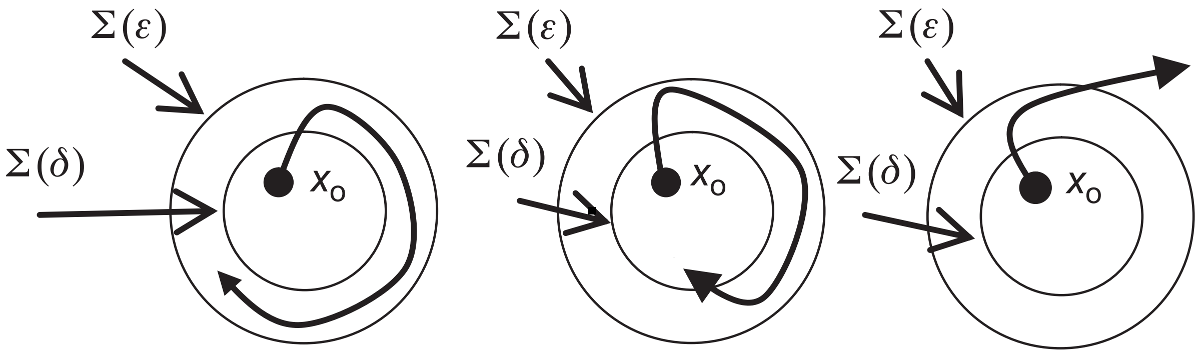


Figure 4.8: Lyapunov stable(left), Asymptotic stability(center) and instability(right), where x_0 is initial state and the arrow shows change in state as time goes to infinity, $\sum(\varepsilon)$ and $\sum(\delta)$ symbolize n-dimensional sphere [1] [17].

Error convergence and its rate A good controller design will drive the error to zero or close to zero as quick as possible.

Error dynamics stability type , is characteristic behavior of the error which can be stable, asymptotically stable, exponentially stable, unstable few to mention.
Controller performance can be validated with the help of the *Error response plot* 4.9.

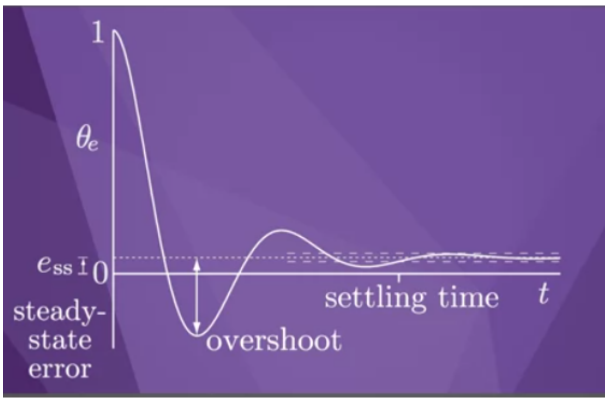


Figure 4.9: Error response plot of orientation error θ_e example with few properties [1].

Best controller performance is achieved when minimum steady-state error response e_{ss} and transient error response which constitutes overshoot and settling time.
One way of introducing desired state performance alongside the actual state to derive error dynamic equations in terms of state error is *Feedback linearization*. But as a drawback *Feedback linearization* cancels the plant dynamics and insert desired tracking error dynamics, by doing so *model error* is introduced [10, p 10]. In [10, p 10] alternative methods which dominate model error like mentioned below are elaborately explained.

- Backstepping,

- Robust Nonlinear Control Design Methods.

4.6 Constraint optimization with Lagrange multiplier

4.6.1 Constraint Optimization

"A mathematical **optimization problem**, or just optimization problem, has the form *minimize or maximize etc.*, $f_0(x)$ *subject to*

$$f_i(x) \leq b_i, i = 1, ..., m. \tag{4.6.1}$$

[3] Here the vector $x = (x_1, ..., x_n)$ is the *optimization variable* of the problem, the function $f_0 : R^n \rightarrow R$ is the ***objective function***, the functions $f_i : R^n \rightarrow R, i = 1, ..., m$, are the (inequality) ***constraint functions***, and the constants $b_1, ..., b_m$ are the limits, or bounds, for the constraints. A vector x^* is called optimal, or a solution of the problem (4.6.1), if it has the smallest objective value among all vectors that satisfy the constraints: for any z with $f_1(z) \leq b_1, ..., f_m(z) \leq b_m$, we have $f_0(z) \geq f_0(x^*)$"
above definition of an optimization problem is taken from [3].
In the context of dynamic systems analysis and control which includes mobile robotics as a specific application, very often constraint optimization problems naturally appear. Especially when dealing with specific tasks of mobile robot like motion and path planning. The type of wheels used in the current mobile robot prototype are standard fixed wheel type as depicted in 3.2. For simplicity many research papers assume two common type of constraints on mobile robot wheel of fixed wheel type 3.2 they being pure rolling (no-slip) constraint and sliding constraint (no-lateral skid).

- ***Rolling constraint*** enforces that there is pure rolling at the contact point. leading to the rolling constraint.,
  $v_x = r\omega$,
  please refer 3.2
- ***sliding constraint*** enforces that there is no lateral skid i.e, no orthogonal motion w.r.t the motion along the planned path for the wheel. leading to the sliding constraint.,
  $v_y = 0$,
  please refer 3.2

But in case of skid-steer type mobile robot lateral skidding is inevitable. Hence the zero sliding constraint is not valid. In [5] an ***operational nonholonomic constraint*** is introduced based on the observation that x_{ICR} x-axis projection of instantaneous center of rotation cannot be outside the interval $x_{ICR} = x_0$, $x_0 \in (-a, b)$ 4.10. The reason being that as x_{ICR} goes beyond the interval the vehicle skids along y-axis and looses control. For vehicle to not loose control the following condition should be valid,
$a > | - \frac{v_y}{\dot{\theta}}| > -b$,

Hence the following operative constraint is introduced in [5],[13]
$v_y + x_0\dot{\theta} = 0$. [13]
or in terms of generalized coordinates.,

$$[-\sin\theta \ \cos\theta \ x_{ICR}] \left[\dot{X} \ \dot{Y} \ \dot{\theta}\right]^T = \mathbf{A(q)\dot{q} = 0}, \tag{4.6.2}$$

Path planning for mobile robot in other words is an constraint optimization problem.

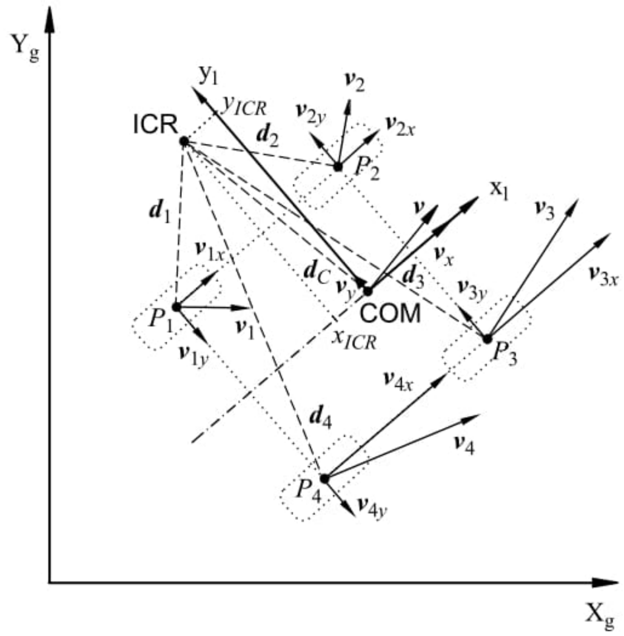


Figure 4.10: ICRprojection on x-axis [13]

4.6.2 Dynamic modeling with constraints

The dynamic equation of mobile robot 4.4.2.2 derived earlier does not include the nonholonomic constraint 4.6.2. To get deeper understanding of Lagrange's multipliers in context of optimization problem with equality constraint please refer [3, p. 141].
In below equation 4.6.2.1 the nonholonomic constraint is attached to the dynamic equation of mobile robot derived earlier. This is done based on the well known theorem called ***Method of Lagrange's multipliers***.

$$M(q)\ddot{q} + R(\dot{q}) = \mathbf{F}(q) + A^T(q)\lambda \tag{4.6.2.1}$$

4.7 Parameter estimation using Curve fitting

Very often manufacturers of the electro-mechanical components like motors etc, do not provide parameters relevant to mathematically model such system. In such cases experimental modeling is the way to go. Where input-output data of the system is collected and analyzed. Curve fitting using least-squares is one of the commonly used methods to

obtain a mathematical description of the system.
In the current project BLDC motor Orbit 25-16 3.1 is used. Voltage vs Speed and current vs torque data are provided by the manufacturers. On request to manufacturers, info in table 4.2 is provided.

Table 4.2: BLDC motor Orbit 25-16 from Plettenberg Elektromotoren Parameters.

