

Non-holonomic Differential Drive Mobile Robot Control & Design :
Critical Dynamics and Coupling Constraints

by

Iman Anvari

A Thesis Presented in Partial Fulfillment
of the Requirement for the Degree
Master of Science

Approved November 2013 by the
Graduate Supervisory Committee:

Armando A Rodriguez, Chair
Jenni Si
Konstantinos Tsakalis

ARIZONA STATE UNIVERSITY

December 2013

ABSTRACT

Mobile robots are used in a broad range of application areas; e.g. search and rescue, reconnaissance, exploration, etc. Given the increasing need for high performance mobile robots, the area has received attention by researchers. In this thesis, critical control and control-relevant design issues for differential drive mobile robots is addressed.

Two major themes that have been explored are the use of kinematic models for control design and the use of decentralized proportional plus integral (PI) control. While these topics have received much attention, there still remain critical questions which have not been rigorously addressed. In this thesis, answers to the following critical questions are provided:

When is

1. a kinematic model sufficient for control design?
2. coupled dynamics essential?
3. a decentralized PI inner loop velocity controller sufficient?
4. centralized multiple-input multiple-output (MIMO) control essential?

and how can one design the robot to relax the requirements implied in 1 and 2?

In this thesis, the following is shown:

1. The nonlinear kinematic model will suffice for control design when the inner velocity (dynamic) loop is much faster (10X) than the slower outer positioning loop.

2. A dynamic model is essential when the inner velocity (dynamic) loop is less than two times faster than the slower outer positioning loop.
3. A decentralized inner loop PI velocity controller will be sufficient for accomplishing high performance control when the required velocity bandwidth is small, relative to the peak dynamic coupling frequency. A rule-of-thumb which depends on the robot aspect ratio is given.
4. A centralized MIMO velocity controller is needed when the required bandwidth is large, relative to the peak dynamic coupling frequency. Here, the analysis in the thesis is sparse making the topic an area for future analytical work. Despite this, it is clearly shown that a centralized MIMO inner loop controller can offer increased performance vis-à-vis a decentralized PI controller.
5. Finally, it is shown how the dynamic coupling depends on the robot aspect ratio and how the coupling can be significantly reduced. As such, this can be used to ease the requirements imposed by 2 and 4 above.

To my loving parents, without whom none of my success would have been possible

ACKNOWLEDGMENTS

It is with immense gratitude that I acknowledge the support and help of my advisor, Professor Rodriguez, his guidance and persistent help motivated me through difficulties and made this thesis possible.

I would like to thank all of my friends for their unconditional moral support, and for being there for me when my family could not.

TABLE OF CONTENTS

	Page
LIST OF TABLES	vii
LIST OF FIGURES	viii
CHAPTER	
1 Introduction	1
1.1 A Brief History	1
1.2 Literature Sruvey	2
1.2.1 Main Problems	2
1.3 Objective	5
1.4 Thesis Organization	6
1.5 Summary and Conclusion	8
2 Mathematical Model	9
2.1 Non-Holonomic Constraint	9
2.2 Robot Kinematics	12
2.3 Robot Dynamics	14
2.4 Actuator Dynamics	18
2.5 Kinematics Vs. Dynamics	22
2.6 Robot + Actuator Dynamics	25
2.6.1 Plant Characteristics	27
2.6.2 Power	30
2.6.3 Mass	31
2.6.4 Plant Analysis	32
2.6.5 Robot Aspect Ratio	34
2.7 Conclusion	40
3 Dynamics Control Design	41

CHAPTER	Page
3.1 Decentralized Control	41
3.1.1 Proportional Controller	42
3.1.2 PI Controller	45
3.2 Inner Loop (Dynamics) Vs. Outer Loop (Kinematics)	47
3.2.1 Cartesian Stabilization	47
3.2.2 Kinematic Design Limitations.....	49
3.3 Decentralized Control Limitation	51
3.4 Centralized Control (Linear Quadratic Regulator)	55
3.5 Summary and Conclusion	57
4 Trajectory Planning	59
4.1 Planning	59
4.2 Trajectory:Path and Timing law	59
4.3 Effects of Kinematic Constraint	60
4.4 Differential Flatness.....	61
4.5 Conclusion	62
5 Summary and Future Work	63
REFERENCES	66
APPENDIX	
A MATLAB Codes	69

LIST OF TABLES

Table	Page
2.1 Dynamic Model Parameter Description and their Nominal Values	28

LIST OF FIGURES

Figure	Page
2.1 Pure rolling disk and its generalized coordinates in 2D plane	11
2.2 Mecanum wheel can move sideways and is holonomic	12
2.3 Generalized coordinates for a mobile robot	13
2.4 Linear and Angular velocity of the robot	15
2.5 Circuit equivalent of a DC motor with a free body attached	19
2.6 Torque applied to a free body	20
2.7 DC Motor block diagram	21
2.8 Block diagram of a mobile robot's kinematic	22
2.9 Block diagram of a mobile robot including actuator and body dynamics	23
2.10 Actuator and body dynamics block diagram from $\omega_{R_{ref}}$ & $\omega_{L_{ref}}$ to ω_R & ω_L	24
2.11 Singular Value plot of Mobile Robot Dynamics	28
2.12 Bode Magnitude Plot of Mobile Robot Dynamics	29
2.13 Variation of Power Vs. Km	31
2.14 Magnitude of Minimum Singular Value for Variations of K_m	32
2.15 Open Loop Bandwidth Vs. K_m	33
2.16 Magnitude of Diagonal and Off-Diagonal elements for Variations of K_m	34
2.17 Magnitude of Off-Diagonal to Diagonal ratio	35
2.18 Cuboid Shape Mobile Robot	36
2.19 Peak coupling ratio behavior Vs. robot's aspect ratio	37
2.20 Magnitude of Off-Diagonal to Diagonal ratio for Variations of K_m	37
2.21 Magnitude of Minimum Singular Value for Variations of Mass	38
2.22 Open Loop Bandwidth Vs. <i>mass</i>	39
2.23 Magnitude of Diagonal and Off-Diagonal elements for Variations of Mass	39

Figure	Page
3.1 Decentralized Controller Architecture for Speed Control	42
3.2 Magnitude of Diagonal and off-Diagonal elements for variations of K ...	43
3.3 Minimum singular value for variations of K	43
3.4 Bandwidth of the system Vs. Proportional gain (K)	44
3.5 Decentralized Controller Architecture for Speed Control	44
3.6 Magnitude of Diagonal and off-Diagonal elements for variations of (a) variations of K_p and (b) variations of K_i	46
3.7 (a) Bandwidth Vs. K_p (b) Bandwidth Vs. K_i	46
3.8 (a) Bandwidth Vs. K_p (b) Bandwidth Vs. K_i	47
3.9 Displacement Control Block Diagram from S_{ref} and θ_{ref} to s and θ	48
3.10 Mobile Robot in Cartesian Stabilization mode	48
3.11 Positioning System (Displacement Control) Block Diagram	49
3.12 Error between ideal (Kinematic) and actual (Kinematic + Dynamics) system Vs. BW ratio	50
3.13 (a) $max S_{12} $ Vs. BW (b) $max S_{11} $ Vs. BW	51
3.14 $ \frac{S_{12}}{S_{11}} $	52
3.15 Peak $ \frac{S_{12}}{S_{11}} $ within BW Vs BW	53
3.16 p_s Vs. Aspect Ratio	54
3.17 Rule of thumb Vs. Aspect Ratio	54
3.18 Dynamics Plant with a Linear Quadratic Regulator	56
3.19 Closed loop coupling ratio with decentralized control	57
3.20 Closed loop coupling ratio with centralized control	58

Chapter 1

INTRODUCTION

1.1 A Brief History

Contrary to popular belief, *Robots* are relatively old devices, with *Leonardo's mechanical knight* dating back to 1495 being the first robot recorded in history [1]. First major wave of robots started in late 60's at industrial environments, where manual labor was gradually being replaced by automated robots in the production lines [2] [3].

The presence of robots in industry have been fortified for many years now; however, there still remains a huge gap in the market for other types mostly due to technology limitations and high prices. Recent developments have significantly increased computing capabilities of processors while lowering the costs. This allows cheap, precise and powerful robots to become a reality in the upcoming years, where they will only be limited by human imagination.

In 1948 W.Grey Walter designed the first *Mobile Robot* called *Machina Specultrix*. This robot was equipped with a light sensor to explore the environment. Because of the simple design this machine was extremely unreliable and in need of constant attention [4].

Johns Hopkins University developed the *Beast* in 1960 utilizing sonar to wander around the halls until its batteries ran low [5].

In 1969 *Mowbot* was introduced to market where as the first attempt in automatic lawn mowing [6]. In early 90's *Joseph Engelberger*, father of industrial robotic arm, designed the first commercially available autonomous mobile hospital robot [7]. Later in 1997 NASA sent the *Mars Pathfinder* with its rover *Sojourner* to Mars. Equipped with a hazard avoidance system, Sojourner was able to autonomously find its way through unknown martian terrain.

Over the past decade the development of mobile robots has faced a new era with ever increasing processing power of computers along with accurate sensors. In the past two decades mobile robots, along with their capabilities and their design aspects have been a very popular topic between scientist from various fields such as controls, robotics, computer science, etc.

1.2 Literature Sruvey

In this section relevant research will be explored in order to put a foundation for our work and justify the objective of this document. Although research in this area has been going on for many years, the most recent articles will be more emphasized.

1.2.1 *Main Problems*

There are some major problems concerning *Mobile Robots* which robotic and control community try to answer. A Mobile Robot, as the name suggests, has to move from an initial point and reach a final destination, while satisfying speed and/or position constraints on its way.

This task has been broken down into different problems and addressed separately or together. These problems are classified as:

1. Path Tracking (Trajectory Tracking)
2. Point to Point (Cartesian) Stabilization
3. Posture Regulation (Parking Problem)
4. Velocity Control

Path Tracking is the highest level problem which consists of a robot following a predefined path and reaching a destination. A more general form of path tracking is the *Trajectory Tracking* problem which is proposed by defining a timing law on the desired path; implicitly putting a velocity constraint on the robot at each sample point.

One of the most common solutions for this class of problems is through Liapunov-Like stabilization [8] [9] [10]. In this method a linear or non-linear controller is proposed and the stability of the closed loop system is proved through Liapunov function [11] [12] [13]. In this approach a non-linear geometric model of mobile robot (Kinematics) is incorporated for control design and closed loop stability analysis. [14], [15] and [16] are some examples of using model predictive controller for trajectory tracking of nonholonomic systems.

Point to Point stabilization in nature is a simpler problem, where the robot only has to start from an initial point and reach a destination point. In this class of problems the behavior of the robot between the initial and final point, and also the final orientation of the robot is not explicitly controlled. Point to Point stabilization can be

addressed as a subclass of Path Tracking or *Posture Regulation* problems, depending on the the goal being to follow a path or just reaching a reference point.

Posture regulation is a general form of *Point to Point stabilization*. The objective of the robot in this problem is to start from an initial posture and end up at a final posture. Due to the non-holonomic nature of the system and it limitations, this class of problems has been recognized as the hardest issues in mobile robotic society.

Liapunov stabilization is the oldest method to solve this problem at kinematic level [17] [13] [18] [19]. However, recent studies have managed to simplify this problem by transforming the inputs from posture to displacement and orientation and use linear controllers to address the problem [20] [21]. This approach not only simplifies the controller structure, but also allows a more performance based control system design as well.

Other than [20] and [21], in which the dynamics are included but not explicitly controlled, all of the previous problems have been addressed in a *Kinematic* level. This means that the actuator and robot dynamics are neglected and it is assumed that velocity commands are realized instantaneously. This negligence is justified provided that the motor is powerful enough or it is already being controlled using lower level controllers [18] [22] [19] [11] [23]. This brings out the importance of *Velocity Control*.

Velocity Control of the mobile robot is a very fundamental problem. This is because underneath any technique addressing the problems mentioned earlier, there is a need for seamless velocity tracking.

In order to achieve this goal different approaches have been proposed. One method is to cancel the dynamics of the system using state feedback based on the exact knowledge of such dynamics [13], [24], [25]. This method is highly sensitive to the parameter error and is not considered a very practical approach.

Recent studies have put more focus on the dynamic model and its effects on the system as a whole. Both the robot and a simplified actuator dynamics have been considered in [20] and [21]. As it was mentioned earlier, two PID controllers are incorporated to solve both path and trajectory problems. In this method the velocity is not sensed or explicitly controlled. Solely depending on position sensing, which is in general more prone to errors compared to velocity sensing, can make the system more susceptible to errors.

In [26] a detailed model of mobile robot including the dynamics and torque coupling has been proposed, the dynamic are then controlled using a Model Reference Adaptive controller at torque level. Although this is a genuine effort in considering the dynamics, in most systems commanding torques is not a viable option.

1.3 Objective

From literature survey one can observe while there are many control approaches for each of the proposed problems, there are gaps in the dynamic modeling aspects of mobile robots. While all of the surveyed works address the proposed problems, they are heavily based on assumptions of neglecting the dynamics, which from a control system design point of view may be unjust.

This document explores two major themes : the use of nonlinear kinematic models for control design and the use of decentralized proportional plus integral (PI) control. While these topics have received much attention, there still remain critical questions which have not been rigorously addressed. In this document answers to the following fundamental questions are provided:

1. When is the *Kinematic Model* sufficient ?
2. When is the *Dynamic Model* essential?
3. When is a *Decentralized Control* scheme sufficient?
4. When is a *Centralized Control (MIMO)* essential?

The answers to the proposed questions are intended to be used for development of a *Mobile Robotic System (MRS)* as a part of *Flexible Autonomous Machines operating in an uncertain Environment (FAME)* project at Arizona State University.

1.4 Thesis Organization

The remainder of the thesis is organized as follows:

Chapter 2 provides explanations on the mathematical model of a differential drive mobile robot. In this chapter dynamic and kinematic model are explained along with non-holonomic constraints of the robot. Additionally, their differences and limitations are thoroughly explored in this chapter. The detailed dynamic model of the Mobile robot with torque coupling is then introduced. Performance metrics such as *Coupling Ratio* and *Bandwidth* effects of Power and Mass on such system are then analyzed. Finally the dependency of dynamic coupling on the aspect ratio of the robot is discussed in details. Coupling analysis shows that for a cuboid shape robot with

aspect ratio of $\sqrt{5}$ the coupling goes to zero, allowing for simpler control structures to be used. At the end by summarizing our analysis we answer how can one design a system to facilitate a kinematic design, helping with fundamental question 1 and 2.

In Chapter 3, in order to answer the first two previously mentioned fundamental questions, effects of inner loop system (Dynamics Velocity Loop) on the outer loop system (Kinematic Position Loop) is compared and a rule of thumb is derived. It's concluded that if the Inner loop dynamics is much faster (ten times faster) than the outer loop kinematics, the error will be small enough, allowing for a kinematic design. On the other hand if the inner loop dynamics are not fast enough (less than two time faster than the outer loop) then the error will be large, thus the need for dynamic model consideration.

Different control schemes for the dynamic model are then analyzed. Decentralized P and PI controller are designed for such systems and different performance aspects of such scheme is explored. The limitations of using a decentralized control is then addressed and a rule of thumb for the third fundamental question is derived. It is stated that operating in low frequencies, relative to the peak coupling frequency (ω_c), would yield high performance closed loop characteristics. The driven rule of thumb for the third question is dependent on the aspect ratio of the robot and can become less strict as we reach the zero coupling aspect ratio of $\sqrt{5}$.

Finally, it's shown that if high velocity bandwidth, relative to the peak dynamic coupling frequency, is desired A Centralized LQR controller is required. Further analysis clearly states that the centralized control is able to overcome limitations of the decentralized scheme, thus allowing us to answer the forth fundamental question.

Here, the analysis in the thesis is sparse making the topic an area for future analytical work

Chapter 4 discusses the outer loop path generation problem of the mobile robot, focusing on generating viable speed commands for a desired path, which can be applied to the controlled dynamics discussed in previous chapters.

Chapter 5 summarizes the results in this thesis and proposes the possibility of future works that hasn't been addressed in this document.

1.5 Summary and Conclusion

In section 1.1 a brief history of mobile robots was given. Section 1.2 thoroughly discussed the research that has been done on mobile robots, addressing main problems of the field. In section 1.3 the main objective of this thesis, and the reasoning behind it was proposed. Finally section 1.4 showed how the rest of this thesis is organized and what is discussed in each chapter.

Chapter 2

MATHEMATICAL MODEL

Deriving a precise mathematical model is a crucial part of designing control system for any physical plants such as mobile robots. In this chapter dynamics and kinematics of a differential drive robot are derived and differences between the two models and limitations of the kinematic model are explored.

The pure rolling nature of the wheels causes a reduction in the local mobility of the robot. This limitation is expressed as a *non-holonomic constraint* which is further discussed. In later chapters the importance of the *non-holonomic constraint* in trajectory planning is thoroughly discussed.

2.1 Non-Holonomic Constraint

Wheeled vehicles are generally subjected to a constraint. For instance, a car can reach any final configuration in its plane, but it can never move sideways. Hence, depending on the goal configuration, it requires to perform a series of maneuvers (such as parallel parking) to reach the desired state.

First, holonomic and non-holonomic systems have to be defined. Let's consider a mechanical system with *generalized coordinates* $q \in C$, where C is the configuration space of the proposed system and coincides with \mathbb{R}^n . For such system, a constraint is called *Kinematic* when it only involves generalized coordinates (q) and velocities (\dot{q}).

Kinematic Constraints are usually defined in *Pfaffian Form*

$$v_i^T(q)\dot{q} = 0 \quad i = 1, \dots, k < n \quad (2.1)$$

where v_i 's are k linearly independent vectors.

If all of the kinematic constraints defined by Equation 2.10 are *integrable* to a form of

$$h_i(q) = m_i \quad i = 1, \dots, k < n$$

where, m_i is the integration constant, then they are considered to be *holonomic* constraints and the system subjected to them is called a *holonomic system*. Joints in a robotic manipulator are common example of such constraints.

Each holonomic constraint causes a loss of accessibility of the system in its configuration space. Hence, for a system with k holonomic constraints, the accessible configurations are reduced to a $n - k$ dimensional subset of C .

A *non-holonomic system* on the other hand, is subjected to at least one non-integrable (i.e. *non-holonomic*) constraint. Although such constraint limits the local mobility of the system, due to its non-integrable nature, the accessibility to C is not affected. Hence, generalized coordinates are not reduced. However, generalized velocities in a system subjected to k non-holonomic constraint belongs to a $(n - k)$ dimensional subspace.

Wheels are typical sources of non-holonomic constraints. Consider the disk in Figure 2.1 with generalized coordinates $q = [x \ y \ \theta]^T$, assuming the disk can only roll on the touching plane without slipping to the sides (i.e. there is no velocity

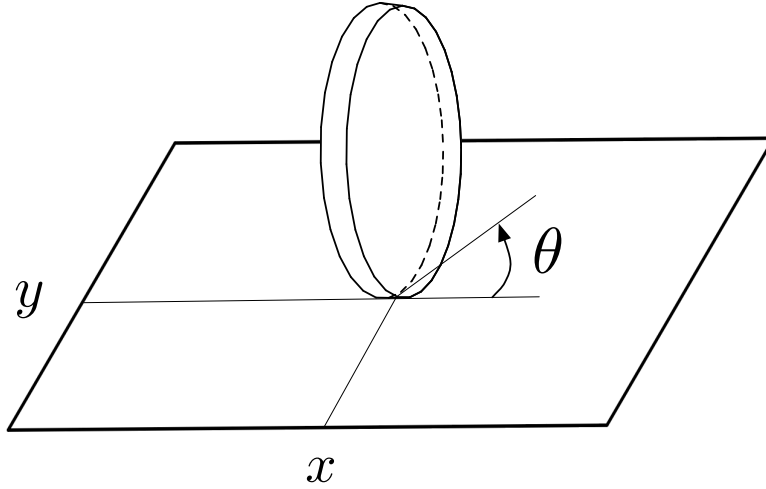


Figure 2.1: Pure rolling disk and its generalized coordinates in 2D plane

component for the contact point perpendicular to the plane containing the disk).

This can be defined as:

$$\dot{x} \sin \theta - \dot{y} \cos \theta = 0 \quad (2.2)$$

Rewriting Equation 2.2 in *pfaffian form* will result in

$$[\sin \theta \quad -\cos \theta \quad 0] \dot{q} = 0 \quad (2.3)$$

As it can be seen, Equation 2.3 is not integrable causing the nature of the wheel to be non-holonomic. Also, it should be emphasized that this constraint implies no loss in accessibility of the wheel configuration space, meaning that wheel can reach any goal configuration $q_f = [x_f \quad y_f \quad \theta_f]^T$ starting from any initial state $q_i = [x_i \quad y_i \quad \theta_i]^T$.



Figure 2.2: Mecanum wheel can move sideways and is holonomic

This kinematic constraint applies to all wheel-based systems, making them non-holonomic. However, it should be noted that not all wheels are non-holonomic. Configuration of caster wheel proposed in mic or *Mecanum wheels* (as shown in Figure 2.2), which are commonly used in omnidirectional robots, are exempt from this constraint and in fact are considered, holonomic.

2.2 Robot Kinematics

Reordering k kinematic constraints in Equation 2.10 into matrix form $V^T(q)\dot{q} = 0$, shows that the generalized velocities (\dot{q}) belongs to null space of $V^T(q)$, which is $(n-k)$ dimensional and agrees with what was stated earlier in this chapter.

Choosing a basis for $\mathcal{N}(V^T(q))$ denoted by $[b_1(q)\dots b_{n-k}(q)]$ a *kinematic model* of the constrained mechanical system is given by:

$$\dot{q} = \sum_{i=1}^{n-k} b_i(q)u_i = B(q)\mathbf{u} \quad (2.4)$$

where $\mathbf{u} = [u_1 \dots u_{n-k}]^T \in \mathbb{R}^{n-k}$ is the input vector and $q \in \mathbb{R}^n$ is the state vector.

The basis for nullspace of $V^T(q)$ is not unique and typically, it can be chosen such that inputs u_i represent a physical concept. However, these inputs should not directly represent forces or torques, hence the name *kinematic model*.

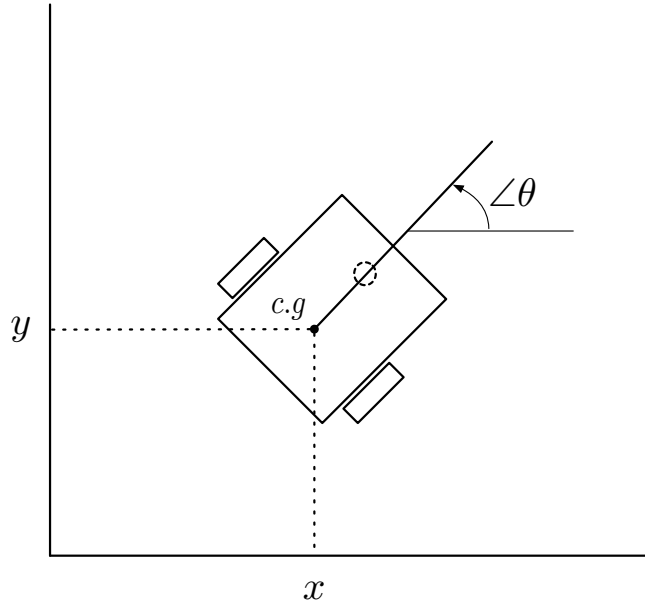


Figure 2.3: Generalized coordinates for a mobile robot

Consider the mobile robot in Figure 2.3. Using generalized coordinate vector $q = [x \ y \ \theta]$ the robot's posture can be defined on its whole configuration space. The wheels driving the robot make it non-holonomic and imposes the pure rolling constraint on the system which as discussed before, is expressed as

$$V^T(q)\dot{q} = [\sin \theta \quad -\cos \theta \quad 0]\dot{q} = 0 \quad (2.5)$$

a basis for $\mathcal{N}(V^T(q))$ is then chosen as

$$B(q) = [b_1(q) \quad b_2(q)] = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \quad (2.6)$$

Using this basis and based on Equation 2.4 the kinematic model will be

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta \\ \sin \theta \\ 0 \end{bmatrix} v + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega \quad (2.7)$$

where, the inputs have clear physical interpretation, v and ω are the linear velocity and angular velocity of the robot, respectively, as shown in Figure 2.3.

There exists a one to one relation between formerly mentioned velocities and actual velocity inputs, which are angular speed of two wheels denoted by ω_L and ω_R for left and right wheels, respectively and is governed by:

$$v = \frac{r(\omega_R + \omega_L)}{2} \quad \omega = \frac{r(\omega_R - \omega_L)}{l} \quad (2.8)$$

where, r is the radius of the wheels and l is the distance between the wheels as shown in Figure 2.4.

2.3 Robot Dynamics

Inputs in a kinematic model do not directly represent actual inputs (i.e. forces and/or torques). In another words, we are neglecting *dynamics* of a system when dealing just with a kinematic model. Consequently, It is important to derive the

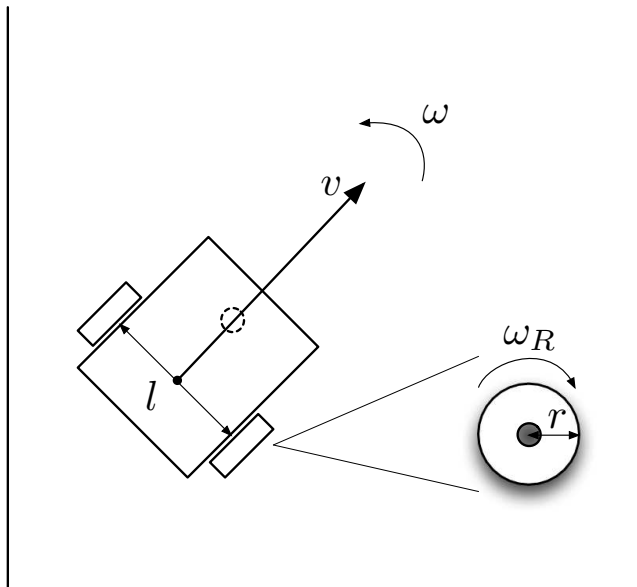


Figure 2.4: Linear and Angular velocity of the robot

dynamic model and explore its characteristics.

There are two methods for dynamic model derivation. *Newton-Euler* method describes the system in terms of all the forces and momentum acting on the system based of direct interpretations of Newtons Second Law of Motion.

On the other hand, *Lagrange* method incorporates the concepts of *Work and Energy* to indirectly derive the equations of motion. Here, Lagrange method is chosen due to its more systematic nature and automatic elimination of workless and constraint forces.

Lagrangian of a system is defined as the difference between its kinetic and potential energy

$$\mathcal{L}(q, \dot{q}) = \mathcal{T}(q, \dot{q}) - \mathcal{U}(q) = \frac{1}{2} \dot{q}^T I(q) \dot{q} - \mathcal{U}(q) \quad (2.9)$$

where, $\mathcal{T}(q, \dot{q})$ and $\mathcal{U}(q)$ are the kinetic and potential energy, respectively and $I(q)$ is the inertia matrix of the mechanical system.

Lagrange-Euler equations representing the dynamics are expressed as

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{q}} \right)^T - \left(\frac{\partial \mathcal{L}}{\partial q} \right)^T = 0 \quad (2.10)$$

This general form of Lagrange equation applies to holonomic system. In case of a non-holonomic system we have to replace Equation 2.10 by

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{q}} \right)^T - \left(\frac{\partial \mathcal{L}}{\partial q} \right)^T = S(q)\tau + V(q)\lambda \quad (2.11)$$

where, $S(q)$ is a $(n \text{ by } m)$ matrix mapping the $(m = n - k)$ external inputs τ to generalized forces, $V(q)$ is the transpose of $V^T(q)$ in Equation 2.5 governing the non-holonomic constraint. $\lambda \in \mathbb{R}^m$ is the vector of the Lagrange multipliers representing the forces required to impose such constraint in the configuration plane. $V(q)\lambda$ is the reaction forces at generalized coordinate plane.

Based on Equation 2.9 and Equation 2.10, the dynamical model of a non-holonomic mechanical system is obtained as

$$I(q)\ddot{q} + n(q, \dot{q}) = S(q)\tau + V(q)\lambda \quad (2.12)$$

$$V^T(q)\dot{q} = 0 \quad (2.13)$$

$$n(q, \dot{q}) = \dot{I}(q)\dot{q} - \frac{1}{2} \left(\frac{\partial}{\partial q} (\dot{q}^T I(q) \dot{q}) \right)^T + \left(\frac{\partial \mathcal{U}(q)}{\partial q} \right)^T \quad (2.14)$$

where $n(q, \dot{q})$ given in Eq 2.14 represents vector of centripetal and coriolis terms [26] [27].

Let I be the moment of inertia around the central vertical axis and m the mass of the differential drive mobile robot in 2.3. Using the Lagrange representation in Equation 2.12 and Equation 2.13, the dynamic model of the robot is then derived.

$$\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix} \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \tau_l \\ \tau_a \end{bmatrix} + \begin{bmatrix} \sin \theta \\ -\cos \theta \\ 0 \end{bmatrix} \lambda \quad (2.15)$$

$$\begin{bmatrix} \sin \theta & -\cos \theta & 0 \end{bmatrix} \dot{q} = 0 \quad (2.16)$$

Where, τ_l and τ_a represent the linear force and angular torque of the mobile robot, respectively. The robot is in inertial frame coriolis and centripetal term $n(q, \dot{q})$ is non-existence [26].

The relations between linear velocity (v), angular velocity (ω) and the generalized velocities (\dot{q}) are

$$v = \sqrt{\dot{x}^2 + \dot{y}^2} \quad (2.17)$$

$$\omega = \dot{\theta} \quad (2.18)$$

Using derivatives of Equation 2.17 and Equation 2.18, the dynamic model represented in matrix form in Equation 2.15 can be rewritten in a more familiar form.

$$\dot{x} = v \cos \theta \quad (2.19)$$

$$\dot{y} = v \sin \theta \quad (2.20)$$

$$\dot{\theta} = \omega \quad (2.21)$$

$$\dot{v} = \frac{\tau_l}{m} \quad (2.22)$$

$$\dot{\omega} = \frac{\tau_a}{I} \quad (2.23)$$

Where, Equations 2.19 through 2.21 are the kinematic models and Equations 2.21 & 2.22 integrate the dynamics of the robot.

It should be noted that the constraint equation (Equation 2.16) is valid in any case. Similar to linear and angular velocity of the robot and wheels' angular velocity, angular torque τ_a and linear torque τ_l are related to the torques of each wheel by Equation 2.24:

$$\tau_l = \frac{\tau_R + \tau_L}{r} \quad \tau_a = \frac{l(\tau_R - \tau_L)}{r} \quad (2.24)$$

where, τ_R and τ_L respectively represent right and left wheel torques.

Such toques and velocities are produced by the actuators driving each wheel. It is important to appreciate the fact that these actuators have their own internal dynamics and can not realize speed commands instantaneously.

2.4 Actuator Dynamics

DC motors are widely used in robotic applications and are the main type of actuators used in mobile robots. Consequently, it is important to analyze and integrate their dynamics into robot's model. There are two classes of DC motors: *Field-Current*

Controlled and Armature-Current Controlled. In a Field-Current Controlled motor, the armature current i_a is kept constant while the field-current is controlled using field voltage V_f commands.

On the other hand, in a Armature-Current Controlled motor, the armature voltage V_a is the command to control the armature current while keeping the field-current i_f constant. Armature-current controlled DC motors are more common choice in mobile robots and are the basis of further discussions in this text. For a more detailed discussion on DC motor modeling refer to [28], [29] and [30].

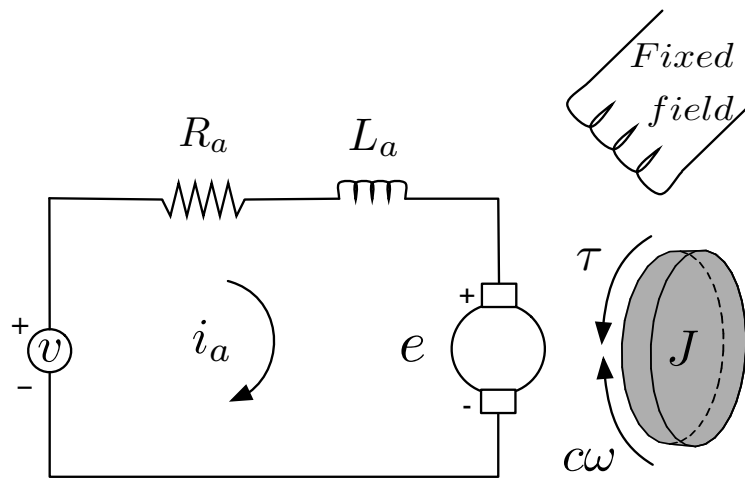


Figure 2.5: Circuit equivalent of a DC motor with a free body attached

In an Armature-Current Controlled structure, the motor torque is linearly dependent on the armature current by

$$\frac{\tau_m(s)}{I_a(s)} = K_m \quad (2.25)$$

where, $\tau_m(s)$ is the motor torque in S-domain and K_m is called the motor torque constant.

Based on circuit model provided in Figure 2.5, and considering the *back EMF* voltage (v_b), induced by the rotation of armature winding, the voltage relation on the armature will be

$$v_a = v_r + v_L + v_b \quad (2.26)$$

Back EMF has a linear relation to angular speed through back EMF constant K_b , taking Laplace transform of Equation 2.26 the following equation is achieved.

$$V_a(s) - V_b(s) = V_a(s) - K_b \omega(s) = (R_a + L_a s)I_a(s) \quad (2.27)$$

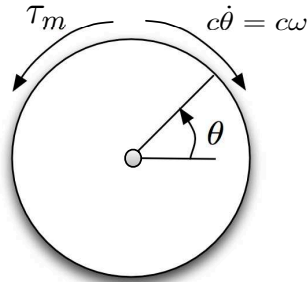


Figure 2.6: Torque applied to a free body

For the free body connected to the motor(Figure 2.6) rotational motion is formulated by

$$J\dot{\omega} + c\omega - \tau_m \quad (2.28)$$

where, ω is the angular velocity, c is motor friction constant and J is the moment of inertia of the rotor.

Taking Laplace transform the transfer function from the input motor torque to angular velocity is obtained

$$\frac{\omega(s)}{T_m(s)} = \frac{1}{J.s + c} \quad (2.29)$$

Using Equations 2.25,2.27 and 2.29 transfer function from armature voltage to angular velocity is

$$\frac{\omega(s)}{V_a(s)} = \frac{K_m}{(L_a \cdot s + R_a)(J.s + c) + K_b K_m} \quad (2.30)$$

Closed loop block diagram of DC motor model expressed in Equation 2.30 is shown in Figure 2.7, angular displacement can also be found by integrating $\omega(s)$.

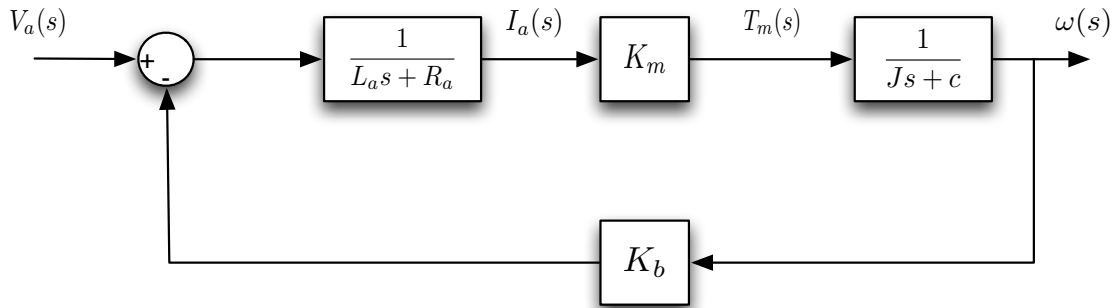


Figure 2.7: DC Motor block diagram

2.5 Kinematics Vs. Dynamics

In previous sections kinematics and dynamics of a differential drive mobile robot was systematically derived. In robotic society it is very common to use the kinematic model as the plant for control design [13] [18] [31] [12] [19]. This is justified by assuming that the motor is *powerful enough* to make the dynamic effects negligible.

This section is intended to have a deeper look into this matter by comparing the kinematic and dynamic model and exploring the limitations of the kinematic model.

Kinematic model (Equation 2.7) considers v and ω as the main inputs of the plant, which means that the linear and angular velocity of the system is realized instantaneously. But, how accurate is this assumption? Block diagram of a kinematic model is shown in Figure 2.8.

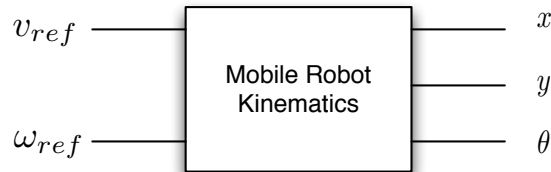


Figure 2.8: Block diagram of a mobile robot's kinematic

On the other hand complete system's block diagram so more similar to Figure 2.9, where τ_R and τ_L represent the effective torque applied to right and left wheel, respectively. Also, $\omega_{R_{ref}}$ and $\omega_{L_{ref}}$ are respectively right and left angular velocity commands calculated through:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = M \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} \quad (2.31)$$

where M is a transformation matrix defined as:

$$M = \begin{bmatrix} \frac{r}{2}, & \frac{r}{2} \\ \frac{r}{l}, & \frac{-r}{l} \end{bmatrix} \quad (2.32)$$

r and l are the radius of the wheels and the distance between them respectively.

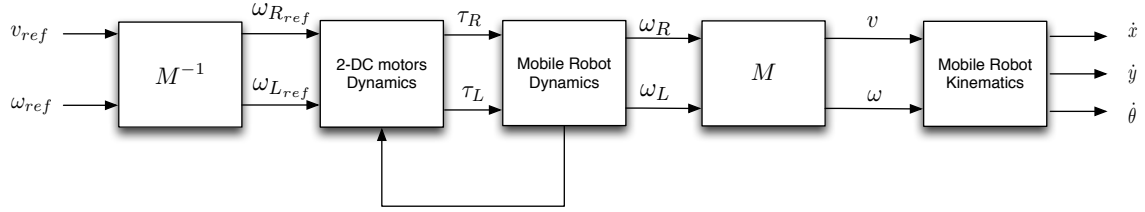


Figure 2.9: Block diagram of a mobile robot including actuator and body dynamics

In order to inspect the effects of actuator and mobile dynamics, the DC Motor model derived in section 2.4 along with derived dynamics in Equation 2.22 and Equation 2.23, are used to derive a precise model of the *actuator + mobile robot dynamics*. This model is illustrated in Figure 2.10. In this model, DC motors are considered to be identical.

Following previous discussions, an ideal system would have a transfer function matrix as follows.

$$T_{\omega\omega_{ref}} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (2.33)$$

this indicates perfect command following and absolutely no *coupling* in actuator +

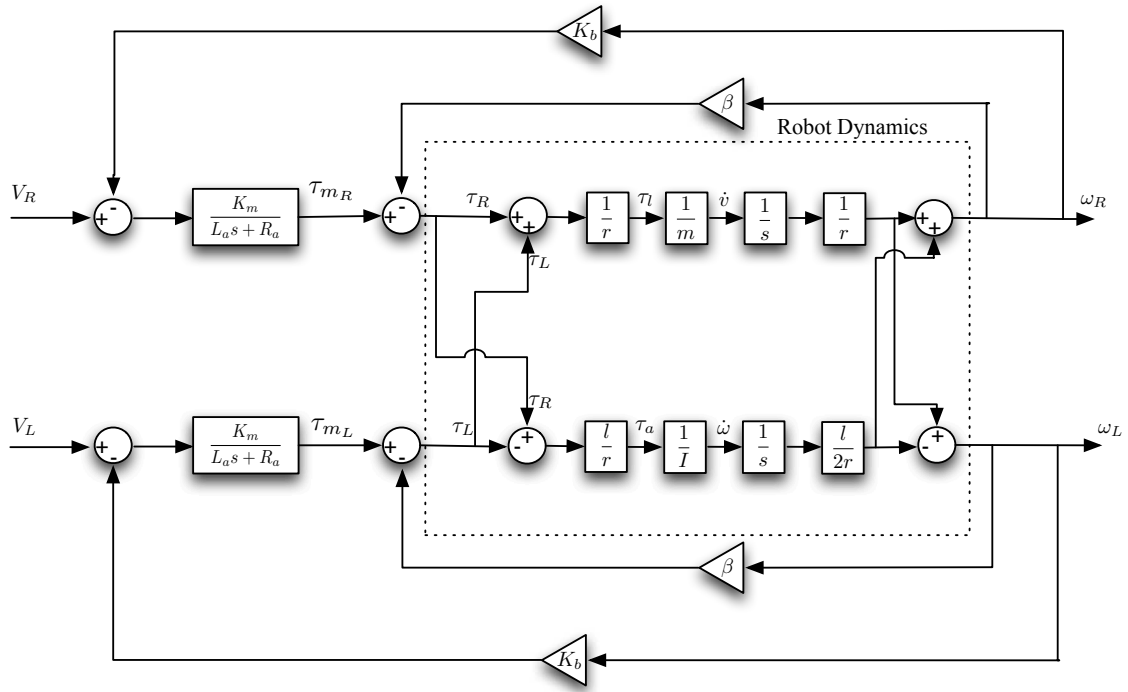


Figure 2.10: Actuator and body dynamics block diagram from $\omega_{R_{ref}}$ & $\omega_{L_{ref}}$ to ω_R & ω_L

robot dynamics.