Resistance mOhm	**Inductance** μH	**no-load rpm/volt** rpm/V	**Number of poles** -
32	65	585	10

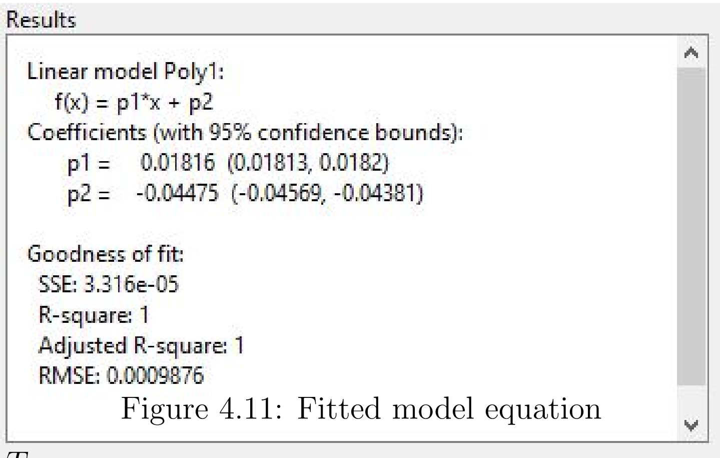


Figure 4.11: Fitted model equation

For mathematical description of the BLDC motor, parameters like motor current constant k_i, motor voltage constant k_e, are estimated using the curve fitting tool from MATLAB. Fig 4.12 shows the fitted curve, and the polynomial function representing the fitted curve is $f(x) = 0.01816 * x - 0.04475$ as seen in fig 4.11.
Based on the above equation $k_i = \frac{Torque}{current} \approx 0.01816$. As discussed earlier using the relationship between k_i and k_e i.e, $k_i = k_e * \sqrt{3}$, the value k_e is obtained.

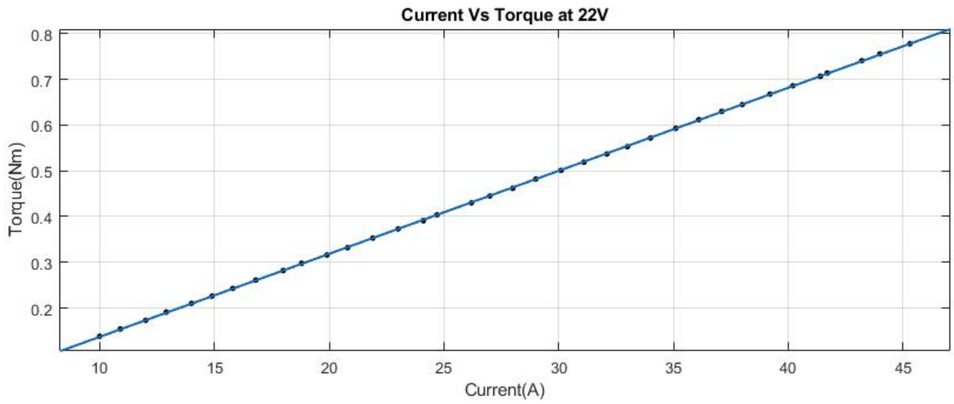


Figure 4.12: Curve-Fitting based on Input-Output data

4.8 Regression Matrix estimation

Multiple linear regression based on least-squares estimation method is used to estimate a regression matrix from the available data.
In terms of the current project, the dynamic model as function of desired control signal

u_d is linearly parameterized in the following form in [13],

$$\overline{\overline{M}}\dot{u}_d + \overline{\overline{C}}u_d + \overline{\overline{R}} = Y_d(u_d, \dot{u}_d, \tilde{q}, \theta, \eta_r)\vartheta, \tag{4.8.1}$$

where $\vartheta = [m\ I\ \mu_s m\ \mu_l m]^T$, is vector of dynamical parameters, and $Y_d(u_d, \dot{u}_d, \tilde{q}, \theta, \eta_r)$ denotes a regression matrix to be estimated.
μ_s, μ_l, friction coefficients are assumed constant through out the terrain for this project. Although using Model-Based Terrain Identification friction coefficients are dynamically obtained in [20].
Using MATALB command $b = regress(y, X)$, the regression matrix Y_d is obtained.
where L.H.S of eq 4.8.1 is taken as y of size $p \times 2$, ϑ as X of size $p \times 4$ and obtained b of size 4×2 is Y_d^T.

4.9 Numerical Methods for Embedded System

Once the control law is designed and simulation results are satisfactory, the designed controller will have to be programmed into a processor. The process is called digitization. The following points taken from [6] guide us in digital implementation of the controller.

- Determine the sampling period Ts and the number of bits used in analog-to-digital converter (ADC) and digital-to-analog converter (DAC)
- Convert continuous-time transfer function to discrete-time form
- Derive the difference equations

Conversion of a continuous-time system to discrete-time form is done using *Numerical Methods*.

- Euler's forward and backward methods
- Trapezoidal method or bilinear transformation
- Runge–Kutta methods

Chapter 5

Algorithm Design for Field robot controller by Integrating the above-learned Theory and Methods

Mobile robot controller algorithm design process is the result of numerous sub-tasks where the transfer of results from one sub-task to another happens discretely at a certain frequency. The fig 5.1 gives an idea of the various sub-tasks researched during the design of the controller for the *Regulation problem* objective. This algorithm can be extended to *Trajectory tracking* with a few changes. Backward Euler numerical method is used in the following algorithm for simplicity, although a more effective numerical method like Runga-Kutta shall be used for actual implementation on the Embedded system.

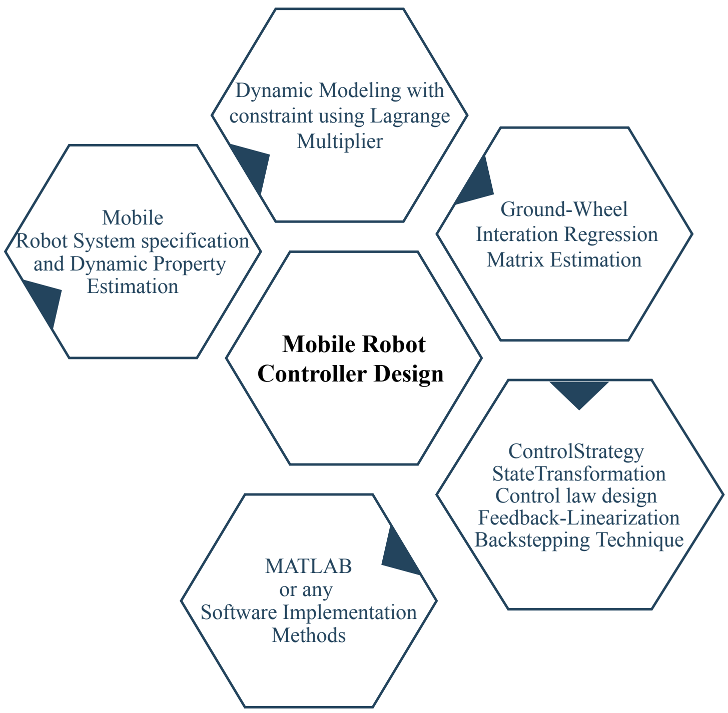


Figure 5.1: Sub-Tasks which make up the Algorithm

5.1 Controller Simulation Approach

Obtaining a mathematical model of the actual system to be controlled is not always an easy task, depending on the knowledge of the system multiple approaches to obtain the mathematical model is possible. The fig 5.2 gives an outline of the authors understanding for possible approaches. It also gives an overview of the controller Simulation process. Blue arrows indicate the actual approach taken while designing the current controller for the mobile robot.