On the other hand from the proposed block diagram (Figure 2.10), one can clearly see that not only there exists a *torque coupling* between left and right channels, but also it is highly unlikely that $T_{\omega_{R_{ref}}\omega_R} = 1$ and $T_{\omega_{L_{ref}}\omega_L} = 1$ are inherent characteristic of such system.

In the following sections the real properties of this system is analyzed and different methods are proposed to make it behave closer to the ideal model.

2.6 Robot + Actuator Dynamics

In this section, properties of the actuator + robot dynamics will be discussed in more details. For a system shown in 2.9 one can derive equations as expressed Eq 2.34 to Eq 2.35:

$$\begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} = J_{m2 \times 2} T_{2 \times 2} K_{m2 \times 2} i - J_{m2 \times 2} T_{2 \times 2} \beta_{2 \times 2} \omega_w \quad (2.34)$$

$$\dot{i} = -L_{2 \times 2} R_{2 \times 2} i - L_{2 \times 2} K_{2 \times 2} \omega_w + L_{2 \times 2} V \quad (2.35)$$

where J_m , R , T , K , β , L and i matrices are defines in Eq 2.36 through 2.41.

$$J_m = \begin{bmatrix} \frac{1}{m} & 0 \\ 0 & \frac{1}{I} \end{bmatrix} \quad (2.36)$$

$$R = \begin{bmatrix} R_a & 0 \\ 0 & R_a \end{bmatrix} \quad (2.37)$$

$$T = \begin{bmatrix} \frac{1}{r} & \frac{1}{r} \\ \frac{l}{r} & -\frac{l}{r} \end{bmatrix} \quad (2.38)$$

$$K = \begin{bmatrix} K_b & 0 \\ 0 & K_b \end{bmatrix} \quad (2.39)$$

$$\beta = \begin{bmatrix} \beta & 0 \\ 0 & \beta \end{bmatrix} \quad (2.40)$$

$$L = \begin{bmatrix} \frac{1}{L_a} & 0 \\ 0 & \frac{1}{L_a} \end{bmatrix} \quad (2.41)$$

$$i = \begin{bmatrix} i_{a1} \\ i_{a2} \end{bmatrix} \quad (2.42)$$

Assuming $L_a \approx 0$, one can approximate the transfer function matrix of the system shown in Fig 2.10 as

$$P_{\omega v} = \begin{bmatrix} P_{\omega v11} & P_{\omega v12} \\ P_{\omega v21} & P_{\omega v22} \end{bmatrix} \quad (2.43)$$

where,

$$P_{\omega v11} = P_{\omega v22} \approx \frac{a(s + z_1)}{(s + p_1)(s + p_2)} \quad (2.44)$$

$$P_{\omega v12} = P_{\omega v21} \approx \frac{ds}{(s + p_1)(s + p_2)} \quad (2.45)$$

Gains, poles and zeros are approximately located at

$$a = \frac{K_m}{JL_a} \quad (2.46)$$

$$d = \frac{K_m(J_2 - J_1)}{J_1 J_2 L_a} \quad (2.47)$$

$$p_1 \approx \frac{2(R_a \beta + K_b K_m)}{R_a J_1} \quad (2.48)$$

$$p_2 \approx \frac{2(R_a \beta + K_b K_m)}{R_a J_2} \quad (2.49)$$

$$z_1 \approx \frac{(R_a \beta + K_b K_m)}{R_a J_2} \quad (2.50)$$

where, J , J_1 and J_2 are inertial parameters which are used to model mass and inertia of the robot. These parameters are expressed as

$$J = \frac{J_1 J_2}{J_1 + J_2} = \frac{2I m r^2}{2I + l^2 m} \quad (2.51)$$

$$J_1 = m r^2 \quad (2.52)$$

$$J_2 = \frac{2r^2 I}{l^2} \quad (2.53)$$

Table 2.1 describes the physical representation of each parameter along with the nominal value of them. Further numerical calculations and simulations are based upon the nominal plant.

Figure 2.11 and Figure 2.12, respectively depict the Singular value and Bode plot of P_d .

From Figure 2.12 one can easily conclude that, depending on the application, neglecting the dynamics can have drastic outcomes. Before proceeding further, performance metrics have to be selected to assist us in in-depth analysis of the plant.

2.6.1 Plant Characteristics

In general, when designing and analyzing a system, one needs to satisfy a performance goal or goals. These goals are quantified using performance metrics. Based on

Table 2.1: Dynamic Model Parameter Description and their Nominal Values

Parameter	Description	Nominal Value
K_m	Torque Constant	0.0487 $N.m/Amp$
L_a	Armature Inductance	$0.64 \times 10^{-3} H$
R_a	Armature Resistance	0.27 ohm
r	Wheel Raduis	0.1 m
m	Mass	30 Kg
I	Interia	0.83 $Kg.m^2$
l	Distance between the wheels	0.5 m
β	Friction Constant	0.021 $N.m.s$
K_b	Back EMF Constant	0.0487 $V/(rad/sec)$

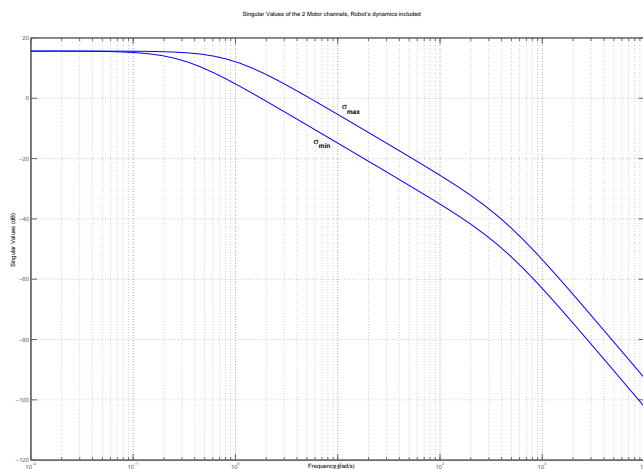


Figure 2.11: Singular Value plot of Mobile Robot Dynamics

previous discussions, we need this system to look close to $I_{2 \times 2}$. This means there are two important factors to consider:

- *Bandwidth*

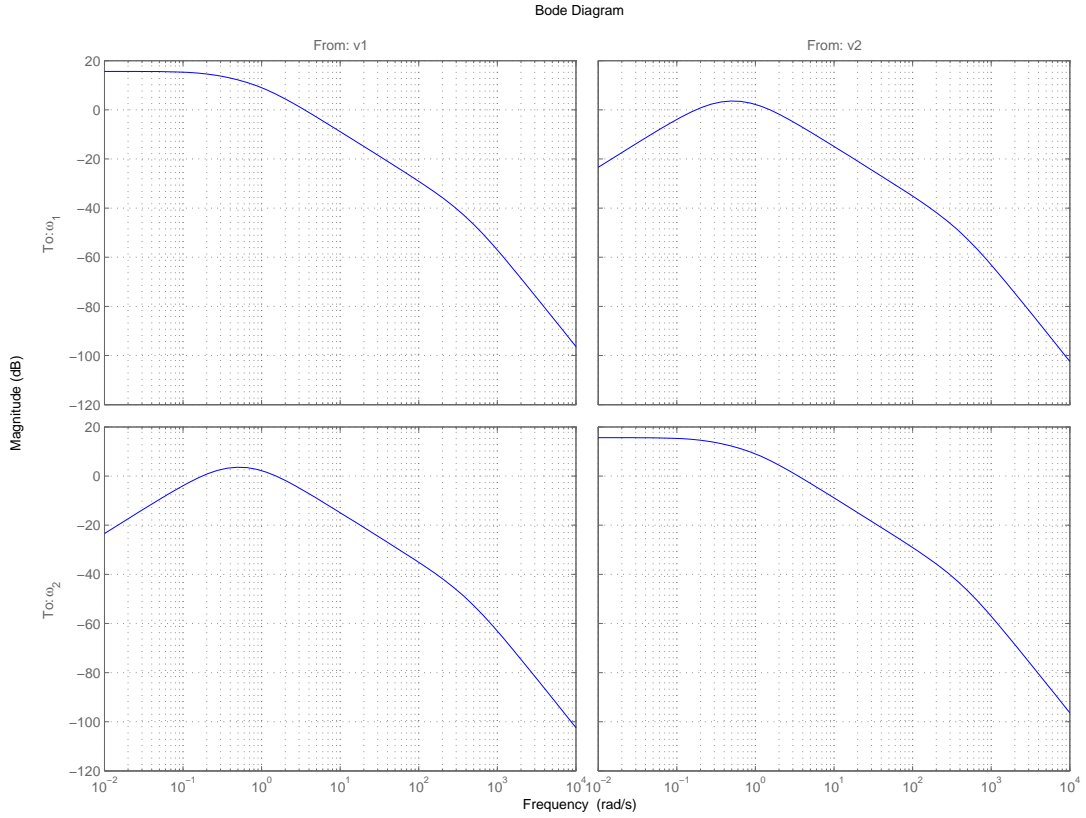


Figure 2.12: Bode Magnitude Plot of Mobile Robot Dynamics

- *Coupling*

Bandwidth is a measure of system's speed, larger bandwidth generally means less response time. In other words, *Bandwidth* measures the frequency range at which the system behaves close to a constant, and is easier to be controlled.

Bandwidth can have different definitions based on the case. In this document, **3dB Bandwidth** of plant's minimum *Singular Value* will be used as a performance metric which is defined as:

$$|\sigma_{min}(\omega_{3dB})| = \frac{|\sigma_{min}(0)|}{\sqrt{2}} \quad (2.54)$$

Coupling is the behavior of the off-diagonal elements in the transfer function matrix, while it is not considered as a metric by itself. However, it is crucial for it to get quantified.

Based on bode plot of the system (Figure 2.12) it is clear that this system has small coupling at low and high frequencies with a peak in the middle. As discussed before, ideally this term has to be small compared to the diagonal term, which justifies using the following ratio as a measure of coupling.

$$C_{ratio} = \frac{|P_{12}(\omega)|}{|P_{11}(\omega)|} \quad (2.55)$$

In this equation, smaller C_{ratio} means smaller coupling, thus better plant characteristics. It should be noted that each of these metrics can have slightly different meaning for different type of systems. A *desirable plant*, would be a system with high bandwidth and small coupling. In following sections designing a robot with desirable characteristics is discussed in details.

2.6.2 Power

It was previously mentioned that it is common for robotic scientists to neglect robot and actuator dynamics based on the concept that *Motors are powerful enough*. In order to have an in depth discussion about this statement, *Power* should be defined in terms of motor parameters. Using DC-Motor model derived in Section 2.4, dc power can be derived as

$$P(0) = \tau(0)\omega(0) \quad (2.56)$$

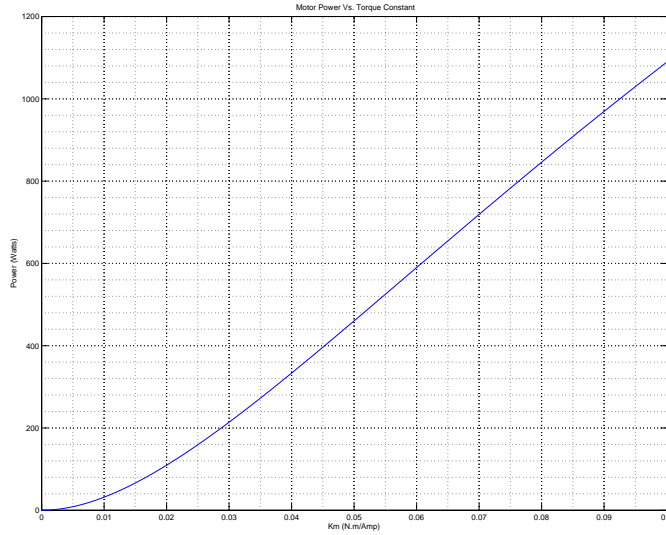


Figure 2.13: Variation of Power Vs. K_m

where,

$$\tau(0) = \frac{K_m \beta}{\beta R_a + K_m K_b} \times V_a \quad (2.57)$$

$$\omega(0) = \frac{K_m}{\beta R_a + K_m K_b} \times V_a \quad (2.58)$$

$\tau(0)$ and $\omega(0)$ represent the dc torque and speed of the motor, respectively. According to these equations, it is obvious that K_m has direct effect on the power. For further analysis, K_m is used as a mean to manipulate power's value. Figure 2.13 shows the relation between power and K_m for this motor.

2.6.3 Mass

The discussion of power is incomplete without considering *mass*. While a motor is considered powerful for a system with mass m_1 , it may not be powerful, or even

sufficient to move a system with mass $m_2 \gg m_1$. In plant analysis mass is varied along with power and the effects of it on performance metrics are explained.

2.6.4 Plant Analysis

In this section, performance metrics of the plant are investigated with respect to power and mass. By analyzing the results of this section we try to show how it is possible to facilitate a kinematic design by having better plant characteristics. All simulations are performed based on the plant equations in Eq 2.34 to Eq 2.35.

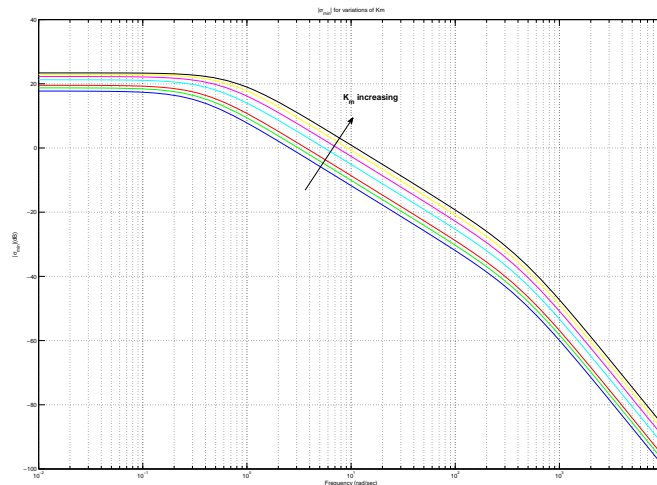


Figure 2.14: Magnitude of Minimum Singular Value for Variations of K_m

Figure 2.14 illustrates the minimum singular value of the plant for variations of K_m . It can be seen that, as K_m increases, dc gain grows larger as well. However, it is not clear what is happening to the *Open Loop Bandwidth*.

In order to clarify, Figure 2.15 plots the *3dB* bandwidth with respect to K_m . As expected, bandwidth is increasing as K_m grows. To confirm our simulation results,

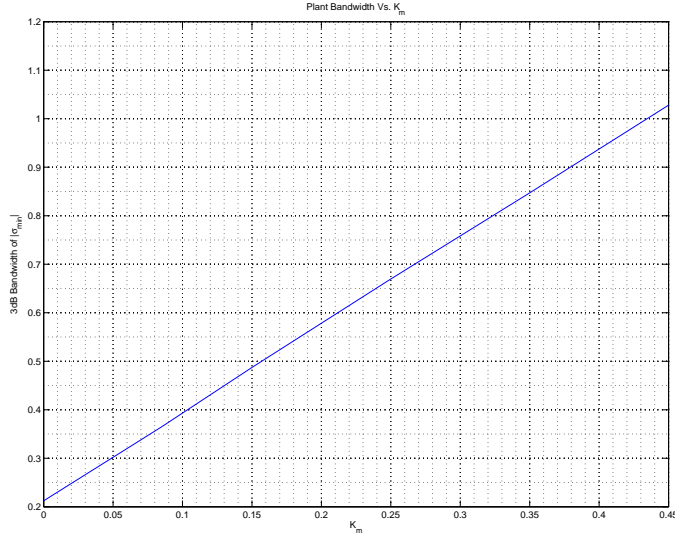


Figure 2.15: Open Loop Bandwidth Vs. K_m

the open loop bandwidth has been calculated analytically in Equation 2.59.

$$BW_{3dB}(\sigma_{min}) \approx \frac{2(R_a\beta + K_bK_m)}{R_aJ_1} \quad (2.59)$$

This confirms that open loop bandwidth increases linearly with K_m .

Plotting the Diagonal with respect to Off Diagonal elements of P_d , as shown Figure 2.16, provides more insight into how the plant behaves. The off-diagonal peak moves further into higher frequencies as K_m increases. This means a larger frequency range of small coupling behavior, which is desirable.

The diagonal and off diagonal elements have exactly similar poles, which means they will have similar behavior in a particular frequency range. This confirms the

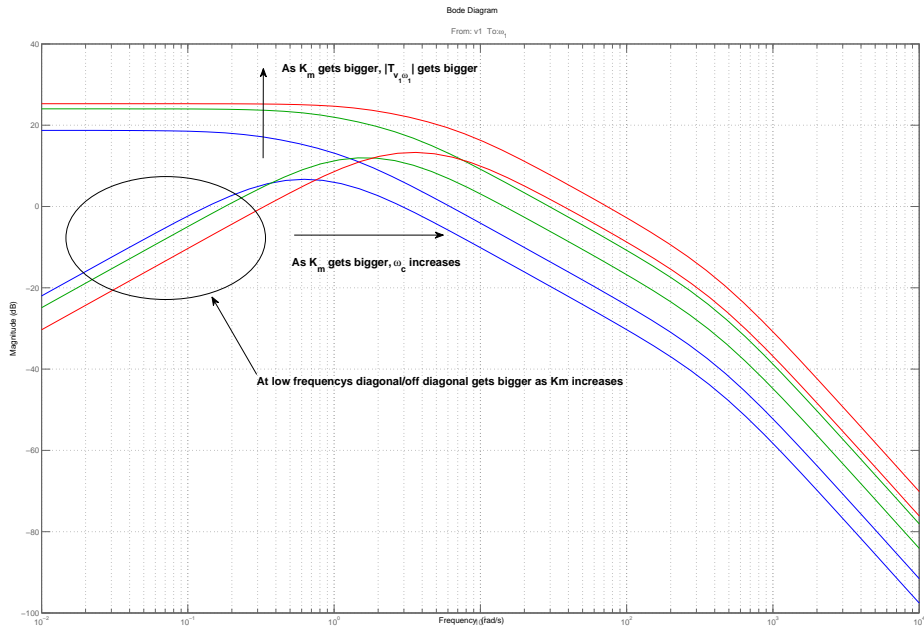


Figure 2.16: Magnitude of Diagonal and Off-Diagonal elements for Variations of K_m
importance of choosing *Coupling Ratio* as a metric.

2.6.5 Robot Aspect Ratio

Figure 2.17 plots the coupling ratio with Vs. frequency for the nominal plant. As it can be seen in this figure, the ratio grows to a constant peak as frequency increases.

The coupling ratio is calculated as

$$C_{ratio} = \left| \frac{P_{\omega v11}}{P_{\omega v12}} \right| = \left| g_1 \frac{s + z_1}{s} \right| \quad (2.60)$$

$$|g_1| = \left| \frac{J_2 + J_1}{J_2 - J_1} \right| \quad (2.61)$$

where, the peak happens at ω_c .

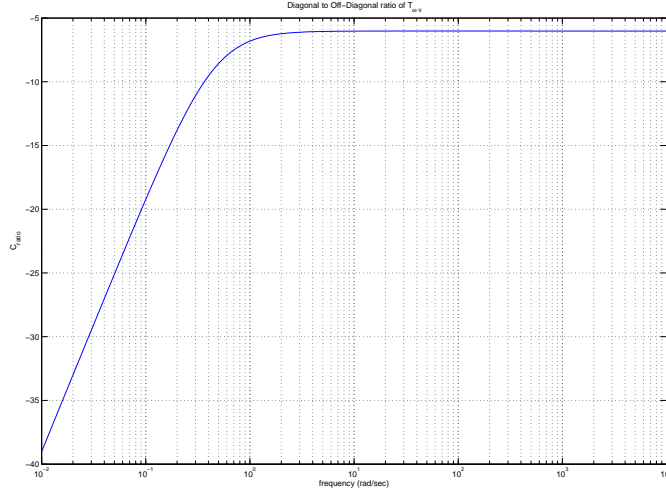


Figure 2.17: Magnitude of Off-Diagonal to Diagonal ratio

The peak value of coupling ratio is defined in Equation 2.61, where it is dependent on the inertial parameters of the system J_1 and J_2 . Substituting inertial parameters into $|g_1|$, the peak can be derived as

$$|g_1| = \left| \frac{\frac{2I}{l^2} - m}{\frac{2I}{l^2} + m} \right| \quad (2.62)$$

It is observed that coupling peak is dependent on mass, inertia and distance between the wheels. In order to gain more insight let's consider the simple mobile robot in Figure 2.18. Assuming an absolute cuboid with length d and width w , Inertia around the z axis is then calculated by

$$I = \frac{m}{12}(w^2 + d^2) \quad (2.63)$$

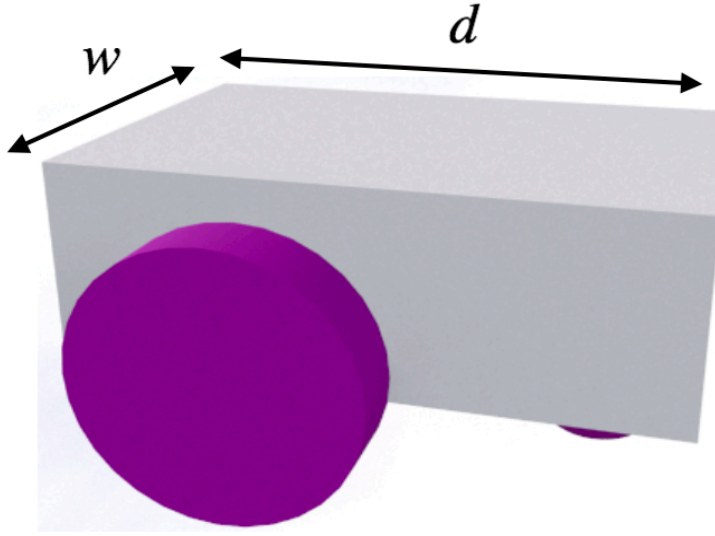


Figure 2.18: Cuboid Shape Mobile Robot

Assuming the distance between the wheels is almost equal to the robot width ($l \approx w$), by substituting I from Equation 2.63 into 2.62, $|g_1|$ can be calculated as :

$$|g_1| = \left| \frac{-5w^2 + d^2}{7w^2 + d^2} \right| \quad (2.64)$$

which shows the dependency of peak coupling on the structure of the robot, more specifically the *aspect ratio* of the robot. The aspect ratio of the robot is defined as :

$$\text{robot aspect ratio (RAR)} = \frac{d}{w} \quad (2.65)$$

Fig 2.19 depicts how peak coupling changes as we change the aspect ratio. As the aspect ratio grows, peak coupling reaches 0 at $\frac{d}{w} = \sqrt{5}$, and as we deviate from this point the peak grows to larger values. This means an aspect ratio of $\sqrt{5}$ would ensure zero coupling for the robot, assuming the robot has an absolute cuboid shape of course.

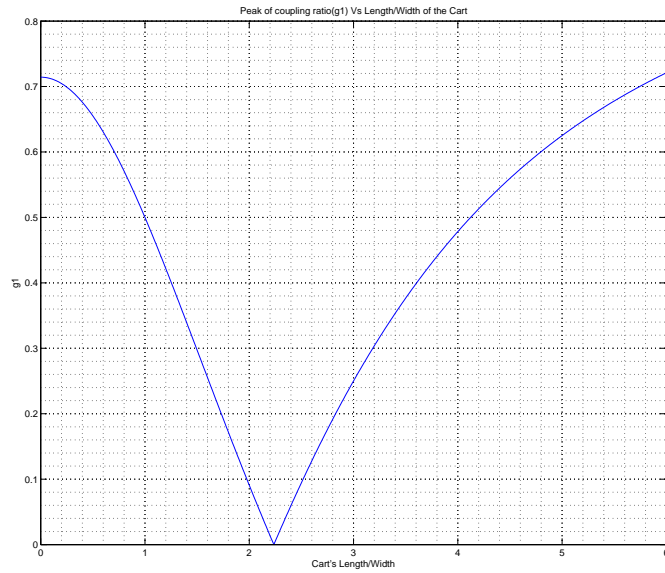


Figure 2.19: Peak coupling ratio behavior Vs. robot's aspect ratio

Figure 2.20 plots a family of systems with different K_m s. As K_m grows, ω_c grows larger, which causes the desirable effect of smaller ratio in wider frequency ranges.

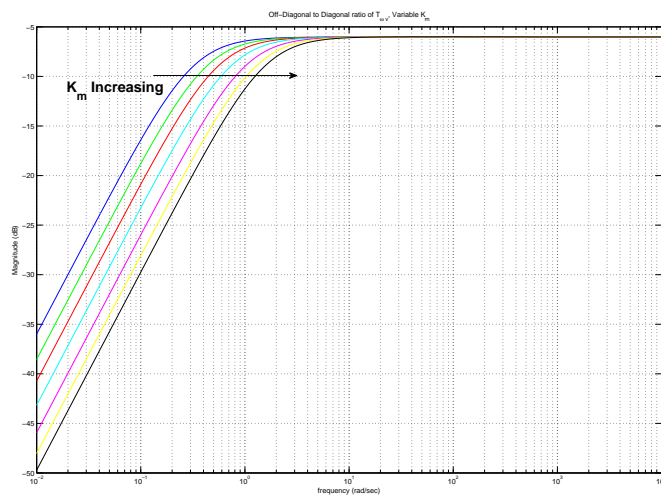


Figure 2.20: Magnitude of Off-Diagonal to Diagonal ratio for Variations of K_m

Similar analysis approach is applied to mass. From Figure 2.21, one can see that changing mass does not change the dc value of σ_{min} . However, as Equation 2.59 suggests, its 3dB bandwidth is inversely related to system's mass (Figure 2.22).

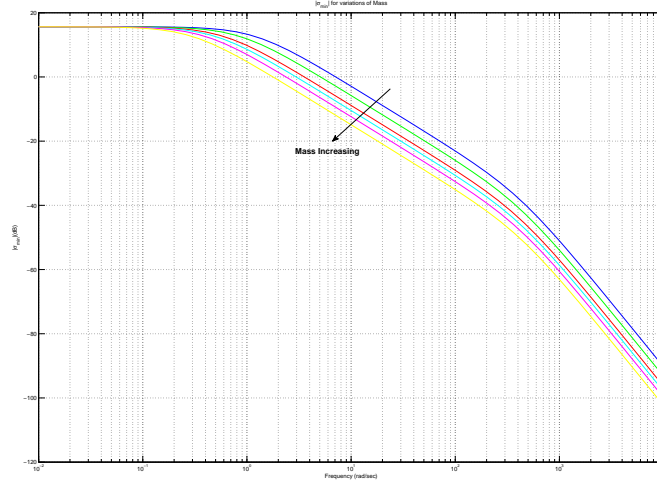


Figure 2.21: Magnitude of Minimum Singular Value for Variations of Mass

Investigating the coupling ratio illustrated in Figure 2.23 confirms that as system becomes heavier we have to expect larger coupling in lower frequencies, making it harder to neglect dynamics.

Before answering the questions, it is worth to summarize our analysis:

- Peak Coupling is related to the structure of the robot with zero value at $\frac{d}{w} = \sqrt{5}$.
- Open Loop Bandwidth is directly proportional to K_m which mean it's proportional to *Power*.
- Open Loop Bandwidth is inversely proportional to mass.
- As Power increases the coupling becomes less significant in lower frequencies.

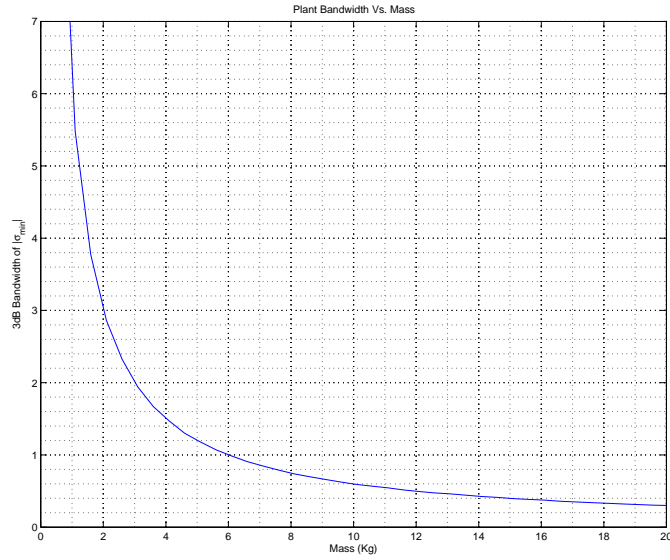


Figure 2.22: Open Loop Bandwidth Vs. *mass*

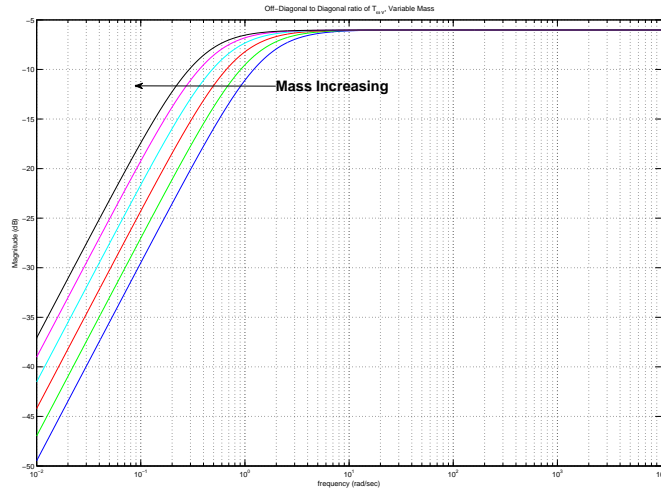


Figure 2.23: Magnitude of Diagonal and Off-Diagonal elements for Variations of Mass

- As mass grows couplings becomes more significant in lower frequencies.

From all of the above one can conclude that the robot can be designed to facilitated

a kinematic control design, more power, smaller mass and an optimum aspect ratio is all that is needed.

2.7 Conclusion

In this chapter, mathematical modelling of a differential drive mobile robot was discussed. Furthermore, the differences and limitations of both dynamic and kinematic models were explained. The detailed dynamic model of the Mobile robot with torque coupling is then introduced followed by the effects of power, mass and aspect ratio of the robot on Bandwidth and coupling characteristics of the plant. Finally, using all this discussion it's addressed how can one design a mobile robot system to facilitate kinematic control design.

Chapter 3

DYNAMICS CONTROL DESIGN

This chapter is dedicated to address the control of the Mobile Robot Dynamics (Inner Loop). Decentralized control architecture based on P and PI controllers is proposed and applied to the Dynamics plant. One mode of the outer loop is briefly discussed, allowing us to analyze the relation between the inner loop (Dynamics) and outer loop (Kinematics). Analyzing such relation results in answering the first two fundamental questions:

1. When is the Kinematic model sufficient?
2. When is the Dynamic model essential?

In section 3.3 the limitation of a decentralized control architecture is exposed, and a rule of thumb based on the aspect ratio of the robot is derived, hence answering the third fundamental question : ” When is the Decentralized control sufficient?”.

Finally a centralized control architecture (LQR) is proposed and implemented, confirming that it’s possible to overcome decentralized control limitations using centralized scheme. maximum error

3.1 Decentralized Control

In this section different schemes of decentralized controller are implemented in order to control the dynamic plant of the mobile robot. The block diagram of such implementation is shown in Figure 3.1.

The plant (2-DC motors + Mobile Robot Dynamics) is governed by Eq 2.34 to Eq 2.35 through out the whole chapter, the controller is specifically defined in each section.

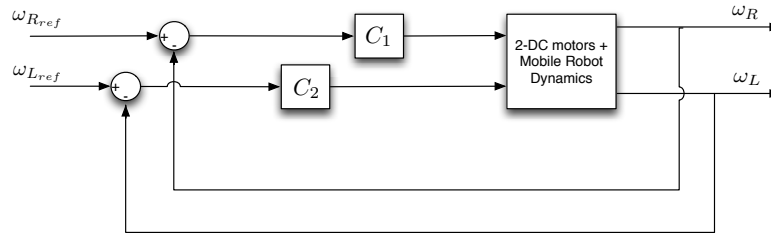


Figure 3.1: Decentralized Controller Architecture for Speed Control

Ideally the motors on the robot are identical, which justifies for C_1 and C_2 to be equal to each other.

3.1.1 Proportional Controller

Proportional or P Controller is the simplest form of decentralized control, where $C_1 = C_2 = K$ and K is just a gain. Figure 3.2 plots how the diagonal and off diagonal elements of $T_{\omega_{ref}\omega}$ change as the proportional gain changes, as K increases:

- Steady state error decreases .
- Peak of the off-diagonal element moves to higher frequencies.
- Off-diagonal element gets smaller in lower frequencies.

As it can be seen in Figure 3.3, increasing the proportional gain also increases the dc gain of minimum singular value.

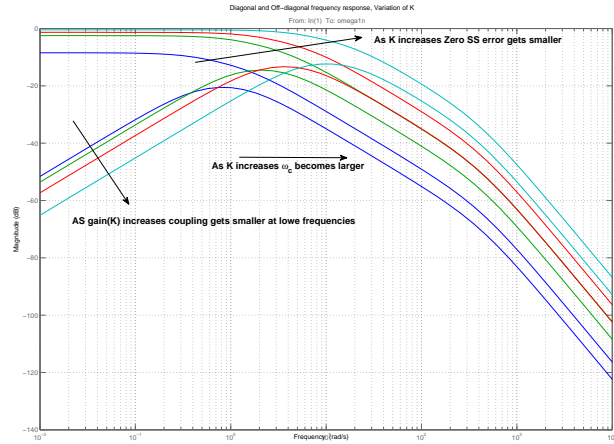


Figure 3.2: Magnitude of Diagonal and off-Diagonal elements for variations of K

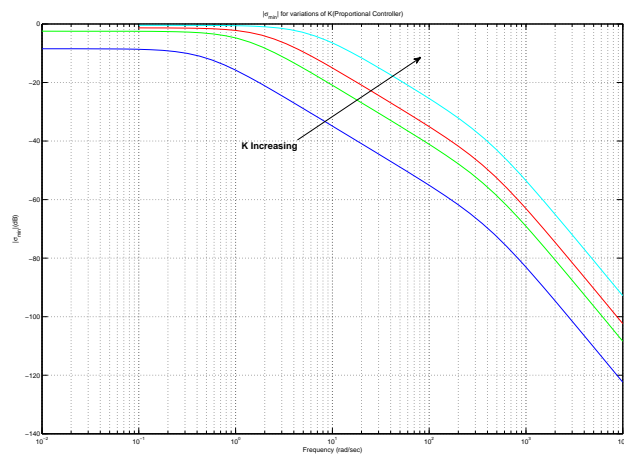


Figure 3.3: Minimum singular value for variations of K

Bandwidth of the closed loop system grows linearly with respect to K, as shown in Figure 3.4.

Off-diagonal to diagonal ratio is plotted in Figure 3.5. As K increases, the peak of the coupling ratio moves to higher frequencies. This will result in smaller ratios at

low frequencies, hence better closed loop behavior.

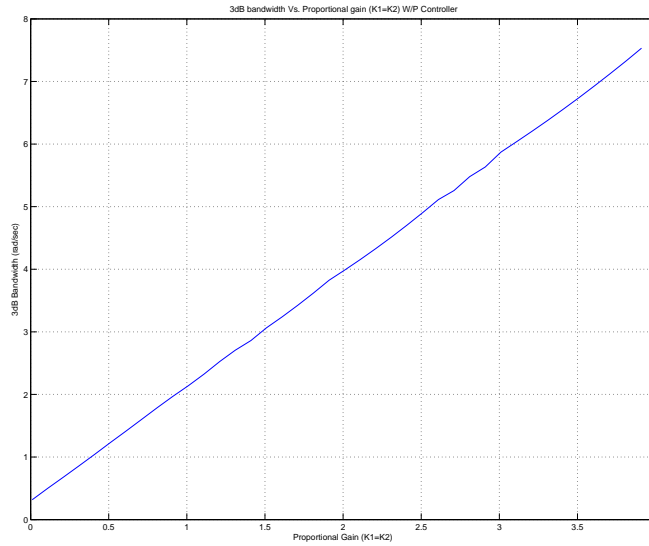


Figure 3.4: Bandwidth of the system Vs. Proportional gain (K)

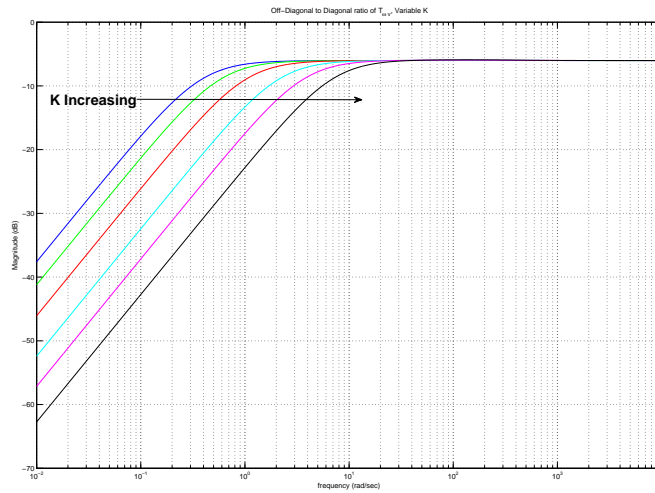


Figure 3.5: Decentralized Controller Architecture for Speed Control

One can argue that desired performance specifications are achievable if K is arbitrary large. However, in practice we are always limited by non-linearities such as *Saturation* and amplification of *High frequency Noise*. The other downside of using a P controller is the non-zero steady state error.

In order to eliminate the steady state error a PI architecture is implemented in the next section.

3.1.2 PI Controller

A PI controller is essential to eliminate the steady state error and follows this general structure :

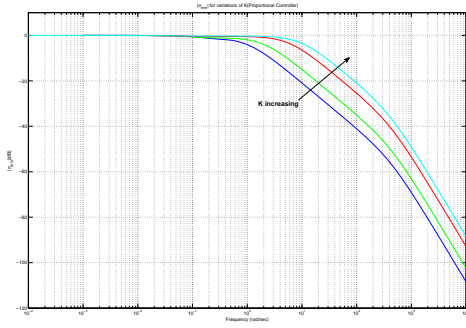
$$C_1 = C_2 = K_p + \frac{K_i}{s} \quad (3.1)$$

where, K_p and K_i are the proportional and integral gain respectively.

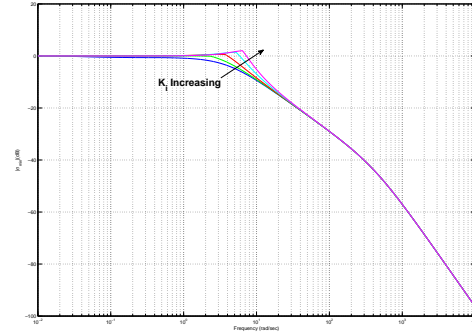
Same analysis approach is followed for both parameter. Figure 3.6 illustrate how σ_{min} changes as K_p and K_i change. It is worth to mention that increasing each one of them increases the bandwidth.

Proportional gain has a more dominant effect compared to the integral gain as shown in Figure 3.7. It should be noted that increasing K_i causes bigger transients as well, which may not be desirable. Closed loop dc gain of the system is 0 dB, indicating zero steady state error to input commands as expected.

Similar to P controller, increasing K_p and K_i moves the coupling peak to higher



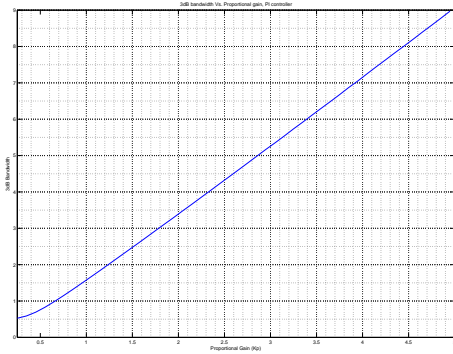
(a)



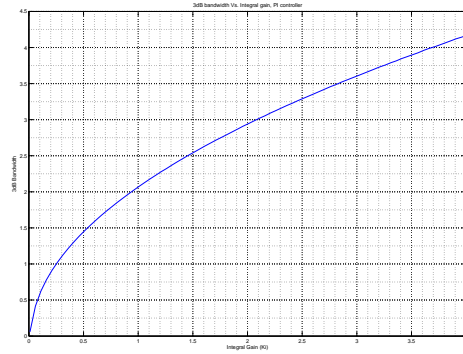
(b)

Figure 3.6: Magnitude of Diagonal and off-Diagonal elements for variations of (a) variations of K_p and (b) variations of K_i

frequencies, as illustrated in Figure 3.8. However, there are two important facts to consider:



(a)



(b)

Figure 3.7: (a) Bandwidth Vs. K_p (b) Bandwidth Vs. K_i

- Increasing K_p does not have a considerable effect on coupling ratio at very low frequencies.
- Increasing K_i causes a transient at the coupling peak frequency, resulting in bigger coupling in that frequency.