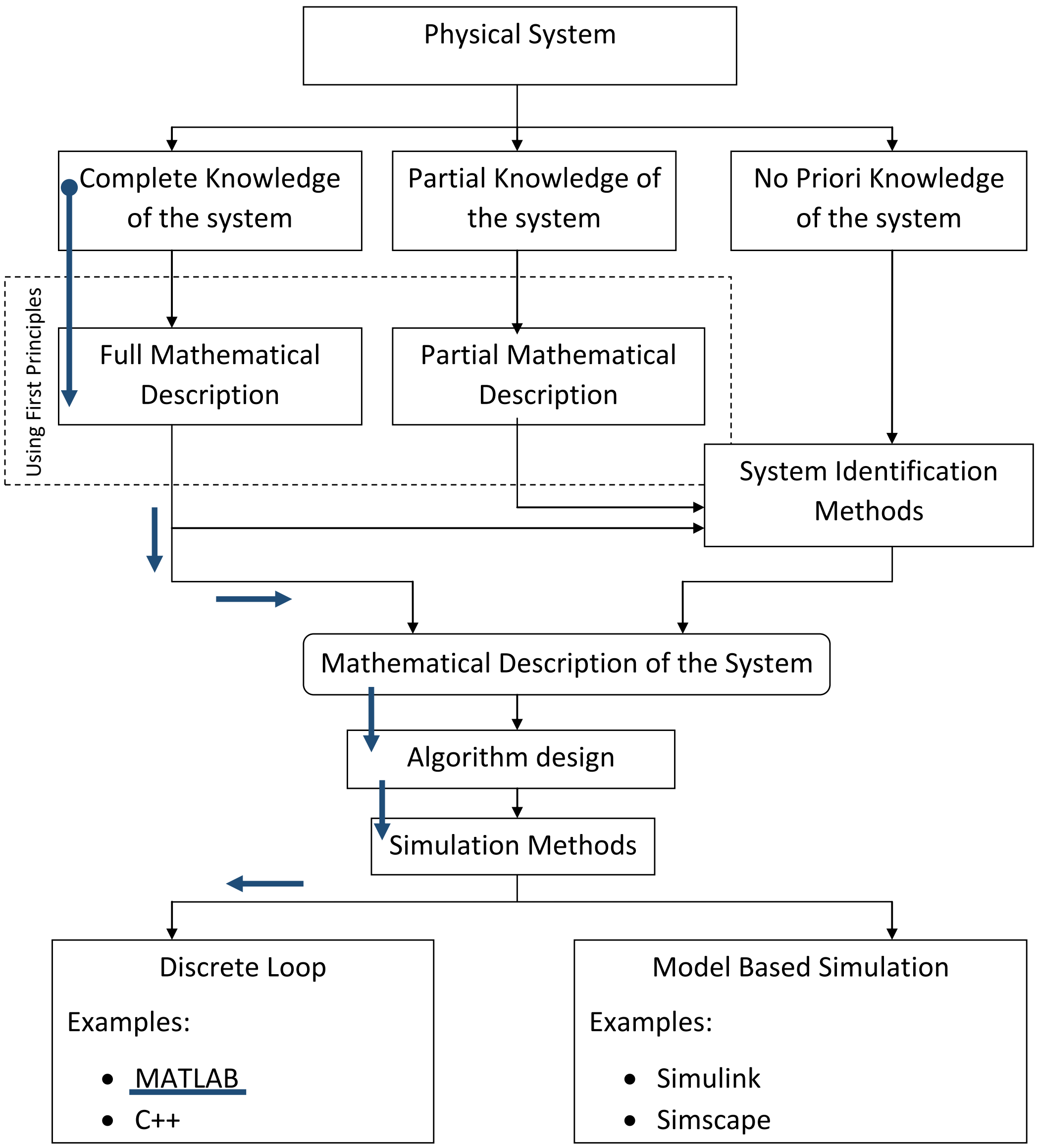


Figure 5.2: Overview of Controller Simulation

5.2 Controller Framework

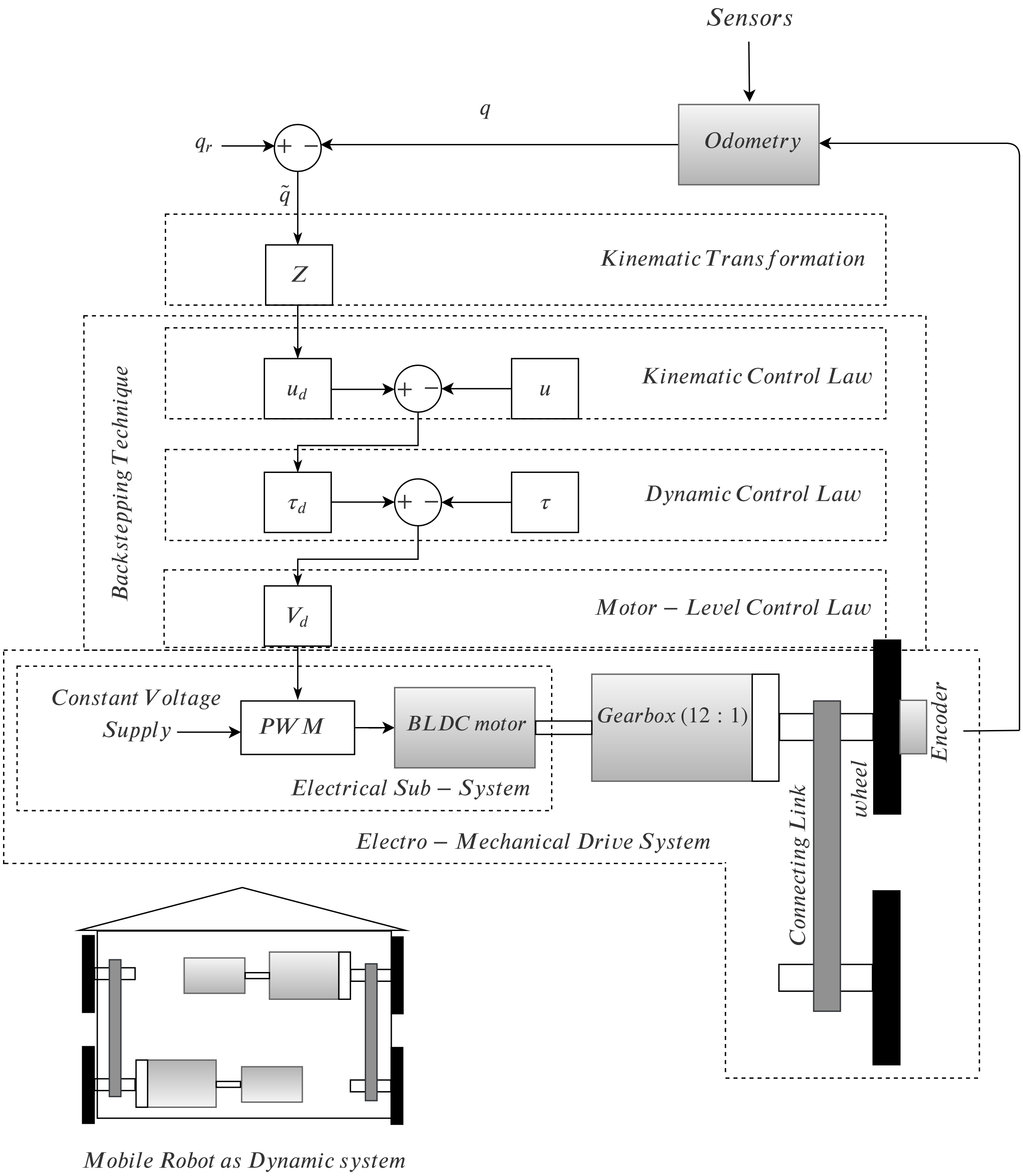


Figure 5.3: Controller Framework

$$Z = P(q, \tilde{q})[13, equation\ 76]$$
$$u_d = f(Z) = u_a - k_2 z[13, equation\ 98]$$
$$\tau_d = g(u, u_d) = \overline{\overline{B}}^{-1}(wJz + \tilde{z} + Y_d\vartheta_0 + \tau_a + k_3\tilde{u}), [13, equation\ 139]$$
$$V_d = h(\tau, \tau_d) = L(k_i N)^{-1}(\dot{\tau}_d + k_4\tilde{\tau} + \overline{\overline{B}}^T\tilde{u}) + Ri + k_e N\omega_w[13, equation\ 145]$$

Fig 5.3 shows multiple sub-systems among which flow of state information is shown, it also outlines the flow of state information in the context of **Backstepping technique**, where there is a known stable subsystem at motor level, upon which dynamic level and kinematic level subsystems are progressively stabilized [14]. The process of stabilizing, the initial subsystem at voltage level and layers of subsystems upon the initial one continue until the final external command is achieved up to the preset proximity.
Outline of the Mobile Robot showing its drive system located on the chassis can also be seen in the fig 5.3.