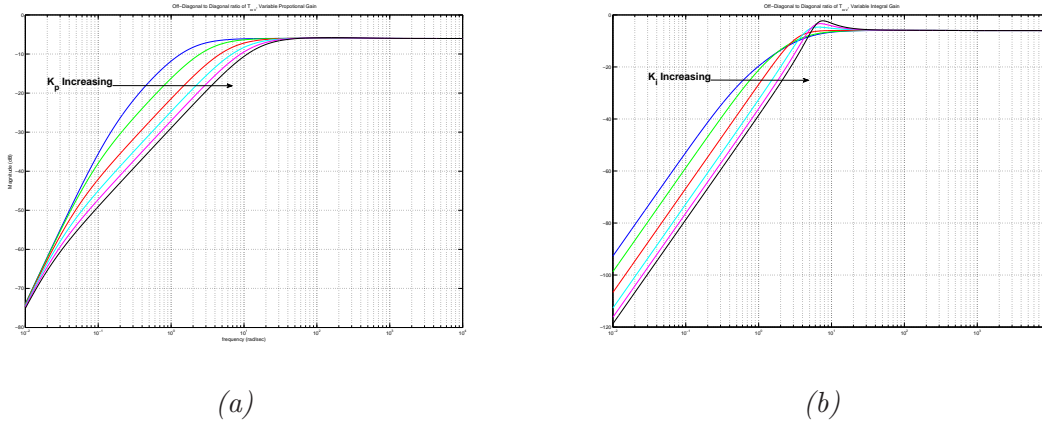


Figure 3.8: (a) Bandwidth Vs. K_p (b) Bandwidth Vs. K_i

3.2 Inner Loop (Dynamics) Vs. Outer Loop (Kinematics)

Now that decentralized control schemes are analyzed for such system it's time to answer the fundamental question of when is the kinematic-only design is sufficient, in order to do so first there should be discussion about outer loop plant.

3.2.1 Cartesian Stabilization

Displacement control is one the modes of operation we discussed in chapter 1, in this mode the objective of the robot is to start form an initial point ($[x\ y]^T$) and move to a desired point ($[x_{ref}\ y_{ref}]^T$), without specifying the path between the points. In order to facilitate linear thinking one can define a system with inputs $[s_{ref}\ \theta_{ref}]^T$ and outputs $[s\ \theta]^T$, where s is the linear displacement along saggital axis and θ is the orientation of the robot [20], given by :

$$\dot{s} = v \tag{3.2}$$

$$\dot{\theta} = \omega \tag{3.3}$$

block diagram of such system is shown in Fig 3.9. The outer loop controller can be designed based on any classical controller which makes addressing the problem much easier. In practice however measuring s is impossible and commanding s_{ref} is meaningless. However these problems can be addressed using the right calculations.

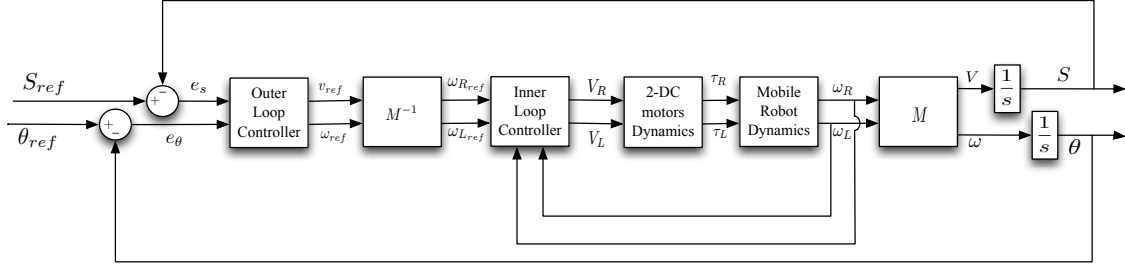


Figure 3.9: Displacement Control Block Diagram from S_{ref} and θ_{ref} to s and θ

As stated s is immeasurable but e_s can be calculated, consider the robot in Fig 3.10, the robot positioning problem will be solved if $\Delta l \rightarrow 0$.

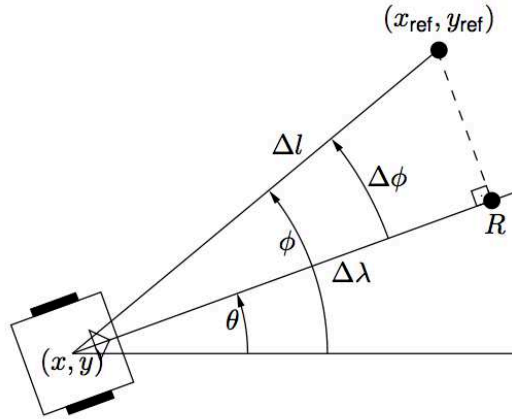


Figure 3.10: Mobile Robot in Cartesian Stabilization mode

In order for the robot to go to the desired position s_{ref} and θ_{ref} should be generated such that $\Delta \lambda$ and $\Delta \phi$ go to zero, meaning $e_s = \Delta \lambda$ and $e_\theta = \Delta \phi$, thus if

the controller converges s and θ error to zero the displacement problem of the system is solved. One can generate θ_{ref} and e_s using the following equations

$$\theta_{ref} = \tan^{-1} \left(\frac{\Delta y_{ref}}{\Delta x_{ref}} \right) \quad (3.4)$$

$$e_s = \Delta l \cdot \cos(\Delta\phi) = \sqrt{(\Delta y_{ref})^2 + (\Delta x_{ref})^2} \cdot \cos \left[\tan^{-1} \left(\frac{\Delta y_{ref}}{\Delta x_{ref}} \right) - \theta \right] \quad (3.5)$$

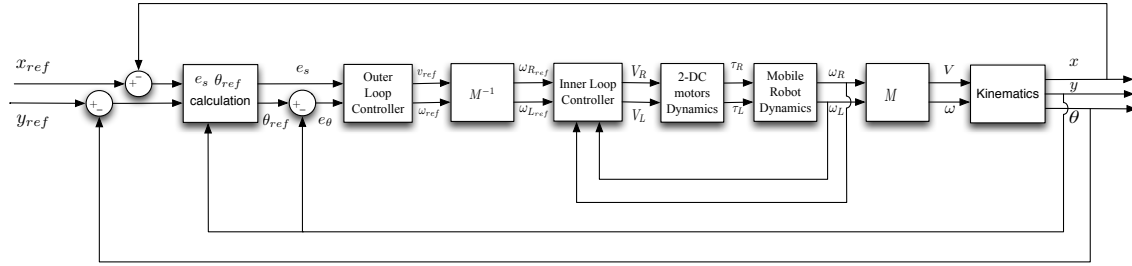


Figure 3.11: Positioning System (Displacement Control) Block Diagram

The complete diagram of a positioning system using this method is shown in Fig 3.11, it should be noted that although using linear controller is simpler but the effects of moving the non-linearities outside the loop may be undesirable, which is not discussed here.

Using decentralized proportional controller for both inner loop and outer loop system one can analyze how changing the bandwidth of the inner loop affects the whole system. As inner loop system gets faster with respect to the outer loop, the actual system becomes more similar to the ideal Kinematic model, meaning it is easier to neglect the dynamic and design based on kinematic thinking.

3.2.2 Kinematic Design Limitations

Fig 3.12 shows the maximum error of σ_{min} between the actual system (Kinematic + Dynamics) and an Ideal system (Kinematics Only), using nominal value parameters

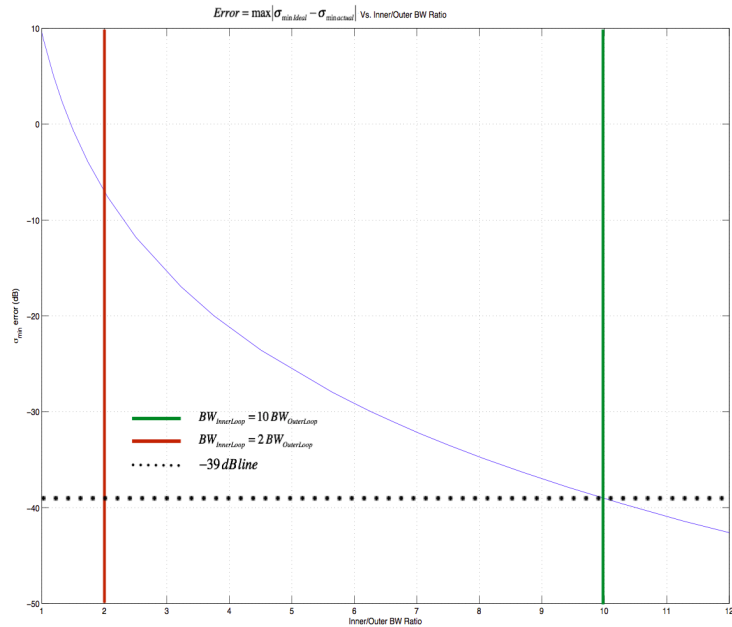


Figure 3.12: Error between ideal (Kinematic) and actual (Kinematic + Dynamics) system Vs. BW ratio

given in chapter 2. It is observed that as the bandwidth of the inner loop grows the error becomes smaller, allowing us to answer the first two fundamental questions:

1. When is the kinematic model sufficient?

If the faster inner loop is much faster than the slower outer loop the kinematic model is sufficient

As a rule of thumb : $BW_{InnerLoop} \geq 10BW_{OuterLoop}$ (green line) will yield an error less than $-39dB$

2. When is the dynamic model essential?

If the faster inner loop is not fast enough compared to the slower outer loop then considering dynamic model is essential

As a rule of thumb: $BW_{InnerLoop} \leq 2BW_{OuterLoop}$ (red line) can yield an error up to $10dB$

3.3 Decentralized Control Limitation

From previous discussions we know that making the inner loop fast is desirable, but of course operating at higher frequencies comes with a price, in our system this price is the sensitivity function. Defining the sensitivity as

$$S = (I + PK)^{-1} = \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \quad (3.6)$$

It is critical for us that the peak of the elements of S are small in our frequency of operation and also the off-diagonal element is much smaller than the diagonal element so that the cross coupling is minimum.

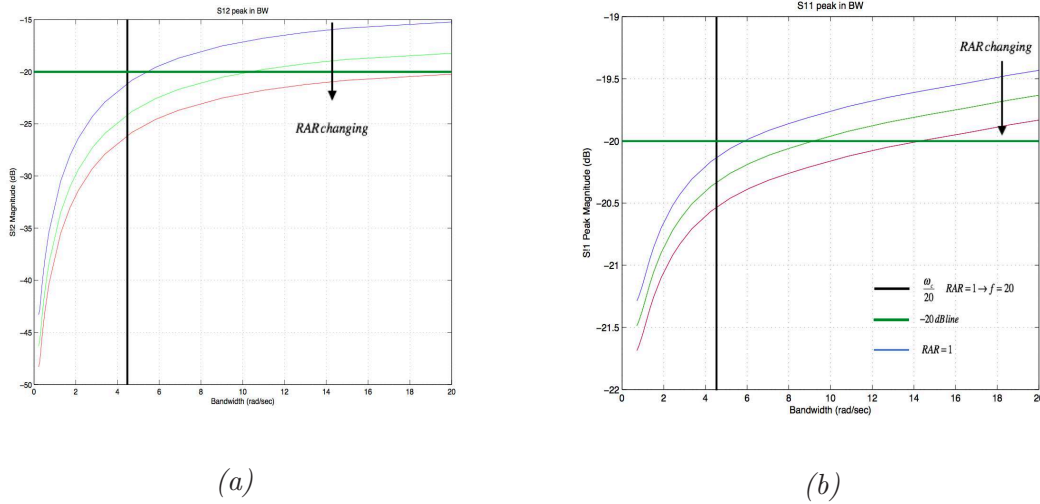


Figure 3.13: (a) $\max|S_{12}|$ Vs. BW (b) $\max|S_{11}|$ Vs. BW

Fig 3.13 plots the peak magnitude of these elements for systems with different bandwidths, as bandwidth increases the peak becomes bigger which is undesirable.

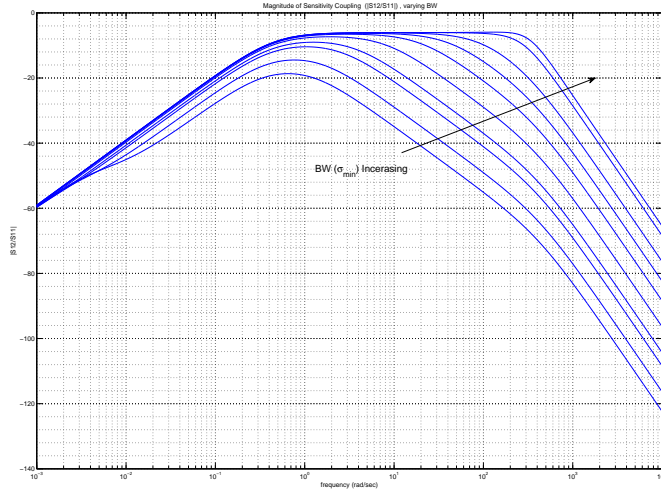


Figure 3.14: $\left| \frac{S_{12}}{S_{11}} \right|$

Fig 3.14 shows the off-diagonal to diagonal ratio of S , as the bandwidth is increasing we see the ratio getting bigger, and reaches a constant peak. The peak of this ratio in the operating bandwidth is of great importance. Fig 3.15 plots this peak, which also grows with bandwidth increasing, reaching a maximum of p_s , as was expected. It is safe to say that increasing bandwidth arbitrarily can result in worse sensitivity characteristic.

Now that we have enough information we have to answer our third question:

3. When is the decentralized controller sufficient?

If the inner loop dynamics plant operates far enough from the maximum coupling frequency (ω_c) then a decentralized controller can address our control problem and deliver desired closed loop characteristics

As a rule of thumb: $BW < \frac{\omega_c}{f}$ will yield $|S_{11}| \text{ and } |S_{12}| < -20dB$, $\left| \frac{S_{12}}{S_{11}} \right| < -20dB$

the rule of thumb for a system with aspect ratio of 1 (blue line) along with $-20dB$ lines are shown in Fig 3.13 and 3.15.

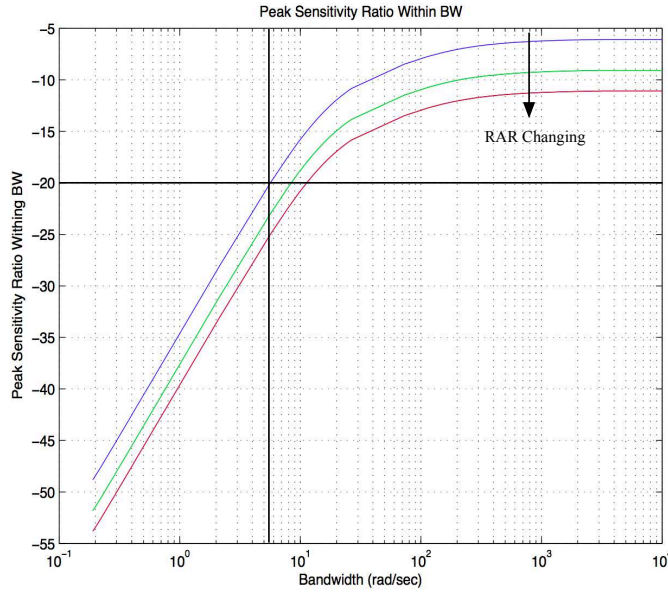


Figure 3.15: Peak $\left| \frac{S_{12}}{S_{11}} \right|$ within BW Vs BW

An important fact is that p_s depends on robot's structure, meaning as aspect ratio changes this peak will move higher or lower, and may call for a different rule of thumb, hence the need for factor f .

Fig 3.16 shows the behavior of p_s versus the aspect ratio of the robot, similar to g_1 in plant, p_s gets smaller as we reach $length/width = \sqrt{5}$, meaning around that point one can use a more tolerant rule of thumb. The rule of thumb proposed was based on an aspect ratio of 1, which by looking at Fig 3.16 we see for systems with smaller aspect ratio ($width > length$) there may be a need for a stricter rule of thumb.

Fig 3.17 plots how the rule of thumb changes as the aspect ratio change, the rule of thumb is designed to deliver a magnitude ratio less than $-20dB$, meaning for a set of systems this is already satisfied by the plant. This means we can operate up to any desired frequency for such systems and have good closed loop specification, of course it is important to note there are many high frequency parameters such as *high*

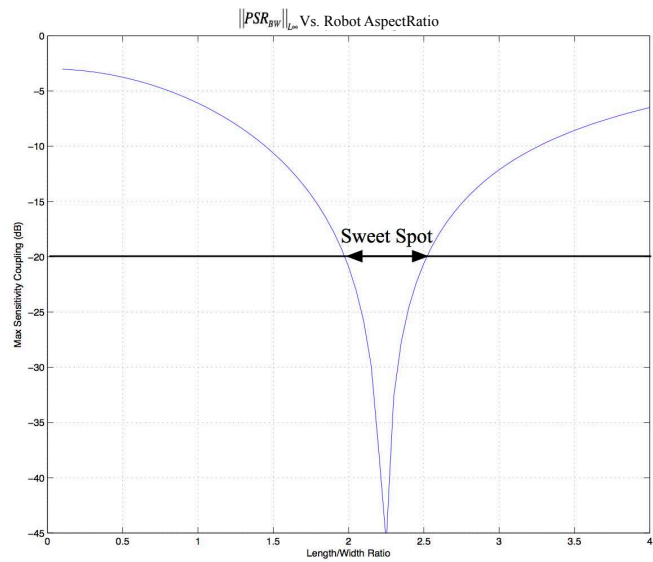


Figure 3.16: p_s Vs. Aspect Ratio

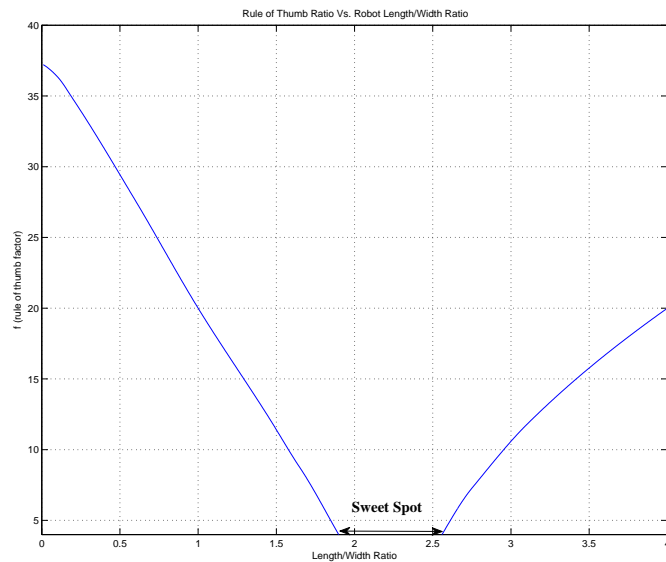


Figure 3.17: Rule of thumb Vs. Aspect Ratio

frequency noise, sensor noise, saturation and non-linearities that are being neglected here.

For boundary systems the rule of thumb is $BW < \omega_c/4$, this means operating at any frequency above this point will not deliver desired specification unless we are meeting the ideal aspect ratio range. This bring us to our last question:

4. When is the centralized controller essential?

If we operate close to the maximum coupling frequency (ω_c) then a centralized controller is essential

As an intuitive rule of thumb: $BW > \omega_c$

3.4 Centralized Control (Linear Quadratic Regulator)

This section is dedicated to design and analysis of a centralized controller for mobile robot dynamics. Controller of choice is a Linear Quadratic Regulator with full state feedback.