5.3 MATLAB Code used for simulating the controller.

Controller terms representatives in MATLAB code

x_{ICR}=j1 Instantaneous center of projection
μ_l=ul lateral coefficient of friction
μ_s=us longitudinal coefficient of friction
δ=delta
τ = Torque
η=velB
η_r=ref velB
Π=PP
S(q)=Sq refer [13, equation 62]
Z=auxerror
$\frac{\dot{\delta}_d}{\delta_d}$=PCoeff
$\frac{k_1 w+f}{\delta_d^2}$=QCoeff
$\Omega 1$=Om1
ϑ_0=V dynamic parameter vector
i=Ad current

```
clear
n=500;                  %Number of time steps
tlim=60;                %time limit

% control law parameters section
alpha0=2.7;
alpha1=0.2;
```

```
eps1=0.01;
eps2=0.02;
k1=0.5;
k2=0.5;
k3=1;
k4=5;
rho=1;
phi=-7*pi/12;
%mobile robot dimensions and specifications
m =12;              %mass in kg
I = 0.18;           %moment of inertia of the mobile robot kg/m^2
L=0.000065;        %inductance in H
R=0.032;           %resistance in ohm
j1=-0.1;            %ICR projection along x direction in the robot frame or local frame
a=0.3;              %should be refered to in fig:4.1(b)in meters
b=0.3;              %should be refered to in fig:4.1(b)in meters
r = 0.0765;         %radiusof wheel in meters
c = 0.154;          %should be refered to in fig:4.1(b)in meters
J =[0 -1; 1 0];
g=9.8;              %acceleration of gravity in m/sec^2

%time steps
t(1,n+1)=0;         %Initializing time
for i=1:n+1
t(1,i)=tlim/n*(i-1);
end
dt=t(1,2)-t(1,1); %size of time step



% Normal forces on each wheel------
N1=(m*g*b)/(2*(a+b));
N4=(m*g*b)/(2*(a+b));
N2=(m*g*a)/(2*(a+b));
N3=(m*g*a)/(2*(a+b));
ul=0.15;                    %lateral friction coefficent
us=0.34;                    %longitudinal orslip friction
N=12;                       %because of the gear reduction ratio of 12:1
ki=0.018;                   %estimated motor current constant
ke=ki/sqrt(3);              %estimated motor voltage constant

%initialization-------------------
deltad(1,n)=0;
deltad_dot(1,1)=-alpha1*(deltad(1,1)-eps1); %rate of change of intermediate auxilary term deltad
ref_velB(1:2,n+1)=0;        %reference velocity
```

```
q(1:3,1)=[0 0 0]';          %the actual initial position of the robot
q_dot(1:3,1)=0;             %global frame velocity of the robot center
qr=[0 1 0]';                %the final goal position
Tq(1:2,1)=[1 1]';           %torque in Nm
Tqd_dot(1:2,1)=0;           %rate of change of torque
velB(1:2,1)=0;              %local frame velocity of robot
ud_dot(1:2,1)=0;            %rate of change of control signal ud
l_dot(1,1)=0;               %rate of change of intermediate auxilary term l
T_dot(1:2,1:2,1)=0;         %rate of change of intermediate auxilary term T
PP_dot(1:2,1)=0;            %rate of change of intermediate auxilary term PP
Sq_dot(1:3,1:2,1)=0;        %rate of change of intermediate auxilary term Sq
q_hat(1:3,1)=0;             %pose error
l(1,1)=0;                   %auxilary term l
    T(1:2,1:2,1)=0;             %auxilary matrix
    P(1:3,1:3,1)=0;             %state transformation matrix
    auxerror(1:3,1)=0;          %Transformed error
    z(1:2,1)=0;                 %transformed error z1 z2
    PP(1:2,1)=0;               %aux term
    PCoeff(1,1)=0;            %aux term
    f(1,1)=0;                %drift term
    QCoeff(1,1)=0;            %intermediate auxilary term QCoeff
    Om1(1,1)=0;              %intermediate auxilary term Om1
    v=0;
%     zd1Sol(v)={};
%     zd2Sol(v)={};
    zD1(1,1)=0;
    zD2(1,1)=0;
    zd(1:2,1)=0;
    z_hat(1:2,1)=0;
    ua(1,1)=0;
    ua(2,1)=0;%time varying feedback modulated by zd
    ud(1,1)=0;
    ud(2,1)=0;
    %friction forces ref fig 4.4 and fig 4.3 -
    fl1(1,1)=0;%longitudinal friction force component
    fl2(1,1)=0;
    fl3(1,1)=0;
    fl4(1,1)=0;
    fs1(1,1)=0;%lateral friction force component
    fs2(1,1)=0;
    fs3(1,1)=0;
    fs4(1,1)=0;
```

```
    Frx(1,1)=0;
    Fry(1,1)=0;
    Mr(1,1)=0;
%------------------------------------------------
    wv(1:4,1)= 0;%wheelvelocities of the robot
    awv(1:2,1)=0;%angular velocities of left and right side wheels
    vx(1:4,1)=0;%xcomponent of the wheel velocity
    vy(1:4,1)=0;%ycomponent of the wheel velocity
%M_,C_,B_,R_------------------------------------
    M(1:3,1:3,1)=0;
    Sq(1:3,1:2,1)=0;
    Sq_dot(1:3,1:2,1)=0;
    C_(1:2,1:2,1)= 0;
    M_(1:2,1:2,1)=0;
    R_(1:2,1)=0;
    B_(1:2,1:2,1)=0;
    B_Tq(1:2,1)=0;
%M__,C__,R__,B__ ----------------------------------
    M__(1:2,1:2,1)=0;
    C__(1:2,1:2,1)=0;
    R__(1:2,1)=0;
    B__(1:2,1:2,1)=0;
%solving for actual u signal, as a result actual Tq (torque) from motor---
    InvM__(1:2,1:2,1)=0;
    ac(1:2,1:2,1)=0;
    br(1:2,1)=0;
    cb(1:2,1)=0;
    g=0;
%     u1Sol={};
%     u2Sol={};
    U1(1,1)=0;
    U2(1,1)=0;
    u(1:2,1)=0;
    u_hat(1:2,1)=0;
%error -difference between actual u signal and desired u (ud) signal
%dynamic control law implementation-----------
    ud_dot(1:2,1)=0;%rate of change of desired u signal
    Y(1:2,1:4,1)=0;                          %Y is regression matrix
    YV1(1:2,1)=0;
%divided YV signal into  YV1 and YV2 to observe influence of each part on YV
    YV2(1:2,1)=0;
    YV(1:2,1)=0; %Y is regression matrix and V is vector of dynamical properties
```

```
    V=[m I us*m ul*m]';
    magnt(1,1)=0;%auxilary term
    taua1(1:2,1)=0;
%auxilary term taua sectioned to observe its influence
    taua(1:2,1)= 0;
    %auxilary term taua
    InvB__(1:2,1:2,1)=0;
    Tqd1(1:2,1)=0;
    %Tqd sectioned to observe its terms influence
    Tqd2(1:2,1)=0;
    Tqd3(1:2,1)=0;
    Tqd4(1:2,1)=0;
    Tqd5(1:2,1)=0;

    Tqd(1:2,1)= 0; %desired torque to be generated on the wheel

%motor level voltage control law--------------
    Tqd_dot(1:2,1)=0;
    T_hat(1:2,1)=0;
    %error - difference between desired torque and actual torque

    Ad(1,1)=0;
    %desired current in each motor to produce appropriate torque Tq

    Ad(2,1)=0;

    angvelwheelRL(1:2,1)=0; %angular velocities of wheels on left and right side of mobile robot
    Vd1(1:2,1)=0;%Vd sectioned into Vd1, Vd2, Vd3 to observe its terms influence
    Vd2(1:2,1)=0;
    Vd3(1:2,1)=0;
    Vd(1:2,1)=0;
%---------------------------------------------
for i=1:n+1
    deltad(1,i)=alpha0*exp(-alpha1*t(1,i))+eps1; %auxilary term
end
deltad_dot(1,2:n+1)=diff(deltad)/dt;
%rate of change of intermediate auxilary term deltad

figure ;%start figure window
hold on;

for j=1:n
    q_hat(1:3,j)=q(1:3,j)-qr;       %error signal
    l(1,j)=q_hat(1,j)*sin(q(3,j))-q_hat(2,j)*cos(q(3,j)); %auxilary term l
```

```
    T(1:2,1:2,j)= [l(1,j)  1;1  0];    %auxilary term T
     if j>=2
    l_dot(1,j)=(l(1,j)-l(1,j-1))/dt;
    %rate of change of intermediate auxilary term l
     T_dot(1:2,1:2,j)=(T(1:2,1:2,j)-T(1:2,1:2,j-1))/dt;
    %rate of change of intermediate auxilary term T
     end
    P(1:3,1:3,j) =[-q_hat(3,j)*cos(q(3,j))+2*sin(q(3,j)) -q_hat(3,j)*sin(q(3,j))-2*cos(q(3,j)) -2*j1;
         0  0  1;cos(q(3,j)) sin(q(3,j)) 0];
    %state transformation matrix for more theory ref section 4.2
     auxerror(1:3,j)=P(1:3,1:3,j)*q_hat(1:3,j);
    % Transformed error
     z(1:2,j)=auxerror(2:3,j);
    PP(1:2,j)=[cos(auxerror(2,j)),(-j1*sin(auxerror(2,j))+l(1,j));0,1]*ref_velB(1:2,j); %intermediate auxilary term PP
     if j>=2
      PP_dot(1:2,j)=(PP(1:2,j)-PP(1:2,j-1))/dt;
      %rate of change of intermediate auxilary term PP
     end

%kinematic control law implementation--
     PCoeff(1,j)=deltad_dot(1,j)/deltad(1,j);
    %intermediate auxilary term PCoeff
     f(1,j)=2*[-sin(auxerror(2,j))   (j1+auxerror(3,j)-j1*cos(auxerror(2,j)))]*ref_velB(1:2,j);%drift term
     QCoeff(1,j)=((k1*auxerror(1,j)+f(1,j))/deltad(1,j)^2);
    %intermediate auxilary term QCoeff
    Om1(1,j)= k2+PCoeff(1,j)+auxerror(1,j)*QCoeff(1,j);
    %intermediate auxilary term Om1

    syms zd1(v) zd2(v)
      zd1Sol= symvar(zd1(v));
      zd2Sol= symvar(zd2(v));
    %solving nonstationary differential equation
     eqns = [diff(zd1,v)==PCoeff(1,j)*zd1-(QCoeff(1,j)+auxerror(1,j)*Om1(1,j))*zd2, diff(zd2,v)==PCoeff(1,j)*zd2+(QCoeff(1,j)+auxerror(1,j)*Om1(1,j))*zd1];
    cond=[zd1(0)==deltad(1,1)*cos(phi);zd2(0)==deltad(1,1)*sin(phi)];%deltad(1,1)*cos(phi)for zd1(0);   for zd2(0)deltad(1,1)*sin(phi);
     [zd1Sol(v), zd2Sol(v)] = dsolve(eqns,cond);
     g1=matlabFunction(zd1Sol(v));
     g2=matlabFunction(zd2Sol(v));
    zD1(1,j)=g1(t(1,j));
    zD2(1,j)=g2(t(1,j));
     zd(1:2,j)=[zD1(1,j);zD2(1,j)];%desired auxilary signal zd