The plant is defined in Eq to Eq. In order to achieve zero steady state error to step reference command two integrator have to be augmented to the plant output. The augmented plant, denoted by P_d has the state equation:

$$\dot{x} = Ax + Bu \tag{3.7}$$

where

$$u = u_p \tag{3.8}$$

$$x = \begin{bmatrix} x_I \\ x_p \end{bmatrix} = \begin{bmatrix} x_I \\ y_p \\ x_r \end{bmatrix} \tag{3.9}$$

$x_I = [\theta_1 \theta_2]^T$ are the integrator states and x_r is the rest of the plant's states other

than plant outputs y_p . Now by minimizing the quadratic cost function one can reach a optimal control law for such plant:

$$J(u) = \frac{1}{2} \int_0^{\infty} (x^t Q x + \rho u^t u) dt \quad (3.10)$$

where $\rho = 0.01$ and $Q = \text{diag}[1, 1, 1, 1, q_{I_{a1}}, q_{I_{a2}}, 2, 2]$. $q_{I_{a1}}$ and $q_{I_{a2}}$ penalize the armature currents allowing for different coupling characteristics as discussed further in the following section. Selecting $u = -Gx$ where $G = [G_{yp} \ G_r \ G_I]$ will result in an LQR architecture shown in Fig 3.18.

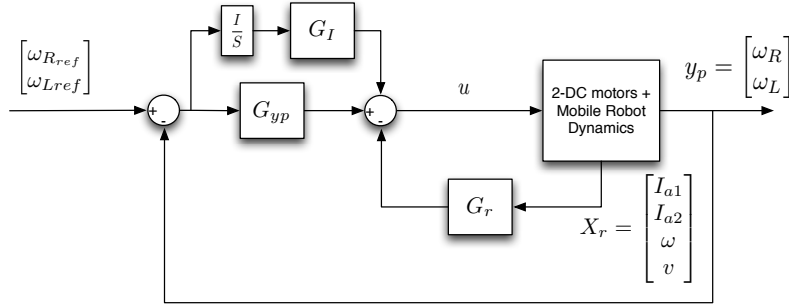


Figure 3.18: Dynamics Plant with a Linear Quadratic Regulator

As stated in section 3.3, the closed loop coupling ratio ($\left| \frac{T_{\omega_{ref}12}}{T_{\omega_{ref}11}} \right|$) has a constant peak at high frequencies, which is dependent on the aspect ratio of the robot. Using a decentralized controller, one can increase the closed loop peak frequency (ω_C) by increasing the bandwidth of the system (Fig 3.19). While increasing the bandwidth results in some desirable closed loop characteristics, as discussed in section 3.3 can cause undesirable properties as well. On the other hand, using a centralized LQR controller and a proper selection of Q, it is possible to shape the closed loop coupling ratio.

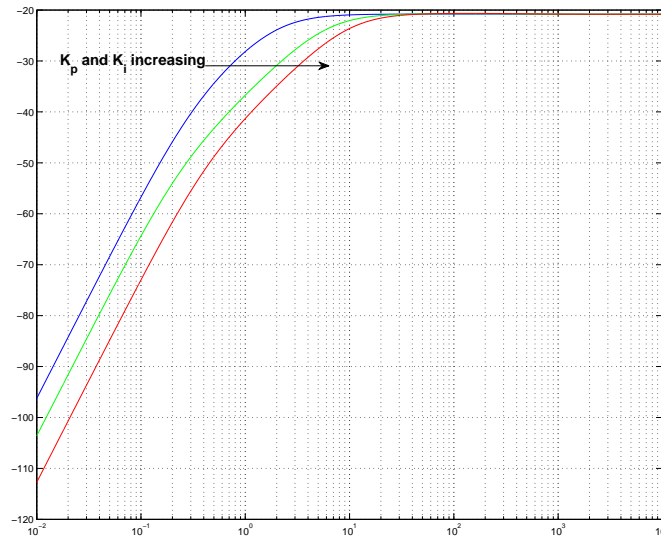


Figure 3.19: Closed loop coupling ratio with decentralized control

Figure 3.20 depicts the closed coupling ratio for family of LQR controllers. It can be observed that by manipulating Q , one can not only reduce the peak magnitude, but change the behavior of the coupling ratio in the frequencies higher than the peak as well, overcoming the limitations of decentralized control architecture.

3.5 Summary and Conclusion

In this chapter, different control schemes for the dynamic model were analyzed. The relation between the inner loop dynamics and outer loop kinematics was discussed, leading to answers for the first fundamental questions : ” When is the kinematic model sufficient? ” and ” When is the dynamic model essential? ”

Different performance aspects of decentralized P and PI controllers, along with their differences, were studied. Additionally, the limitations of using a decentralized control were explained. Consequently, last two fundamental questions were answered:

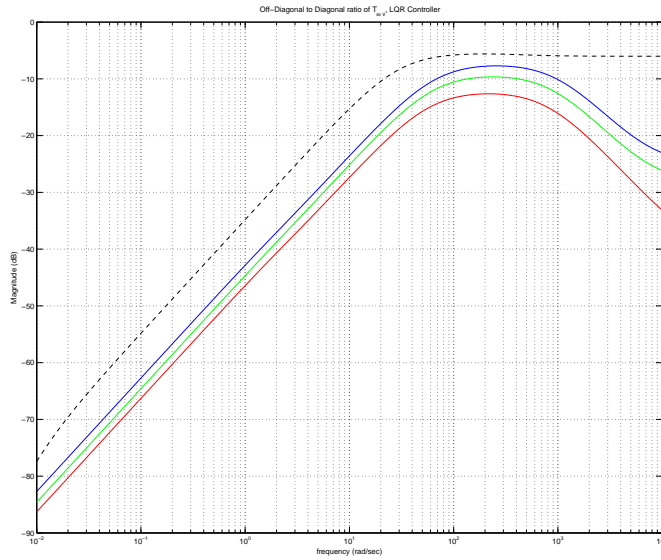


Figure 3.20: Closed loop coupling ratio with centralized control

” When is the decentralized control sufficient? ” and ” When is the centralized control essential?”

Finally, by implementing a centralized control architecture (LQR) and performing further analysis, it was possible to show that the centralized control is able to overcome limitations of the decentralized scheme.

Chapter 4

TRAJECTORY PLANNING

4.1 Planning

In an industrial setting or in the field a mobile robot needs a *trajectory* to follow and complete a goal. Planning this trajectory can be done in many different ways to satisfy conditions such as minimum distance, minimum travel time, etc. However, in general, this task can be broken down into finding a *path* and define a required *timing law* on such path.

Trajectory planning is a considerably challenging topic. What can make this topic even more challenging topic in non-holonomic systems is the fact that not only it has to meet the boundary conditions. However, the non-holonomic constraint has to satisfied at all points.

In this chapter path planning for a non-holonomic mobile robot and timing law is discussed. A flat output system and its characteristics is then defined. Finally admissible trajectory planning is thoroughly discussed.

4.2 Trajectory:Path and Timing law

Consider a trajectory $q(t)$, $t \in [t_i, t_f]$ that guides a mobile robot from initial configuration $q(t_i) = q_i$ to final configuration $q(t_f) = q_t$ in time $T = t_i - t_f$. This trajectory can be broken down into a geometric path $q(g)$, where $\frac{dq(g)}{dg} \neq 0$ and a

timing law $g = g(t)$ where $g(t)$ is monotonically increasing function of time on $[t_i, t_f]$, i.e. $\dot{g}(t) \geq 0$. Generalized velocity vector can then be obtained as

$$\dot{q}(t) = \frac{dq}{dt} = \frac{dq}{dg} \frac{dg}{dt} = q' \dot{g} \quad (4.1)$$

where q' is the tangent vector to the path.

4.3 Effects of Kinematic Constraint

A kinematic constraint such as 2.5 can be re expressed as

$$V^T(q)\dot{q} = V^T(q)q'\dot{g} = 0 \quad (4.2)$$

If $g(t)$ is strictly increasing, i.e. $\dot{g}(t) > 0$, then it is trivial that

$$V^T(q)q' = 0 \quad (4.3)$$

has to hold.

Essentially it means that in a mechanical system subject to non-holonomic constraint a geometric path is admissible if and only if it satisfies 4.3. Similar to 2.4, a set of all admissible paths can be derived as a solution to

$$q' = \sum_{i=1}^{n-k} b_i(q)\hat{u}_i = B(q)\hat{\mathbf{u}} \quad (4.4)$$

where, $\hat{\mathbf{u}}$ is the vector of geometric inputs related to kinematic input vector \mathbf{u} by $\mathbf{u}(t) = \hat{\mathbf{u}}(g)\dot{g}(t)$.

In order to acquire a unique admissible path, selecting the geometric inputs for $g \in [g_i, g_f]$ would suffice. In the case of non-holonomic robot, admissible paths must satisfy

$$[\sin \theta \quad -\cos \theta \quad 0]q' = 0 \quad (4.5)$$

Therefore, all the admissible paths can be formulated as

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 \\ \sin \theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \hat{v} \\ \hat{\omega} \end{bmatrix} \quad (4.6)$$

where,

$$v(t) = \hat{v}(g)\dot{g}(t) \quad (4.7)$$

$$\omega(t) = \hat{\omega}(g)\dot{g}(t) \quad (4.8)$$

The kinematic constraint in Equation 4.5 states that an admissible path for a non-holonomic robot should have a tangent aligned with the robot's sagittal axis. In another words, no edges or sharp points are allowed on the path.

4.4 Differential Flatness

Consider a non-linear system defined by

$$\dot{x} = f(x) + g(x)u \quad (4.9)$$

$$y = h(x) + d(x)u \quad (4.10)$$

Such system is *differentially flat* if there exists a set of outputs y , where states x and control inputs u can be expressed as unique functions of y and its derivatives:

$$x = fcn_1(y, \dot{y}, \ddot{y}, \dots, y^{(n)}) \quad (4.11)$$

$$u = fcn_2(y, \dot{y}, \ddot{y}, \dots, y^{(n)}) \quad (4.12)$$

Outputs y are called flat outputs. Cartesian coordinates $[x, y]$ in mobile robots are considered flat outputs, consider geometric model in Equation 4.6, by defining an output Cartesian path $[x(g), y(g)]$ one can calculate the orientation from

$$\theta(g) = atan2(y'(g), x'(g)) + k\pi \quad k = 0, 1 \quad (4.13)$$

where, k defines if the robot is moving forward ($k = 0$) or backward ($k = 1$) and atan2 is a variation of *arctangent*¹ that calculates the angle between the x axis and the line passing through point (x, y) from origin.

The states are then obtained as $q(g) = [x(g) \ y(g) \ \theta(g)]^T$ and the geometric velocity inputs are uniquely defined by Equation 4.14 and 4.15.

$$\hat{v}(g) = \pm \sqrt{x'(g)^2 + y'(g)^2} \quad (4.14)$$

$$\hat{\omega}(g) = \frac{y''(g)x'(g) - x''(g)y'(g)}{x'(g)^2 + y'(g)^2} \quad (4.15)$$

This means that a unique path along with unique velocities can be defined for the robot.

4.5 Conclusion

In this chapter the outer loop path generation problem of the mobile robot was discussed. For this purpose, generating viable speed commands for a desired path had more focus on.

At first, path planning for non-holonomic mobile robots were presented. After defining a flat output system and the features incorporated with it, trajectory planning was fully explained.

¹Using tangent half formula an expression can be derived : $\text{atan2} = 2\arctan(y/(\sqrt{x^2 + y^2} + x))$

SUMMARY AND FUTURE WORK

In this thesis, a thorough discussion on mobile robot control & design, and the problems and limitations incorporated with it, was provided. Additionally, commonly neglected aspects of mobile robot design in the literature were explained. Four fundamental questions were proposed, were answers to them would clarify such neglected aspects.

A thorough study of mobile robot kinematics and dynamics were performed, and the design aspects of a differential drive mobile robot was discussed. The dependency between shape, power and mass of the robot on dynamics and coupling was clearly addressed. Based on such dependencies, facilitating a kinematic-only design through desirable plant characteristics was studied.

Next the relation between the inner loop dynamics and the outer loop kinematics was discussed, leading to answers to the first two fundamental questions proposed earlier:

1. When is the kinematic model sufficient?

When (Faster Inner) Velocity Loop is much faster than (Slower Outer) Position Loop

2. When is the dynamic model essential?

When (Faster Inner) Velocity Loop is not fast enough compared to (Slower Outer) Position Loop

The performance of decentralized control was then studied and the limitation of such control structure was exposed in terms of the closed loop characteristics. Based on such analysis answers were provided to the last two fundamental questions:

3. When is the decentralized control sufficient?

When system operates at low enough frequencies with respect to coupling peak

4. When is the centralized control essential?

When system operates at frequencies close to coupling peak or higher frequencies

Finally a centralized control architecture (LQR Servo) was implemented confirming the possibility of overcoming limitation arising from the centralized control.

In this thesis, many details concerning design and control of mobile robots were discussed and addressed, however mobile robotic is a very vast and complicated field of science, and one can always go into more details about every aspect of it. The following topics are proposed as a guideline for possible future work for this article:

- More complicated inner loop dynamics

As discussed before there are parameters such as surface friction] and saturation that yet to be considered in the dynamic plant, allowing further analysis for more aggressive specification (higher bandwidth, less cross coupling) of such plant. Additionally further analysis on the structured and unstructured uncertainties (parametric/dynamic) of the plant, and robustness of different control scheme to such uncertainties is suggested.

- Outer loop kinematics issues

Position control aspect of mobile robot, such as outer loop control design and performance analysis has yet to be discussed in greater details. A systematic comparison of distinct combination of outer loop and inner loop strategies is highly suggested.

- Hardware Implementation

The proposed material in this document has provided a guide to design and control differential drive mobile robots, while minimizing undesirable characteristics of such system. The next step is to design and implement a robot based on results driven in this thesis. Of course an important discussion which would be complementary to our results is the trade off analysis between desired performance and cost for an actual system.

REFERENCES

- [1] M.E. Rosheim. In the footsteps of leonardo [articulated anthropomorphic robot]. *Robotics Automation Magazine, IEEE*, 4(2):12–14, 1997.
- [2] RD Schraft and G Schmierer. *Serviceroboter*, 1998.
- [3] Joseph L Jones and Anita M Flynn. *Mobile robots: inspiration to implementation*. AK Peters, Ltd., 1993.
- [4] Peter F Bladin. W. grey walter, pioneer in the electroencephalogram, robotics, cybernetics, artificial intelligence. *Journal of clinical neuroscience*, 13(2):170–177, 2006.
- [5] P Marsh. Machines with mobilityjohns hopkins beast robot. *Robots*, 1985.
- [6] S Lawrence Bellinger. Self-propelled random motion lawnmower, March 16 1971. US Patent 3,570,227.
- [7] Joseph F Engelberger. Health-care robotics goes commercial: the helpmateexperience. *Robotica*, 11(06):517–523, 1993.
- [8] Claude Samson and K Ait-Abderrahim. Feedback control of a nonholonomic wheeled cart in cartesian space. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 1136–1141. IEEE, 1991.
- [9] Brigitte d’Andréa Novel, Gianni Campion, and Georges Bastin. Control of non-holonomic wheeled mobile robots by state feedback linearization. *The International journal of robotics research*, 14(6):543–559, 1995.
- [10] Alessandro De Luca and Marika D Di Benedetto. Control of nonholonomic systems via dynamic compensation. *Kybernetika*, 29(6):593–608, 1993.
- [11] G. Klancar, D. Matko, and S. Blazic. Mobile robot control on a reference path. In *Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation*, pages 1343–1348, 2005.
- [12] Shouling He. Feedback control design of differential-drive wheeled mobile robots. In *Advanced Robotics, 2005. ICAR’05. Proceedings., 12th International Conference on*, pages 135–140. IEEE, 2005.
- [13] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: modelling, planning and control*. Springer, 2011.
- [14] A Ollero and O Amidi. Predictive path tracking of mobile robots. application to the cmu navlab. In *Proceedings of 5th International Conference on Advanced Robotics, Robots in Unstructured Environments, ICAR*, volume 91, pages 1081–1086, 1991.

- [15] Julio E Normey-Rico, Juan Gómez-Ortega, and Eduardo F Camacho. A smith-predictor-based generalised predictive controller for mobile robot path-tracking. *Control Engineering Practice*, 7(6):729–740, 1999.
- [16] Gregor Klančar and Igor Škrjanc. Tracking-error model-based predictive control for mobile robots in real time. *Robotics and Autonomous Systems*, 55(6):460–469, 2007.
- [17] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi. A stable tracking control method for an autonomous mobile robot. In *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, pages 384–389 vol.1, 1990.
- [18] Roland Siegwart and Illah Reza Nourbakhsh. *Intro to Autonomous Mobile Robots*. MIT press, 2004.
- [19] PK Padhy, Takeshi Sasaki, Sousuke Nakamura, and Hideki Hashimoto. Modeling and position control of mobile robot. In *Advanced Motion Control, 2010 11th IEEE International Workshop on*, pages 100–105. IEEE, 2010.
- [20] Frederico C Vieira, Adelardo AD Medeiros, Pablo J Alsina, and Antônio P Araújo Jr. Position and orientation control of a two-wheeled differentially driven nonholonomic mobile robot. In *ICINCO (2)*, pages 256–262, 2004.
- [21] A.D. Araujo, P.J. Alsina, and S.M. Dias. Nonholonomic wheeled mobile robot positioning controller using decoupling and variable structure model reference adaptive control. In *American Control Conference, 2006*, pages 6 pp.–, 2006.
- [22] Alessandro De Luca, Giuseppe Oriolo, and Marilena Vendittelli. Control of wheeled mobile robots: An experimental overview. In *Ramsete*, pages 181–226. Springer, 2001.
- [23] G. Oriolo, A. De Luca, and M. Vendittelli. Wmr control via dynamic feedback linearization: design, implementation, and experimental validation. *Control Systems Technology, IEEE Transactions on*, 10(6):835–852, 2002.
- [24] R. Fierro and F.L. Lewis. Control of a nonholonomic mobile robot: backstepping kinematics into dynamics. In *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, volume 4, pages 3805–3810 vol.4, 1995.
- [25] S. Nandy, S.N. Shome, G. Chakraborty, and C.S. Kumar. A modular approach to detailed dynamic formulation and control of wheeled mobile robot. In *Mechanics and Automation (ICMA), 2011 International Conference on*, pages 1471–1478, 2011.
- [26] Divya Aneesh. Tracking controller of mobile robot. In *Computing, Electronics and Electrical Technologies (ICCEET), 2012 International Conference on*, pages 343–349. IEEE, 2012.

- [27] Francesco Mondada, Edoardo Franzini, and Andre Guignard. The development of khepera. In *Proceedings of the 1st international Khepera workshop*, volume 64, pages 7–13. Citeseer, 1999.
- [28] Richard C Dorf and Robert H Bishop. *Modern control systems*. Pearson, 2011.
- [29] Richard C Dorf and Robert H Bishop. *Modern Control Systems: Solutions Manual*. Addison-Wesley, 1998.
- [30] Naresh K Sinha, Colin D Dicenzo, and Barna Szabados. Modeling of dc motors for control applications. *Industrial Electronics and Control Instrumentation, IEEE Transactions on*, (2):84–88, 1974.
- [31] Gerald Cook. *Mobile robots: navigation, control and remote sensing*. Wiley-IEEE Press, 2011.