```

```
    z_hat(1:2,j)=zd(1:2,j)-z(1:2,j);%error between desired aux signal obtained from a tunable oscillator and transformed error z
    ua(1,j)=-PCoeff(1,j)*zd(2,j)+Om1(1,j)*zd(1,j);
    ua(2,j)=PCoeff(1,j)*zd(1,j)+Om1(1,j)*zd(2,j);%time varying feedback modulated by zd
    ud(1,j)=ua(1,j)-k2*auxerror(2,j);
    ud(2,j)=ua(2,j)-k2*auxerror(3,j);

%--------------------------------------------------------------
    wv(1:4,j)= [1 -c;1 c;0 -j1+b;0 -j1-a]*velB(1:2,j);%wheelvelocities of the robot
    awv(1:2,j)=(1/r)*[wv(1,j);wv(2,j) ];%angular velocities of left and right side wheels

    vx(1:4,j)=[wv(1,j); wv(1,j); wv(2,j); wv(2,j)];%xcomponent of the wheel velocity
    vy(1:4,j)=[wv(4,j); wv(3,j); wv(3,j); wv(4,j)];%ycomponent of the wheel velocity


%friction forces ref fig 4.4 and fig 4.3 -
    fl1(1,j)=ul*N1*sign(vy(1,j));%longitudinal friction force component
    fl2(1,j)=ul*N2*sign(vy(2,j));
    fl3(1,j)=ul*N3*sign(vy(3,j));
    fl4(1,j)=ul*N4*sign(vy(4,j));

    fs1(1,j)=us*N1*sign(vx(1,j));%lateral friction force component
    fs2(1,j)=us*N2*sign(vx(2,j));
    fs3(1,j)=us*N3*sign(vx(3,j));
    fs4(1,j)=us*N4*sign(vx(4,j));

%resistive forces----------------------------------------
    Frx(1,j)=cos(q(3,j))*(fs1(1,j)+fs2(1,j)+fs3(1,j)+fs4(1,j))-sin(q(3,j))*(fl1(1,j)+fl2(1,j)+fl3(1,j)+fl4(1,j));
    Fry(1,j)=sin(q(3,j))*(fs1(1,j)+fs2(1,j)+fs3(1,j)+fs4(1,j))+cos(q(3,j))*(fl1(1,j)+fl2(1,j)+fl3(1,j)+fl4(1,j));
    Mr(1,j)=-a*(fl1(1,j)+fl4(1,j))+b*(fl2(1,j)+fl3(1,j))+c*(-(fs1(1,j)+fs2(1,j))+(fs3(1,j)+fs4(1,j)));

%M_,C_,B_,R_----------------------------------------------
    M(1:3,1:3,j)=[m 0 0;0 m 0;0 0 I];
    Sq(1:3,1:2,j)=S(q(3,j));
    if j>=2
        Sq_dot(1:3,1:2,j)=(Sq(1:3,1:2,j)-Sq(1:3,1:2,j-1))/dt;
    end
```

```
    C_(1:2,1:2,j)= Sq(1:3,1:2,j)'*M(1:3,1:3,j)*Sq_dot(1:3,1:2,j);
    M_(1:2,1:2,j)=[m 0;0 m*j1^2+I];
    R_(1:2,j)=[Frx(1,j);(j1*Fry(1,j))+Mr(1,j)];
    B_(1:2,1:2,j)=1/r*[1 1; -c c];

    B_Tq(1:2,j)=B_(1:2,1:2,j)*Tq(1:2,j);

%M__,C__,R__,B__ ----------------------------------------
    M__(1:2,1:2,j)= T(1:2,1:2,j)'*M_(1:2,1:2,j)*T(1:2,1:2,j);
    C__(1:2,1:2,j)=T(1:2,1:2,j)'*(C_(1:2,1:2,j)*T(1:2,1:2,j)+M_(1:2,1:2,j)*T_dot(1:2,1:2,j));
    R__(1:2,j)=T(1:2,1:2,j)'*(C_(1:2,1:2,j)*PP(1:2,j)+M_(1:2,1:2,j)*PP_dot(1:2,j)+R_(1:2,j));
    B__(1:2,1:2,j)=T(1:2,1:2,j)'*B_(1:2,1:2,j);

%solving for actual u signal, as a result actual Tq (torque) from motor---
    InvM__(1:2,1:2,j)=pinv(M__(1:2,1:2,j));

    ac(1:2,1:2,j)=InvM__(1:2,1:2,j)*C__(1:2,1:2,j);
    br(1:2,j)=InvM__(1:2,1:2,j)*R__(1:2,j);
    cb(1:2,j)=InvM__(1:2,1:2,j)*B__(1:2,1:2,j)*Tq(1:2,j);

    syms u1(g) u2(g)
    u1Sol=symvar(u1(g));
    u2Sol=symvar(u2(g));
    eqns = [diff(u1,g)==ac(1,1,j)*u1+ac(1,2,j)*u2+cb(1,j)+br(1,j), diff(u2,g)==ac(2,1,j)*u1+ac(2,2,j)*u2+cb(2,j)+br(2,j)];
    cond=[u1(0)==ud(1,1);u2(0)==ud(2,1)];
    [u1Sol(g), u2Sol(g)] = dsolve(eqns,cond);
    r1=matlabFunction(u1Sol(g));
    r2=matlabFunction(u2Sol(g));

    if j==1
        U1(1,1)=ud(1,1);
        U2(1,1)=ud(1,1);
        u(1:2,1)=[U1(1,1);U2(1,1)];
    end
    if j>=2
        U1(1,j)=r1(t(1,j));
        U2(1,j)=r2(t(1,j));
        u(1:2,j)=[U1(1,j);U2(1,j)];
    end

    u_hat(1:2,j)=ud(1:2,j)-u(1:2,j);
%error -difference between actual u signal and desired u (ud)
```

```
    signal

%dynamic control law implementation—————
    if j>=2
        ud_dot(1:2,j)=(ud(1:2,j)-ud(1:2,j-1))/dt;%rate of change
            of desired u signal
    end
    YV1(1:2,j)=M__(1:2,1:2,j)*ud_dot(1:2,j);
    %divided YV signal into  YV1 and YV2 to observe influence of
        each part on YV
    YV2(1:2,j)=C__(1:2,1:2,j)*ud(1:2,j);
    YV(1:2,j)=M__(1:2,1:2,j)*ud_dot(1:2,j)+C__(1:2,1:2,j)*ud
       (1:2,j)+R__(1:2,j); %Y is regression matrix and V is
       vector of dynamical properties
    V=[m I us*m ul*m]';
% V=[m I us*m ul*m]';
% Xregg(1:(net-1),1:4)=repelem(V',net-1,1);
% Yregg(1:(net-1),1:2)= YV(1:2,1:net-1)';
% y=Yregg(1:(net-1),2);
% X = [ Xregg(1:(net-1),1) Xregg(1:(net-1),2) Xregg(1:(net-1),3)
    Xregg(1:(net-1),4) ];
% b = regress(y,X);
%regression matrix obtained by running above commented code,
    where YV there is obtained by ud from kinematic control law
    and some assumptions
      beta=[   -0.5745    1.8878
           0             0
           0             0
           0              0];
    Y(1:2,1:4,j)=beta';                             %Y is regression
        matrix
    magnt(1,j)=norm(Y(1:2,1:4,j)'*u_hat(1:2,j));%auxilary term
    taua1(1:2,j)=(Y(1:2,1:4,j)*(rho^2)*Y(1:2,1:4,j)'*u_hat(1:2,j
       ));
    %auxilary term taua sectioned to observe its influence
    taua(1:2,j)= (Y(1:2,1:4,j)*(rho^2)*Y(1:2,1:4,j)'*u_hat(1:2,j
       ))/(magnt(1,j)*rho+eps2);
    %auxilary term taua
    InvB__(1:2,1:2,j)=pinv(B__(1:2,1:2,j));
    Tqd1(1:2,j)=pinv(B__(1:2,1:2,j))*(auxerror(1,j)*J*z(1:2,j));
    %Tqd sectioned to observe its terms influence
    Tqd2(1:2,j)=pinv(B__(1:2,1:2,j))*z_hat(1:2,j);
    Tqd3(1:2,j)=pinv(B__(1:2,1:2,j))*YV(1:2,j);
    Tqd4(1:2,j)=pinv(B__(1:2,1:2,j))*taua(1:2,j);
    Tqd5(1:2,j)=pinv(B__(1:2,1:2,j))*k3*u_hat(1:2,j);

    Tqd(1:2,j)= pinv(B__(1:2,1:2,j))*(auxerror(1,j)*J*z(1:2,j)+
       z_hat(1:2,j)+YV(1:2,j)+taua(1:2,j)+k3*u_hat(1:2,j)); %
```

```
desired torque to be generated on the wheel

%motor level voltage control law—————————
    if j>=2
        Tqd_dot(1:2,j)=(Tqd(1:2,j)-Tqd(1:2,j-1))/dt;
    end
    T_hat(1:2,j)=Tqd(1:2,j)-Tq(1:2,j);
    %error - difference between desired torque and actual torque

    Ad(1,j)=(55.556*Tq(1,j)+2.457)/N;
    %desired current in each motor to produce appropriate torque Tq

    Ad(2,j)=(55.556*Tq(2,j)+2.457)/N;

    angvelwheelRL(1:2,j)=[(velB(1,j)+velB(2,j)*c)/r ; (velB(1,j)-velB(2,j)*c)/r ]; %angular velocities of wheels on left and right side of mobile robot
    Vd1(1:2,j)=(L*(ki*N)^-1*(Tqd_dot(1:2,j)+k4*T_hat(1:2,j)+B__(1:2,1:2,j)'*u_hat(1:2,j)));%Vd sectioned into Vd1, Vd2, Vd3 to observe its terms influence
    Vd2(1:2,j)=R*Ad(1:2,j);
    Vd3(1:2,j)=ke*N*angvelwheelRL(1:2,j);
    Vd(1:2,j)=(L*(ki*N)^-1*(Tqd_dot(1:2,j)+k4*T_hat(1:2,j)+B__(1:2,1:2,j)'*u_hat(1:2,j))+1*(R*Ad(1:2,j)+ke*N*angvelwheelRL(1:2,j)));
    %desired voltage signal to make the robot track or reach the desired goal path or point
    Tq(1:2,j+1)=(((Vd(1:2,j)-(angvelwheelRL(1:2,j)*ke*N))*ki)/R)*N; %actual torque generated by the motor on the wheel

    velB(1:2,j+1)=T(1:2,1:2,j)*ud(1:2,j)+PP(1:2,j); %obtaining actual velocity of the mobile robot in local frame
%obtating atual pose by integrating velocity using eulers backward numerical method
    q_dot(3,j+1)=velB(2,j+1);
    q(3,j+1)=q(3,j)+q_dot(3,j+1)*dt;
    Sq(1:3,1:2,j+1)=S(q(3,j+1));
    q_dot(1:3,j+1)=Sq(1:3,1:2,j+1)*velB(1:2,j+1);
    q(1:3,j+1)=q(1:3,j)+q_dot(1:3,j+1)*dt;
%range is the distance between actual point and goal
    net =j-1; %auxilary term to plot graphs of recorded data
    range=sqrt((qr(1,1)-q(1,j))^2+(qr(2,1)-q(2,j))^2) %#ok<NOPTS> %for constant ref point
    angularvel=velB(2,j) %#ok<NOPTS>
% prints parameter details o the simulation plot window
    plot(q(1,j),q(2,j),'.')
    plot(qr(1,1),qr(2,1),'go')
```

```
    xlabel('x');
    ylabel('y');
    ylim([-2 2])
    xlim([-2 2])
    str=sprintf('n:%d  j1:%d \n alpha0:%d alpha1:%d \n eps1:%d eps2:%d \n k1:%d k2:%d k3:%d k4:%d \n phi:%s', n,j1, alpha0,alpha1,eps1,eps2,k1,k2,k3,k4,phi);
    annotation('textbox',[0.5 0.6 0.5 0.4],'String',str,'FitBoxToText','on');
%condition to stop the simulation if the robot reaches proximity of goal
    drawnow()
    if range<0.1
        break
    end
end
f1=figure;
%plots actual position wrt  time
plot(t(1,1:net),q(1,1:net),'--b', t(1,1:net), q(2,1:net), '--g', t(1,1:net), q(3,1:net), '--r');
hold on
xlabel('time');
ylabel('q.- ');
lgd = legend;
lgd.NumColumns = 2;
legend('q1','q2','q3');
hold off

f2=figure;
%plots evolution of signals z and zd
plot3(z(1,1:net),z(2,1:net),t(1,1:net),'b');
hold on
plot3(zd(1,1:net),zd(2,1:net),t(1,1:net),'g');
xlabel('z(1),zd(1)');
ylabel('z(2),zd(2)');
zlabel('Time');
legend('z(1),z(2)','zd(1),zd(2)');
str = {['n = ' num2str(n) ' ' '$ \alpha 0 =\mbox{ }$' num2str(alpha0) ' ' '$ \alpha 1 =\mbox{ }$' num2str(alpha1)],['k1= ' num2str(k1) ' ' 'k2= ' num2str(k2) ' ' 'k3= ' num2str(k3) ' ' 'k4= ' num2str(k4) ], ['$ \epsilon 1 =\mbox{ }$' num2str(eps1) ' ' '$ \epsilon 2 =\mbox{ }$' num2str(eps2)] , ['$ \phi =\mbox{ }$' num2str(phi)] };
dim=[0.6 0.007 1 1];
annotation('textbox',dim,'string',str,'FitBoxToText','on','Interpreter','latex');

hold off
```

```matlab
f3=figure;
%plots evolution of the pose error-needs to converge close to zero
plot(t(1,1:net),q_hat(1,1:net), 'r', t(1,1:net), q_hat(2,1:net), 'g', t(1,1:net), q_hat(3,1:net), 'b');
xlabel('time');
ylabel('q_hat');
legend('q_hat(1)','q_hat(2)','q_hat(3)');
% ylim([-1 2])

f4=figure;
%plots initial and goal points and path traversed by the robot center
plot(q(1,1:net),q(2,1:net))
hold on
plot(qr(1,1),qr(2,1),'go')
xlabel('x');
ylabel('y');
ylim([-5 5])
xlim([-5 5])
legend('path','goal');

f5=figure;
%plots angular velosities of wheels on left and right sides of mobile robot wrt time
plot(t(1,1:net),angvelwheelRL(1,1:net), 'r', t(1,1:net), angvelwheelRL(2,1:net), 'g');
xlabel('time');
ylabel('angular velocity in rad/sec');
legend('angvelL','angvelR');

f6=figure;
%plots mobile robot velocity wrt time in local and inertial frame
plot(t(1,1:net),velB(1,1:net), 'r',t(1,1:net),q_dot(1,1:net),'g',t(1,1:net),q_dot(2,1:net),'b');
xlabel('time');
ylabel('mobile robot velocity  m/sec');
legend('local frame','Inertial frame x vel','Inertial frame y vel');

f7=figure;
%plots mobile robot angular velocity wrt time
 plot(t(1,1:net), velB(2,1:net), 'g');
 xlabel('time');
ylabel('mobile robot angular velocity  rad/sec');

f8=figure;
```

```
%plots angle of the mobile robot and rate of change of angle wrt time
plot(t(1,1:net),q(3,1:net), 'r',t(1,1:net), velB(2,1:net), 'g');
legend('direction in radians','angular velocity of body localframe in rad/sec');

f9=figure;
%plots graphs of desired torque and actual torqe wrt time
plot(t(1,1:net),Tqd(1,1:net), 'r', t(1,1:net), Tqd(2,1:net), 'g',t(1,1:net),Tq(1,1:net), '-.r', t(1,1:net), Tq(2,1:net), '-.g');
xlabel('time [s]');
ylabel('desired Torque dTR,dTL & actual Torque TR,TL');
legend('dTR','dTL','TR','TL');
str = {['n = ' num2str(n) ' ' '$ \alpha 0 =\mbox{ }$' num2str(alpha0) ' ' '$ \alpha 1 =\mbox{ }$' num2str(alpha1)],['k1= ' num2str(k1) ' ' 'k2= ' num2str(k2) ' ' 'k3= ' num2str(k3) ' ' 'k4= ' num2str(k4) ], ['$ \epsilon 1 =\mbox{ }$' num2str(eps1) ' ' '$ \epsilon 2 =\mbox{ }$' num2str(eps2)] , ['$ \phi =\mbox{ }$' num2str(phi)] };
dim=[0.6 0.007 1 1];
annotation('textbox',dim,'string',str,'FitBoxToText','on','Interpreter','latex');

f10=figure;
%plots current consumption by motors on left and rights side
plot(t(1,1:net),Ad(1,1:net), 'r', t(1,1:net), Ad(2,1:net));
xlabel('time [s]');
ylabel('current in  [A]');
legend('AR','AL');
str = {['n = ' num2str(n) ' ' '$ \alpha 0 =\mbox{ }$' num2str(alpha0) ' ' '$ \alpha 1 =\mbox{ }$' num2str(alpha1)],['k1= ' num2str(k1) ' ' 'k2= ' num2str(k2) ' ' 'k3= ' num2str(k3) ' ' 'k4= ' num2str(k4) ], ['$ \epsilon 1 =\mbox{ }$' num2str(eps1) ' ' '$ \epsilon 2 =\mbox{ }$' num2str(eps2)] , ['$ \phi =\mbox{ }$' num2str(phi)] };
dim=[0.6 0.007 1 1];
annotation('textbox',dim,'string',str,'FitBoxToText','on','Interpreter','latex');

f11=figure;
%plots voltage requirement by motors on left and rights side
plot(t(1,1:net),Vd(1,1:net), 'r', t(1,1:net), Vd(2,1:net));
xlabel('time [s]');
ylabel('Voltage in [v]');
legend('VR','VL');
str = {['n = ' num2str(n) ' ' '$ \alpha 0 =\mbox{ }$' num2str(alpha0) ' ' '$ \alpha 1 =\mbox{ }$' num2str(alpha1)],['k1= '
```

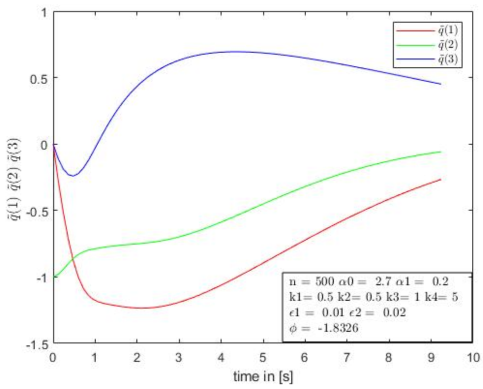
```
    num2str(k1) ' ' 'k2= ' num2str(k2) ' ' 'k3= ' num2str(k3) ' '
     'k4= ' num2str(k4) ], ['$ \epsilon 1 =\mbox{ }$' num2str(
    eps1) ' ' '$ \epsilon 2 =\mbox{ }$' num2str(eps2)] , ['$ \phi
     =\mbox{ }$' num2str(phi)] };
dim=[0.6 0.007 1 1];
annotation('textbox',dim,'string',str,'FitBoxToText','on','
    Interpreter','latex');

%function for kinematic transformation
function s=S(x)
j1=-0.1;
s=[cos(x) j1*sin(x);sin(x) -j1*cos(x);0 1];
end
```