APPENDIX A
MATLAB CODES

```

1 %% MOBILE ROBOT PLANT SETUP AND CONTROLLER DESIGN
2 %
3 % In this Document and specific MR is modeled as a 2x2 plant ,
4 % the plant is
5 % then analyzed and 2 controllers(LQR and PID) are desigend
6 % and compared
7
8 %% Plant Setup
9 % A 2x2 plant modeling two channels of DC motors connected to
10 % the wheels of
11 % a mobile robot , Inputs are voltages and outputs are angular
12 % velocity of
13 % each wheel
14
15 clear all;
16 close all;
17
18 %%Motor Specification
19 Km=0.0487;
20 kb=Km;
21 La=0.64*(10^-3);
22 Ra=0.27;
23
24 %%Robot Specification
25 r=; %wheel diameter in Meters
26 m=; %Mass in Kg
27 L=; %Axis length in Meters
28 I=m*(L^2)/6; %moment of inertia for a cube with width =
29 % length = L
30 beta=; %Surface friction
31
32 h1=tf(Km,[La Ra]);
33 h1.u='e1'; h1.y='taum1';
34
35 h2=tf(1,[(r^2)*m 0]);
36 h2.u='x1'; h2.y='vhat1';
37
38 h3=tf(L^2,[2*(r^2)*I 0]);
39 h3.u='x2'; h3.y='omegahat1';
40
41 h7=tf(beta,1);
42 h7.u='omega1'; h7.y='tauf1';
43
44 h8=tf(kb,1);
45 h8.u='omega1'; h8.y='vb1';
46 sum1= sumblk('e1=omegar1 - vb1');
47 sum2= sumblk('tau1=taum1-tauf1');

```



```

43 sum3= sumblk( 'x1=tau1+tau2' );
44 sum4= sumblk( 'x2=tau1-tau2' );
45 sum5= sumblk( 'omega1 = vhat1 + omegahat1' );
46
47
48 h4=tf(Km,[La Ra]);
49 h4.u='e2'; h4.y='taum2';
50
51 h6=tf(1,[(r^2)*m 0]);
52 h6.u='x4'; h6.y='vhat2';
53
54 h5=tf(L^2,[2*(r^2)*I 0]);
55 h5.u='x3'; h5.y='omegahat2';
56
57 h9=tf(beta,1);
58 h9.u='omega2'; h9.y='tauf2';
59
60 h10=tf(kb,1);
61 h10.u='omega2'; h10.y='vb2';
62
63 sum6= sumblk( 'e2=omegar2 - vb2' );
64 sum7= sumblk( 'tau2=taum2-tauf2' );
65 sum8= sumblk( 'x3=tau1 - tau2' );
66 sum9= sumblk( 'x4=tau2 + tau1' );
67 sum10= sumblk( 'omega2 = vhat2 - omegahat2' );
68
69 ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10,sum1,sum2,
sum3,sum4,sum5,sum6,sum7,sum8,sum9,sum10,{ 'omegar1', '
omegar2' },{ 'omega1', 'omega2' });
70 ML.statementname={ 'ia1', 'x2', 'x3', 'ia2', 'x5', 'x6' };
71
72 %Plant Plots
73
74 %StepPlot
75 f1=figure;
76 f1=stepplot(ML);
77 grid on;
78 title('Step response of the 2 Motor channels, Robot''s
dynamics included');
79
80 %Singular Value plot
81 f2=figure;
82 f2=sigmaplot(ML,{10^-2,10^4});
83 setoptions(f2,'FreqUnits','Hz');
84 grid;
85 title('Singular Values of the plant');
86

```

```

87 MLmin=minreal (ML, [], 0);
88 MLmin.u={ 'omegar1n', 'omegar2n' };
89 MLmin.y={ 'omega1n', 'omega2n' };
90
91 % StepPlot
92 f3=figure;
93 f3=stepplot (MLmin);
94 grid on;
95 title ('Step response of the 2 Motor channels , Robot''s
          dynamics included ');
96
97 % * Singular Value plot
98 f4=figure;
99 f4=sigmaplot (MLmin);
100 setoptions (f4, 'FreqUnits', 'Hz');
101 grid;
102 title ('Singular Values of the 2 Motor channels , Robot''s
          dynamics included ');
103
104 [Aol, Bol, Col, Dol]=ssdata (MLmin);
105
106
107 kff1=1/dcgain (MLmin(1,1));
108 kff2=1/dcgain (MLmin(2,2));
109
110 FFrob=MLmin*[ kff1 ,0;0 ,kff2 ];
111
112 FFrob.u={ 'omegar1n', 'omegar2n' };
113 FFrob.y={ 'omega1n', 'omega2n' };
114
115 %%StepPlot
116 ff1=figure;
117 ff1=stepplot (FFrob);
118 grid on;
119 title ('Step response of the 2 Motor channels Feed Forward
          Open Loop, Robot''s dynamics included ');
120
121 %%Singular Value plot
122 ff2=figure;
123 ff2=sigmaplot (FFrob);
124 setoptions (ff2, 'FreqUnits', 'Hz');
125 grid;
126 title ('Singular Values of the 2 Motor channels Feed Forward
          Open Loop, Robot''s dynamics included ');
127
128 %% LQR Design

```

```

129 % In this section an LQR controller is designed for the plant
      and the
130 % closed loop responses are then compared to the open loop
      propertise
131 close all;
132 clear MLaug P P1 Z Q1 R1 Klqr P2 K OL CL roblqr
133
134 %augment each chanel with 1/s
135
136 h11=tf(1,[1 0]);
137 h11.u='omega1n'; h11.y='omega1/s';
138
139 h12=tf(1,[1 0]);
140 h12.u='omega2n'; h12.y='omega2/s';
141
142 %design plant with omega/s outputs
143
144 MLaug= connect(MLmin,h11,h12,{ 'omegar1n', 'omegar2n' },{ 'omega1
      /s', 'omega2/s' });
145
146 %Klqr design
147
148 P=augstate(MLaug); %Augment states with output
149 P1=P(3:8,1:2); %2 state are the same as the output which can
      be eliminated
150
151 Z=P1.c;
152
153 Q1=Z'*Z;
154
155 R1=0.001*eye(2);
156
157 Klqr=lqr(P1,Q1,R1);
158
159 %Preparing the non-augmented system for simulation
160 P2=augstate(MLmin);
161
162 H=append(1,1,1,1,tf(1,[1 0]),tf(1,[1 0]));
163 K=Klqr*H; %putting integrator on the last two channels
164
165 FB=[0,0,1,0,0,0;
166     0,0,0,1,0,0;
167     0,0,0,0,1,0;
168     0,0,0,0,0,1;
169     1,0,0,0,0,0;
170     0,1,0,0,0,0];
171

```

```

172 OL=FB*P2*K;
173 CL=feedback(OL, eye(6), 1:6, 1:6);
174
175 roblqr=CL([5 6],[5 6]);
176 roblqr.u={'Omegar1','Omegar2'}; roblqr.y={'Omega1','Omega2'};
177
178 f5=figure;
179 f5=stepplot(roblqr);
180 title('Closed Loop Step response of a 2 channel motor/Robot
with LQR controller');
181 grid on;
182
183 w=logspace(-2,4,5000);
184
185 a=bode(roblqr(1,1),w);
186 b=bode(roblqr(1,2),w);
187 rat=b ./ a;
188 ratio=zeros(length(rat(1,1,:)),1);
189 ratio(:)=rat(1,1,:);
190
191 figure;
192 semilogx(w,20*log10(ratio));
193 hold on;
194 title('Off-Diagonal to Diagonal ratio of T_{\omega v}, LQR
Controller');
195 xlabel('frequency (rad/sec)');
196 ylabel('Magnitude (dB)');
197 grid on;
198 %
199 % figure;
200 % bodemag(roblqr);
201
202 [Alqr , Blqr , Clqr , Dlqr]=ssdata(roblqr);
203 %%
204 close all;
205
206 t=0:0.1:15;
207 Td=-0.5 *(t>5 & t<10); %- 0.5 disturbance between 5s to 10s
208 u=[ones(size(t));Td]; % augmenting step of size one and Td
to u as input
209
210 f6=figure;
211 f6=lsimplot(FFrob(1,[1 2]),roblqr(1,[1 2]),'—',u,t);
212 grid on;
213 title('Motor 1 Step response, and reaction to a input
disturbance caused by coupling of motor 2');
214 legend('OpenLoop','LQR Controller');

```

```

215
216 u1=[Td;ones(size(t))];
217 f7=figure;
218 f7=lsimplot(FFrob(2,[1 2]),roblqr(2,[1 2]),'—',u1,t);
219 grid on;
220 title('Motor 2 Step response, and reaction to a input
        disturbance caused by coupling of motor 1');
221 legend('OpenLoop','LQR Controller');
222 %ML_aug= connect(ML,h11,h12,{ 'omegar1','omegar2'},{ 'omega1','
        omega1/s','omega2','omega2/s'});
223
224 f8=figure;
225 f8=stepplot(FFrob,roblqr,'—');
226 grid on;
227 title('step response of the Openloop VS Closed Loop');
228 legend('Open Loop','LQR Controller');
229
230 f9=figure;
231 f9=sigmaplot(FFrob);
232 hold on;
233 sigmaplot(roblqr,'—');
234 grid on;
235 legend('Open Loop','LQR Controller');
236
237 f10=figure;
238 f10=lsimplot(roblqr,u,t);
239 grid on;
240 title('Motor 1 Step response, and reaction to a input
        disturbance caused by coupling of motor 2');
241 legend('LQR Controller');
242
243
244 f11=figure;
245 f11=lsimplot(roblqr,u1,t);
246 grid on;
247 title('Motor 2 Step response, and reaction to a input
        disturbance caused by coupling of motor 1');
248 legend('LQR Controller');
249
250 %% PI Controller
251 % In this section a PI controller is desigend for each
        channel
252
253 C1=pidtune(MLmin(1,1),'pi');
254 C2=pidtune(MLmin(2,2),'pi');
255 % C1.ki=1;
256 % C2.ki=1;

```

```

257 C1.kp=10;
258 C2.kp=10;
259 KPID=append(C1,C2);
260
261 FF=MLmin*KPID;
262
263 robspid=feedback(FF,eye(2));
264
265 w=logspace(-2,4,100);
266
267 a=bode(robspid(1,1),w);
268 b=bode(robspid(1,2),w);
269 rat=b ./ a;
270 ratio=zeros(length(rat(1,1,:)),1);
271 ratio(:)=rat(1,1,:);
272
273 semilogx(w,20*log10(ratio),'k—');
274
275 % stepplot(robspid);
276 %% PID vs. LQR
277 % The following plots provide a comparison between PI and LQR
      controllers
278 % for the same plant
279
280 close all;
281
282 t=0:0.1:15;
283 Td=-0.5 *(t>5 & t<10); %- 0.5 disturbance between 5s to 10s
284 u=[ones(size(t));Td]; % augmenting step of size one and Td
      to u as input
285
286 f6=figure;
287 f6=lsimplot(robspid(1,[1 2]),roblqr(1,[1 2]),'—',u,t);
288 grid on;
289 title('Motor 1 Step response, and reaction to a input
      disturbance caused by coupling of motor 2');
290 legend('PID Controller','LQR Controller');
291
292 u1=[Td;ones(size(t))];
293 f7=figure;
294 f7=lsimplot(robspid(2,[1 2]),roblqr(2,[1 2]),'—',u1,t);
295 grid on;
296 title('Motor 2 Step response, and reaction to a input
      disturbance caused by coupling of motor 1');
297 legend('PID Controller','LQR Controller');
298 %ML_aug= connect(ML,h11,h12,{ 'omegar1', 'omegar2' }, { 'omega1', '
      omega1/s', 'omega2', 'omega2/s' });

```

```

299
300 f8=figure;
301 f8=stepplot(robpid,roblqr,'—');
302 grid on;
303 title('step response of the Openloop VS Closed Loop');
304 legend('PID Controller','LQR Controller');
305
306 f9=figure;
307 f9=sigmaplot(robpid);
308 hold on;
309 sigmaplot(roblqr,'—');
310 grid on;
311 legend('PID Controller','LQR Controller');
312
313 f10=figure;
314 f10=lsimplot(roblqr,u,t);
315 grid on;
316 title('Motor 1 Step response, and reaction to a input
317       disturbance caused by coupling of motor 2');
317 legend('LQR Controller');
318
319
320 f11=figure;
321 f11=lsimplot(roblqr,u1,t);
322 grid on;
323 title('Motor 2 Step response, and reaction to a input
324       disturbance caused by coupling of motor 1');
324 legend('LQR Controller');
325
326 f10=figure;
327 f10=lsimplot(robpid,u,t);
328 grid on;
329 title('Motor 1 Step response, and reaction to a input
330       disturbance caused by coupling of motor 2');
330 legend('PID Controller');
331
332
333 f11=figure;
334 f11=lsimplot(robpid,u1,t);
335 grid on;
336 title('Motor 2 Step response, and reaction to a input
337       disturbance caused by coupling of motor 1');
337 legend('PID Controller');
338
339
340 %% Sensitivity analysis

```

```

341 % This section Sensitivity and Complement sensitivity of
      formerly designed
342 % controllers are plotted and compared
343
344 close all;
345
346 Plant=FB*P2;
347 Controller=ss(K);
348 Loopslqr=loopsens(Plant,Controller);
349
350 Loopspid=loopsens(MLmin,KPID);
351
352 figure;
353 bodemag(Loopslqr.Si,Loopspid.Si);
354 title('Sensitivity Bode Magnitude');
355 legend('LQR Controller','PID Controllers');
356 grid on;
357
358 figure;
359 f19=bodeplot(Loopslqr.Si,Loopspid.Si);
360 title('Sensitivity Bode Phase');
361 legend('LQR Controller','PID Controllers');
362 grid on;
363 setoptions(f19,'MagVisible','off');
364
365
366 figure;
367 bodemag(Loopslqr.Ti,Loopspid.Ti);
368 title('Complement Sensitivity Bode Magnitude');
369 legend('LQR Controller','PID Controller');
370 grid on;
371
372 figure;
373 f20=bodeplot(Loopslqr.Ti,Loopspid.Ti);
374 title('Complement Sensitivity Bode Phase');
375 legend('LQR Controller','PID Controllers');
376 grid on;
377 setoptions(f20,'MagVisible','off');
378
379 figure;
380 f22=bodeplot(Loopslqr.Si,Loopspid.Si);
381 title('Sensitivity Bode');
382 legend('LQR Controller','PID Controllers');
383 grid on;
384
385 figure;
386 f21=bodeplot(Loopslqr.Ti,Loopspid.Ti);

```



```

387 title('Complement Sensitivity Bode');
388 legend('LQR Controller','PID Controllers');
389 grid on;

1 %%
2 clear all;
3 close all;
4 f1=figure;
5
6 tmc= %motor torque constant;
7
8 for i=1:length(tmc)
9
10 %motor specs
11 Km=tmc(i);
12 kb=;
13 La=;
14 Ra=;
15 beta=;
16 J=;
17
18
19 h1= tf(Km,[La,Ra]);
20 h1.u='e'; h1.y='tau';
21
22 h2= tf(1,[J,beta]);
23 h2.u='tau'; h2.y='omega';
24
25 h3= tf(kb,1);
26 h3.u='omega'; h3.y='vb';
27
28 sum1=sumblk('e=v-vb');
29
30 dcm=connect(ss(h1),h2,h3,sum1,'v',{ 'tau','omega'});
31
32 %
33 t=0:1:24;
34 u=t;
35 [y,t]= lsim(dcm,u,t);
36 %
37 Power(i)=y(24,1)*y(24,2);
38 Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2); %Power in
    watts
39 Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
40 dc=Km/(beta*Ra+Km*kb);
41
42 end
43

```

```

44 plot(tmc,Power);
45 hold on;
46
47 plot(tmc,Power,'r');
48
49 %%
50 clear all;
51 close all;
52 f1=figure;
53
54 R=%armature resistance;
55
56 for i=1:length(R)
57
58 Km=;
59 kb=;
60 La=;
61 Ra=R(i);
62 beta=;
63 J=;
64
65
66 h1= tf(Km,[La,Ra]);
67 h1.u='e'; h1.y='tau';
68
69 h2= tf(1,[J,beta]);
70 h2.u='tau'; h2.y='omega';
71
72 h3= tf(kb,1);
73 h3.u='omega'; h3.y='vb';
74
75 sum1=sumblk('e=v-vb');
76
77 dcm=connect(ss(h1),h2,h3,sum1,'v',{ 'tau','omega'});
78
79
80 Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2); %Power in
      watts
81 Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
82
83 end
84 %
85 % plot(tmc,Power);
86 % hold on;
87
88 plot(R,Power,'r');
89

```

```

90
91 %% DC motor bandwidth req
92
93 clear all;
94 close all;
95 f1=figure;
96
97 lineorder={'b','g','r','c','m','k-','b—','r—','k—','b-.',
           'r-.','g—'};
98
99 tmc=[0.01 0.05 0.08 0.1 0.3 0.4 0.6 0.7 0.9 2 3 4] ;
100
101 f2=figure;
102 f3=figure;
103 f4=figure;
104 f5=figure;
105
106 for i=1:length(tmc)
107
108 Km=tmc(i);
109 kb=0.0847;
110 La=0.64*(10^-3);
111 Ra=0.27;
112 beta=0.021;
113 J=0.00057892;
114
115
116 h1= tf(Km,[La,Ra]);
117 h1.u='e'; h1.y='tau';
118
119 h2= tf(1,[J,beta]);
120 h2.u='tau'; h2.y='omega';
121
122 h3= tf(kb,1);
123 h3.u='omega'; h3.y='vb';
124
125 sum1=sumblk('e=v-vb');
126
127 dcm=connect(ss(h1),h2,h3,sum1,'v',{'tau','omega'});
128 figure(f1);
129 stepplot(dcm,lineorder{i});
130 hold on;
131
132 figure(f2);
133 bodemag(dcm,lineorder{i});
134 hold on;
135 grid on;

```

```

136
137 bw(i)=bandwidth(dcm(2,1));
138
139 Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; %Power in
    watts
140 Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
141 % dc=Km/(beta*Ra+Km*kb);
142
143 figure(f4);
144 pzmap(dcm, lineorder{i});
145 hold on;
146
147 S=stepinfo(dcm);
148 settime(i)=S(2).SettlingTime;
149
150
151 end
152 %
153 % plot(tmc,Power);
154 % hold on;
155
156 % plot(tmc,Power,'r');
157 figure(f3);
158 plot(Power,bw);
159 grid on;
160 xlabel('Power');
161 ylabel('Bandwidth');
162
163 figure(f5);
164 plot(tmc,settime);
165
166 %% DC motor + variable inertia plots
167
168 clear all;
169 close all;
170 f1=figure;
171
172 lineorder={'b','g','r','c','m','k-.','b—','r—','k—','b-.',
    'r-.','g—'};
173
174 tmc=[0.01 0.02 0.04 0.06 0.0847 0.1 0.3 0.5 0.7 0.9 1 2 3 4 5
    6];
175 inertia=(10^-5)*[10 40 60 70 80 90];
176
177 f2=figure;
178 f3=figure;
179 f4=figure;

```

```

180 for j=1:length(inertia)
181
182     for i=1:length(tmc)
183
184         Km=tmc(i);
185         kb=0.0847;
186         La=0.64*(10^-3);
187         Ra=0.27;
188         beta=0.021;
189         J=inertia(j);
190
191
192         h1= tf(Km,[La,Ra]);
193         h1.u='e'; h1.y='tau';
194
195         h2= tf(1,[J,beta]);
196         h2.u='tau'; h2.y='omega';
197
198         h3= tf(kb,1);
199         h3.u='omega'; h3.y='vb';
200
201         sum1=sumblk('e=v-vb');
202
203         dcm=connect(ss(h1),h2,h3,sum1,'v',{ 'tau','omega' });
204
205         %         figure(f2);
206         %         bodemag(dcm,lineorder{i});
207         %         hold on;
208         %         grid on;
209         %
210         bw(j,i)=bandwidth(dcm(2,1));
211
212         Power2(j,i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2); %
213         %         Power in watts
214         Power(j,i)=Power2(i)*(1.341*10^-3); %Power in hp
215     end
216 figure(f2);
217 plot(Power2(j,:),bw(j,:),lineorder{j});
218 hold on;
219 grid on;
220
221 end
222
223 %
224 % plot(tmc,Power);
225 % hold on;

```

```

226
227 % plot(tmc,Power,'r');
228 % figure(f3);
229 % plot(Power,bw);
230 % grid on;
231 % xlabel('Power');
232 % ylabel('Bandwidth');
233 %% PURE TF analysis of the motor
234 %
235 clear all;
236 close all;
237 f1=figure;
238
239 % for k=1:40
240
241 tmc=;
242 for i=1:length(tmc)
243
244 Km=tmc(i);
245
246 kb=;
247 La=;
248 Ra=;
249 beta=;
250 J=;
251
252 dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);
253
254 figure(f1);
255 pzmap(dcm);
256 hold on;
257
258 dcg(i)=dcgain(dcm);
259 end
260
261 %% PURE TF analysis of the motor
262 %
263 clear all;
264 close all;
265
266 lineorder={'b','g','r','c','m','k-','b-','r-','k-','b-','r-','g-'};
267 tmc=%%motor torque constant;
268
269 f1=figure;
270 f2=figure;
271 f3=figure;

```

```

272 f4=figure;
273 f5=figure;
274
275 % for k=1:40
276
277
278 for i=1:length(tmc)
279
280 Km=tmc(i);
281 kb=0.0847;
282 La=0.64*(10^-3);
283 Ra=0.27;
284 beta=0.021;
285 J=0.00057892;
286
287 dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);
288 %
289 figure(f1);
290 pzmap(dcm);
291 hold on;
292 %
293 figure(f2);
294 bodemag(dcm, lineorder{i});
295 hold on;
296 %
297 figure(f3);
298 step(dcm, lineorder{i});
299 hold on;
300 %
301 S=stepinfo(dcm);
302 settime(i)=S.SettlingTime;
303
304 % bw(i)=bandwidth(dcm);
305
306 vin=; %input voltage
307 Ts=(Km/Ra)*vin; %Stall Torque
308 omega0=vin/kb; %No load speed
309 powermax(i)=(Ts*omega0)/4;
310 end
311
312 figure(f4);
313 plot(tmc, powermax);
314
315 figure(f5);
316 plot(tmc, settime);
317
318