Chapter 6

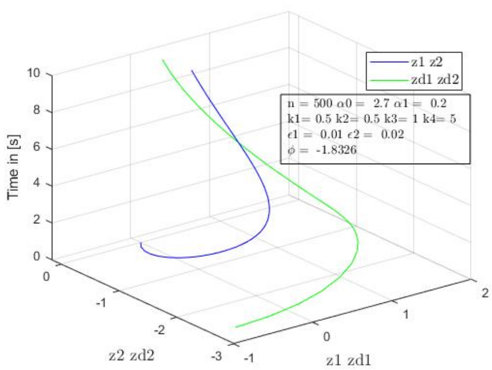
Results and Conclusion

6.1 Results

Figures in 6.1 shows the convergence of the pose error $\tilde{q}$ in inertial frame to the proximity of zero error in 6.1a and also the transformed pose error z converging into the desired signal generated by a similar tunable oscillator described in [13, equation 100] in 6.1b. This shows that the controller designed for the Field robot dynamic model with a backstepping control framework is stable and achieving the desired task of set-point regulation. The smoothness of the signals indicates the stable transient performance of the controller.



(a) Pose error evolution w.r.t Inertial frame and its convergence



(b) Transformed pose error evolution - actual z and desired z_d

Figure 6.1: Pose error evolution

Figures in 6.5 show transient performance of the various sub system signals. In fig 6.5b it can be seen that the controller is not exceeding the max saturation limit capable of the drive system which is $12\,(because\,of\,gearbox\,with\,speed\,reduction\,ratio\,12:1) \times 0.68(torque\,capable\,of\,the\,BLDC\,motor\,in\,Nm) = 8.16\,Nm$. Figures in 6.5c and 6.5d show the reasonable voltage and current requirements demanded by the control system when powered by a 22v battery accompanied by a PWM(pulse width modulation) module.

The max current demanded by the controller is close to 30A which is below the Max ESC(electronic speed controller) current capability which is 60A. Figures in 6.5e and 6.5f show the max linear velocity and angular velocities achieved by the mobile robot. Here the mobile robot speed limit of 2.5 m/s is obeyed as well.

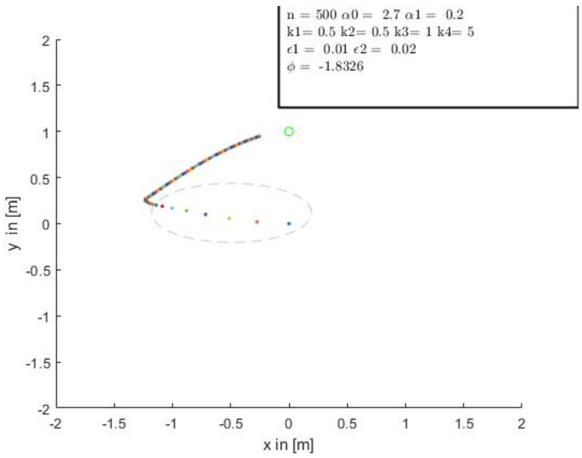


Figure 6.2: Velocity profile

In fig 6.2 the velocity profile of the mobile robot can be observed. The density of the points is the indicator. Farther points indicate higher speeds and closely packed points indicate lower speeds. This can be validated by comparing the profile to the fig local velocity of the mobile robot in 6.5e. Understanding of the controller stability criteria is very important to *tune the controller* for the better performance. Various parameters mentioned in the plots have bounded limits based on the control law design and its stability criteria. Stability topics introduced in section 4.5.2 form the basis for the understanding of the controller parameter bounds.

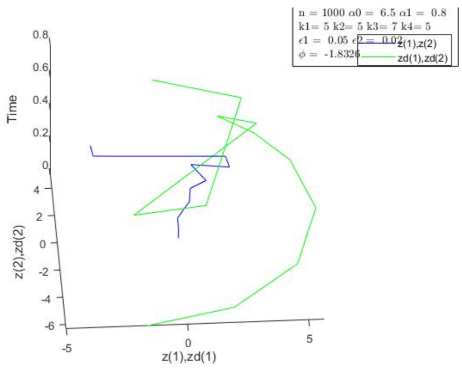


Figure 6.3: Transformed pose error evolution without proper tuning

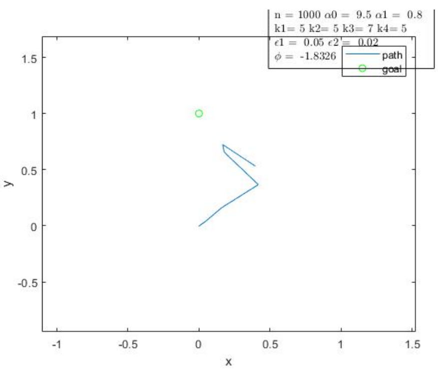


Figure 6.4: Mobile robot path for set point regulation without proper tuning

In figures 6.3, 6.4 it can be seen that without proper tuning of the controller the results obtained are not acceptable. A few other reasons for getting vague, unbounded results are improper modeling of the dynamics, unrealistic step size of the discrete control loop. During the simulation of the controller in MATLAB, multiple times unbounded results were obtained because of the above-mentioned reasons. Open-loop step and error response can be analyzed to validate the dynamic model.

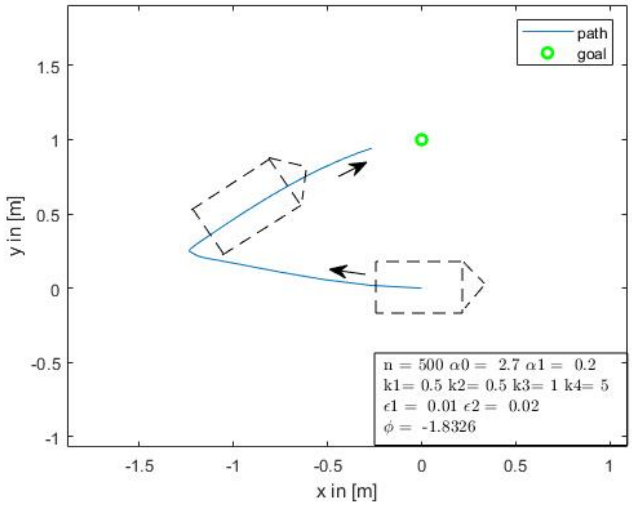
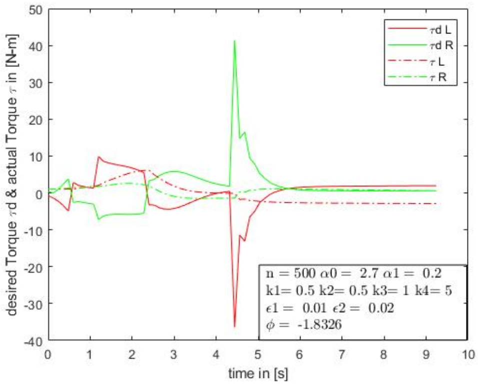


Figure 6.6: Mobile robot path for set point regulation

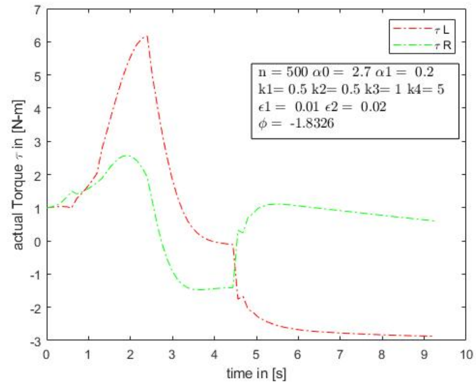
Finally the path traversed by the mobile robot is shown in 6.6 along with its set point regulation task of orienting itself to the desired final pose and orientation.

6.2 Conclusion

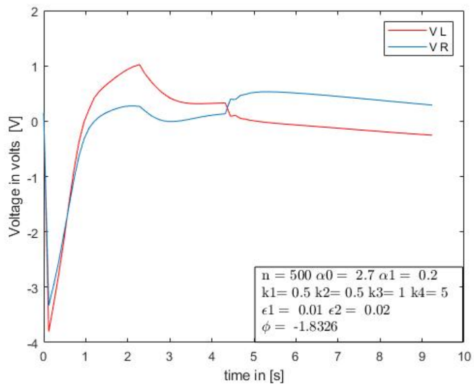
In this research project, the controller algorithm to achieve the task set point regulation based on [13] is modified to suit the self-built *Field robot prototype* and is successfully simulated in MATLAB. Sub-tasks like Coordinate Transformation, Non-holonomic Constraint optimization with Lagrange multiplier, Parameter estimation using Curve fitting, Regression analysis, control law design using Lyapunov stability criteria with Backstepping control framework, Numerical methods for embedded system implementation, Non-stationary differential equations and their solutions, non-linear dynamic system analysis and Error dynamics which make up the controller design process are briefed. The results obtained in MATLAB are presented and discussed. The MATLAB code is also valid for trajectory tracking with appropriate time-varying input, as it is based on unified regulation and tracking algorithm from [13]. The code cannot be implemented on an embedded system as more realistic wheel-ground interaction and Odometry need to be considered while designing the algorithm.



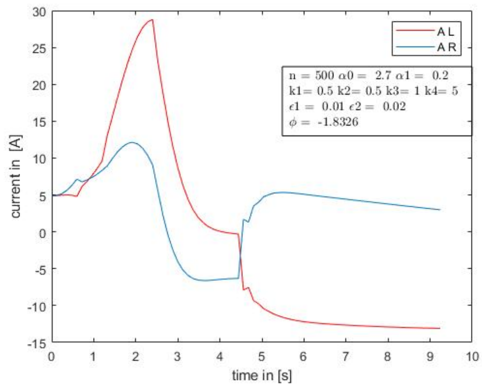
(a) τ_d and τ evolution w.r.t time



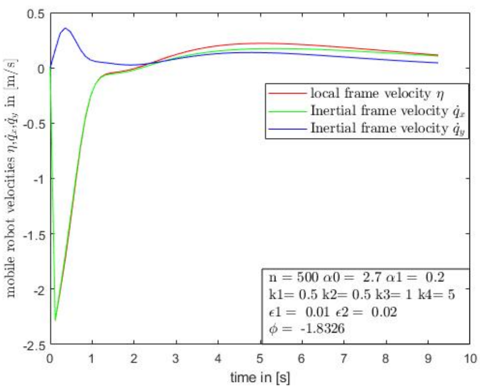
(b) actual torque τ signal



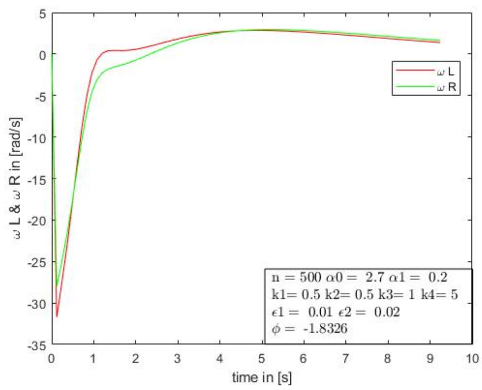
(c) Voltage signal on left and right side of the vehicle



(d) Current signal on left and right side of the vehicle



(e) Mobile robot velocity in local and Inertial frame



(f) Angular velocities of the wheels on left and right side of the mobile robot

Figure 6.5: Regulation case for $q_d = [0\ 1\ 0]$ and $q = [0\ 0\ 0]$

Bibliography

[1] Modern robotics, course 4: Robot motion planning and control.northwestern university. https://www.coursera.org/lecture/modernrobotics-course4/error-response-chapter-11-2-1-m6emr. Accessed: 2019-09-30.

[2] Nazareth Sarkis Bedrossian. *Nonlinear control using linearizing transformations*. PhD thesis, Massachusetts Institute of Technology, 1991.

[3] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[4] Roger W Brockett et al. Asymptotic stability and feedback stabilization. *Differential geometric control theory*, 27(1):181–191, 1983.

[5] Luca Caracciolo, Alessandro De Luca, and Stefano Iannitti. Trajectory tracking control of a four-wheel differentially driven mobile robot. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 4, pages 2632–2638. IEEE, 1999.

[6] Donald Christiansen, Charles K Alexander, and Ronald K Jurgen. *Standard handbook of electronic engineering*. McGraw Hill Professional, 2005.

[7] Peter Corke. *Robotics, vision and control: fundamental algorithms in MATLAB® second, completely revised*, volume 118. Springer, 2017.

[8] Alessandro De Luca, Giuseppe Oriolo, and Marilena Vendittelli. Control of wheeled mobile robots: An experimental overview. In *Ramsete*, pages 181–226. Springer, 2001.

[9] Warren E Dixon, Darren M Dawson, Erkan Zergeroglu, and Aman Behal. *Nonlinear control of wheeled mobile robots*, volume 175. Springer, 2001.

[10] Jay A Farrell and Marios M Polycarpou. *Adaptive approximation based control: unifying neural, fuzzy and traditional adaptive approximation approaches*, volume 48. John Wiley & Sons, 2006.

[11] Rolf Isermann and Marco Münchhof. *Identification of dynamic systems: an introduction with applications*. Springer Science & Business Media, 2010.

[12] Velimir Jurdjevic, Jurdjevic Velimir, and Velimir Ðurđević. *Geometric control theory*. Cambridge university press, 1997.

[13] Krzysztof Kozłowski and Dariusz Pazderski. Modeling and control of a 4-wheel skid-steering mobile robot. *International journal of applied mathematics and computer science*, 14:477–496, 2004.

[14] A Vyas Ojha and Achala Khandelwal. Control of nonlinear system using backstepping. *Journal of Research in Engineering and Technology*, 4(5):606–610, 2015.

[15] Open Stax and Paul Peter Urone. *College Physics*. OpenStax College, Rice University, 2012.

[16] US Tsubaki. The complete guide to chain, 1997.

[17] Spyros G Tzafestas. *Introduction to mobile robot control*. Elsevier, 2013.

[18] Heinz D Unbehauen. *CONTROL SYSTEMS, ROBOTICS AND AUTOMATION–Volume VI: Modeling and System Identification-III*. EOLSS Publications, 2009.

[19] Tianmiao Wang, Yao Wu, Jianhong Liang, Chenhao Han, Jiao Chen, and Qiteng Zhao. Analysis and experimental kinematics of a skid-steering wheeled robot based on a laser scanner sensor. *Sensors*, 15(5):9681–9702, 2015.

[20] LI Yuanping. *Slip Modelling, Estimation and Control of Omnidirectional Wheeled Mobile Robots with Powered Caster Wheels*. PhD thesis, 2009.