```

```

319 hold on;
320
321 %% 2 dc motors comparison
322 close all;
323 clear all;
324
325
326 lineorder={'b','g','r','c','m','k-','b—','r—','k—','b-.',
            'r-.','g—'};
327 tmc=[1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000
      12000];
328
329 f1=figure;
330 f2=figure;
331 f3=figure;
332 f4=figure;
333 f5=figure;
334
335 % for k=1:40
336
337
338 for i=1:length(tmc)
339
340 Km2=tmc(i);
341 kb2=;
342 La2=;
343 Ra2=;
344 beta2=;
345 J2=;
346
347
348 dcm=tf(Km2,[J2*La2 J2*Ra2+beta2*La2 beta2*Ra2+Km2*kb2]);
349 %
350 figure(f1);
351 pzmap(dcm);
352 hold on;
353 %
354 figure(f2);
355 bodemag(dcm, lineorder{i});
356 hold on;
357 %
358 figure(f3);
359 step(dcm, lineorder{i});
360 hold on;
361 %
362 S=stepinfo(dcm);
363 settime(i)=S.SettlingTime;

```



```

364
365 % bw(i)=bandwidth(dcm);
366
367 vin=24; %input voltage
368 Ts=(Km2/Ra2)*(vin^2); %Stall Torque
369 omega0=vin/kb2; %No load speed
370 powermax(i)=(Ts*omega0)/4;
371 Power2(i)=(24^2)*beta2*((Km2/(beta2*Ra2+Km2*kb2))^2) ; %Power
    in watts
372 end
373
374 figure(f4);
375 plot(tmc,powermax);
376 hold on;
377 plot(tmc,Power2,'r');
378
379 figure(f5);
380 plot(tmc,settime);
381
382 %% 2 dc motors comparison
383 close all;
384 clear all;
385
386 lineorder={'b','g','r','c','m','k-','b—','r—','k—','b-.',
    'r-.','g—'};
387 tmc=[0.01 0.05 0.1 0.2 0.3 0.5 1];
388
389 f1=figure;
390 f2=figure;
391 f3=figure;
392 f4=figure;
393 f5=figure;
394
395 % for k=1:40
396
397 for i=1:length(tmc)
398
399 Km2=tmc(i);
400 kb2=;
401 La2=;
402 Ra2=;
403 beta2=;
404 J2=;
405
406
407 dcm=tf(Km2,[J2*La2 J2*Ra2+beta2*La2 beta2*Ra2+Km2*kb2]);
408 %

```

```

409 figure(f1);
410 pzmap(dcm);
411 hold on;
412 %
413 figure(f2);
414 bodemag(dcm, lineorder{i});
415 hold on;
416 %
417 figure(f3);
418 step(dcm, lineorder{i});
419 hold on;
420 %
421 S=stepinfo(dcm);
422 settime(i)=S.SettlingTime;
423
424 % bw(i)=bandwidth(dcm);
425
426 vin=24; %input voltage
427 Ts=(Km2/Ra2)*(vin^2); %Stall Torque
428 omega0=vin/kb2; %No load speed
429 powermax(i)=(Ts*omega0)/4;
430 Power2(i)=(24^2)*beta2*((Km2/(beta2*Ra2+Km2*kb2))^2) ; %Power
    in watts
431 end
432
433 figure(f4);
434 plot(tmc, powermax);
435 hold on;
436 plot(tmc, Power2, 'r');
437
438 figure(f5);
439 plot(tmc, settime);

1 %% ROVER
2 % Properties of the plant
3 clear all;
4 close all;
5
6 %Motor Specification
7 Km=%torque constant;
8 kb=Km;
9 La=%armature inductance;
10 Ra=%armature resistance;
11
12 %Robot Specification
13 r=; %wheel diameter in Meters
14 m=; %Mass in Kg
15 L=; %Axis length in Meters

```

```

16 I=m*(L^2)/6; %moment of inertia for a cube with width =
    length = L
17 beta=; %Surface friction
18
19 h1= tf(Km,[La,Ra]);
20 h1.u='e'; h1.y='tau';
21
22 h2= tf(1,[J,beta]);
23 h2.u='tau'; h2.y='omega';
24
25 h3= tf(kb,1);
26 h3.u='omega'; h3.y='vb';
27
28 sum1=sumblk('e=v-vb');
29
30 dcm=connect(ss(h1),h2,h3,sum1,'v',{ 'tau','omega'});
31
32 Power2=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2); %Power in watts
33 powerrov=Power2*(1.341*10^-3); %Power in hp
34
35 ppmrov=powerrov/m;
36
37 h1=tf(Km,[La Ra]);
38 h1.u='e1'; h1.y='taul1';
39
40 h2=tf(1,[(r^2)*m 0]);
41 h2.u='x1'; h2.y='vhat1';
42
43 h3=tf(L^2,[2*(r^2)*I 0]);
44 h3.u='x2'; h3.y='omegahat1';
45
46 h7=tf(beta,1);
47 h7.u='omegal1'; h7.y='tauf1';
48
49 h8=tf(kb,1);
50 h8.u='omegal1'; h8.y='vb1';
51
52 %sumblocks in channel 1
53 sum1= sumblk('e1=omegal1 - vb1');
54 sum2= sumblk('c1=taul1-tauf1');
55 sum3= sumblk('x1=c1+tau2');
56 sum4= sumblk('x2=c1-tau2');
57 sum5= sumblk('omegal1 = vhat1 + omegahat1');
58
59
60 %Transfer functions and their input output names in channel 2
61

```

```

62 h4=tf(Km,[La Ra]);
63 h4.u='e2'; h4.y='tau2';
64
65 h6=tf(1,[(r^2)*m 0]);
66 h6.u='x4'; h6.y='vhat2';
67
68 h5=tf(L^2,[2*(r^2)*I 0]);
69 h5.u='x3'; h5.y='omegahat2';
70
71 h9=tf(beta,1);
72 h9.u='omega2'; h9.y='tauf2';
73
74 h10=tf(kb,1);
75 h10.u='omega2'; h10.y='vb2';
76
77 sum6= sumblk('e2=omegar2 - vb2');
78 sum7= sumblk('c2=tau2-tauf2');
79 sum8= sumblk('x3=tau1 - c2');
80 sum9= sumblk('x4=c2 + tau1');
81 sum10= sumblk('omega2 = vhat2 - omegahat2');
82
83 ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10,sum1,sum2,
            sum3,sum4,sum5,sum6,sum7,sum8,sum9,sum10,{ 'omegar1', '
            omegar2'},{'omegal1','omega2'});
84 ML.statename={'ia1','x2','x3','ia2','x5','x6'};
85
86 MLminrover=minreal(ML,[],0);
87 MLminrover.u={'omegar1n','omegar2n'};
88 MLminrover.y={'omegaln','omega2n'};
89
90 BWrovol=bandwidth(MLminrover(1,1)); % Motor Bandwidth
91
92 %evaluating the response at 0 rad/sec
93 mag0rov=bode(MLminrover,0);
94 magrat0rovol=mag0rov(1,1)/mag0rov(1,2);
95
96 %evaluating the response at OmegaBW
97 %
98 magbw=bode(MLmin,reqbw);
99 magratbw(k)=magbw(1,1)/magbw(1,2);
100
101 S=stepinfo(MLminrover(1,1));
102 tsrovol=S.SettlingTime;
103
104 rob=feedback(MLminrover,eye(2));
105
106 BWrovel=bandwidth(rob(1,1)); % Motor Bandwidth

```

```

107
108 %evaluating the response at 0 rad/sec
109 mag0rov=bode(rob,0);
110 magrat0rovcl=mag0rov(1,1)/mag0rov(1,2);
111
112 %evaluating the response at OmegaBW
113 %
114 magbw=bode(MLmin,reqbw);
115 magratbw(k)=magbw(1,1)/magbw(1,2);
116
117 S=stepinfo(rob(1,1));
118 tsrovcl=S.SettlingTime;
119
120 %% Open Loop , POWER/MASS Plots
121 % Properties of the plant
122 clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
    BWrovcl BWrovol powerrov ppmrov MLminrover rob;
123 close all;
124
125 mc=%motor torque constant;
126 mass=%system Mass;
127 reqbw=; %Required BW in rad/sec
128
129 lineorder={'b','g','r','c','m','k-','b-','r-','k-','b-.',
    'r-.','g-'};
130
131 %Power
132 for i=1:length(mc)
133
134 Km=mc(i);
135 kb=0.0847;
136 La=0.64*(10^-3);
137 Ra=0.27;
138 beta=0.021;
139 J=0.00057892;
140
141 dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);
142
143 Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2); %Power in
    watts
144 Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
145 end
146
147 f5=figure;
148 plot(mc,Power2);
149 title('Motor Power Vs. Torque Constant');
150 xlabel('Km (N.m/Amp)');

```

```

151 ylabel('Power (Watts)');
152 grid minor;
153
154 f6=figure;
155 f7=figure;
156 f8=figure;
157 f9=figure;
158 f10=figure;
159 f11=figure;
160 f12=figure;
161 f13=figure;
162 f14=figure;
163 f15=figure;
164
165 % rover bode
166 % bodemag(MLminrover,'k-');
167 % h=findobj(gcf,'type','line');
168 % set(h,'linewidth',1.1);
169 % hold on;
170
171
172 k=1;
173
174
175 for i=1:length(mass)
176
177     for j=1:length(mc)
178
179         Km=mc(j);
180         kb=0.0847;
181         La=0.64*(10^-3);
182         Ra=0.27;
183         r=0.1;
184         m=mass(i);
185         L=0.5;
186         I=m*(L^2)/6; %moment of inertia for a cube with width
            = length = L
187         beta=0.021;
188
189         pmr(k)=Power(j)/mass(i); %Computing Power to Mass
            ratio
190         pmr1(i,j)=Power2(j)/mass(i);
191
192         %Transfer functions and their input output names in
            chanel 1
193
194         h1=tf(Km,[La Ra]);

```

```

195     h1.u='e1'; h1.y='tau1';
196
197     h2=tf(1,[(r^2)*m 0]);
198     h2.u='x1'; h2.y='vhat1';
199
200     h3=tf(L^2,[2*(r^2)*I 0]);
201     h3.u='x2'; h3.y='omegahat1';
202
203     h7=tf(beta,1);
204     h7.u='omega1'; h7.y='tauf1';
205
206     h8=tf(kb,1);
207     h8.u='omega1'; h8.y='vb1';
208
209     %sumblocks in channel 1
210     sum1= sumblk('e1=omegar1 - vb1');
211     sum2= sumblk('c1=tau1-tauf1');
212     sum3= sumblk('x1=c1+tau2');
213     sum4= sumblk('x2=c1-tau2');
214     sum5= sumblk('omega1 = vhat1 + omegahat1');
215
216
217     %Transfer functions and their input output names in
           channel 2
218
219     h4=tf(Km,[La Ra]);
220     h4.u='e2'; h4.y='tau2';
221
222     h6=tf(1,[(r^2)*m 0]);
223     h6.u='x4'; h6.y='vhat2';
224
225     h5=tf(L^2,[2*(r^2)*I 0]);
226     h5.u='x3'; h5.y='omegahat2';
227
228     h9=tf(beta,1);
229     h9.u='omega2'; h9.y='tauf2';
230
231     h10=tf(kb,1);
232     h10.u='omega2'; h10.y='vb2';
233
234
235     %sumblocks in channel 1
236     sum6= sumblk('e2=omegar2 - vb2');
237     sum7= sumblk('c2=tau2-tauf2');
238     sum8= sumblk('x3=tau1 - c2');
239     sum9= sumblk('x4=c2 + tau1');
240     sum10= sumblk('omega2 = vhat2 - omegahat2');

```

```

241
242 %connect models
243 ML=connect( ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10,
           sum1,sum2,sum3,sum4,sum5,sum6,sum7,sum8,sum9,sum10,
           {'omegar1','omegar2'},{'omega1','omega2'});
244 ML.statename={'ia1','x2','x3','ia2','x5','x6'};
245
246 %Minimum realization Plant
247
248 MLmin=minreal(ML,[],0);
249 MLmin.u={'omegar1n','omegar2n'};
250 MLmin.y={'omega1n','omega2n'};
251
252 %3dB bandwidth
253 BW(k)=bandwidth(MLmin(1,1));
254 BW1(i,j)=bandwidth(MLmin(1,1));
255
256 %Max Transient frequency
257 [mag,phase,w]=bode(MLmin(1,2));
258 [Y,I]=max(mag);
259
260 maxoffdiagmag(i,j)=Y;
261 maxoffdiagfreq(i,j)=w(I);
262
263 %Transient Mag / DC gain
264
265 TMDC(i,j)=abs(Y/dcgain(MLmin(1,2)));
266
267 %evaluating the response at 0 rad/sec
268 mag0=bode(MLmin,0);
269 magrat0(k)=mag0(1,1)/mag0(1,2);
270
271 %evaluating the response at OmegaBW
272
273 magbw=bode(MLmin,reqbw);
274 magratbw(k)=magbw(1,1)/magbw(1,2);
275
276 S=stepinfo(MLmin(1,1));
277 ts(k)=S.SettlingTime;
278
279 k=k+1;
280
281
282 end
283
284 figure(f6);
285 plot(mc,BW1(i,:),lineorder{i});

```



```

286     hold on;
287
288     figure(f7);
289     plot(mc, maxoffdiagmag(i,:), lineorder{i});
290     hold on;
291
292     figure(f8);
293     plot(mc, TMDC(i,:), lineorder{i});
294     hold on;
295
296     figure(f9);
297     plot(mc, maxoffdiagfreq(i,:), lineorder{i});
298     hold on;
299
300     figure(f10);
301     plot(Power2, BW1(i,:), lineorder{i});
302     hold on;
303
304     figure(f11);
305     plot(Power2, TMDC(i,:), lineorder{i});
306     hold on;
307
308     figure(f12);
309     plot(Power2, maxoffdiagfreq(i,:), lineorder{i});
310     hold on;
311
312     figure(f13);
313     plot(pmr1(i,:), BW1(i,:), lineorder{i});
314     hold on;
315
316     figure(f15);
317     plot(BW1(i,:), pmr1(i,:), lineorder{i});
318     hold on;
319
320     figure(f14);
321     plot(pmr1(i,:), TMDC(i,:), lineorder{i});
322     hold on;
323
324
325
326 end
327
328
329 leg=strcat(tmc, tctr);
330
331 tmc2={' ', BW= '};
332 tctr2=num2str(BW');

```

```

333 leg2=strcat (tmc2, tctr2);
334
335 tmc3={' , Ts= '};
336 tctr3=num2str (ts ');
337 leg3=strcat (tmc3, tctr3);
338
339 legen=strcat (leg , leg2);
340
341
342 lege=strtrim ( cellstr (legen));
343
344 figure (f5);
345 legend ( 'Rover' , lege{:});
346
347 massstr={' Mass(Kg)= '}; %adding Mass= to beginning of each
    torque constant legend
348 mass1str=num2str (mass ');
349 mass2str=strcat (massstr , mass1str);
350
351 line ([0 max(pmr)] , [reqbw reqbw] , 'color' , 'r' , 'LineStyle' , '—')
    % Required Bandwidth Line
352
353 figure (f6);
354 ylabel ( '3dB Bandwidth(rad/second)' );
355 xlabel ( 'Km (N.m/Amp)' );
356 title ( '3dB Bandwidth vs torque constant' );
357 legend ( num2str (mass' ) );
358 grid minor;
359
360 figure (f7);
361 title ( 'Off Diagonal Peak bode magnitude vs torque constant' )
    ;
362 grid minor;
363 xlabel ( 'Km (N.m/Amp)' );
364 ylabel ( 'Maximum Off diagonal transient value' );
365 legend ( num2str (mass' ) );
366
367 figure (f8);
368 title ( 'Off Diagonal Transient Peak Magnitude / DC gain Vs.
    torque constant' );
369 grid minor;
370 xlabel ( 'Km (N.m/Amp)' );
371 ylabel ( 'Transient Peak Magnitude / DC gain' );
372 legend ( num2str (mass' ) );
373
374 figure (f9);

```

```

375 title('Off Diagonal Peak Transient Frequency vs Torque
        constant');
376 grid minor;
377 xlabel('Km (N.m/Amp)');
378 ylabel('Peak Transient Frequency (rad/sec)');
379 legend(num2str(mass));
380
381 figure(f10);
382 ylabel('3dB Bandwidth(rad/second)');
383 xlabel('Power (Watts)');
384 title('3dB Bandwidth vs Power');
385 legend(num2str(mass));
386 grid minor;
387
388 figure(f11);
389 title('(Off Diagonal Transient Peak Magnitude / DC gain) Vs.
        Power');
390 grid minor;
391 xlabel('Power (Watts)');
392 ylabel('Transient Peak Magnitude / DC gain');
393 legend(num2str(mass));
394
395 figure(f12);
396 title('Off Diagonal Peak Transient Frequency vs Power');
397 grid minor;
398 xlabel('Power (Watts)');
399 ylabel('Peak Transient Frequency (rad/sec)');
400 legend(num2str(mass));
401
402 figure(f13);
403 grid minor;
404 ylabel('3dB Bandwidth(rad/second)');
405 xlabel('Power/Mass(Watts/Kg)');
406 title('3dB Bandwidth vs Power/Mass');
407 legend(num2str(mass));
408
409 figure(f13);
410 ylabel('3dB Bandwidth(rad/second)');
411 xlabel('Power/Mass(Watts/Kg)');
412 title('3dB Bandwidth vs Power/Mass');
413 legend(num2str(mass));
414 grid minor;
415
416 figure(f15);
417 xlabel('3dB Bandwidth(rad/second)');
418 ylabel('Power/Mass(Watts/Kg)');
419 title('Power/Mass Vs. 3dB Bandwidth');

```

```

420 legend(num2str(mass'));
421 grid minor;
422 xlim([0 15]);
423 line([0 max(Pmr1(1,:))],[reqbw reqbw],'color','r','LineStyle',
      '—') % Required Bandwidth Line
424
425
426 figure(f14);
427 ylabel('Off Diagonal Transient Peak Magnitude / DC gain');
428 xlabel('Power/Mass(Watts/Kg)');
429 title('(Off Diagonal Transient Peak Magnitude / DC gain) Vs.
      Power/Mass');
430 legend(num2str(mass'));
431 grid minor;
432
433 %% CL(P), Variable Controller gain, Variable Power, OL VS CL
434 % Properties of the plant
435 clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
      BWrovcl BWrovol powerrov ppmrov MLminrover rob;
436 close all;
437
438 mc=%motor torque constant;
439 mass=%system Mass;
440 reqbw=; %Required BW in rad/sec
441
442 lineorder={'b','g','r','c','m','k-','b—','r—','k—','b-.',
      'r-.','g—'};
443
444 gain=%proportional gains;
445
446
447 for i=1:length(mc)
448
449 Km=mc(i);
450 kb=0.0847;
451 La=0.64*(10^-3);
452 Ra=0.27;
453 beta=0.021;
454 J=0.00057892;
455
456 dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);
457
458
459
460
461 Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2); %Power in
      watts

```

```

462 Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
463 end
464
465
466 f6=figure;
467 f7=figure;
468 f8=figure;
469
470
471     for i=1:length(gain)
472
473         C1=tf(gain(i),1);
474         C2=C1;
475
476
477         for j=1:length(mc)
478
479             Km=mc(j);
480             kb=0.0847;
481             La=0.64*(10^-3);
482             Ra=0.27;
483             r=0.1;
484             m=mass;
485             L=0.5;
486             I=m*(L^2)/6; %moment of inertia for a cube with
487                 width = length = L
488             beta=0.021;
489
490             pmr(j)=Power2(j)/m;
491
492             %Transfer functions and their input output names
493                 in chanel 1
494
495             h1=tf(Km,[La Ra]);
496             h1.u='e1'; h1.y='taul';
497
498             h2=tf(1,[(r^2)*m 0]);
499             h2.u='x1'; h2.y='vhat1';
500
501             h3=tf(L^2,[2*(r^2)*I 0]);
502             h3.u='x2'; h3.y='omegahat1';
503
504             h7=tf(beta,1);
505             h7.u='omegal'; h7.y='tauf1';
506
507             h8=tf(kb,1);

```

```

507     h8.u= 'omegar1'; h8.y='vb1';
508
509     %sumblocks in channel 1
510     sum1= sumblk('e1=omegar1 - vb1');
511     sum2= sumblk('c1=tau1-tauf1');
512     sum3= sumblk('x1=c1+tau2');
513     sum4= sumblk('x2=c1-tau2');
514     sum5= sumblk('omegar1 = vhat1 + omegahat1');
515
516
517     %Transfer functions and their input output names
           in channel 2
518
519     h4=tf(Km,[La Ra]);
520     h4.u='e2'; h4.y='tau2';
521
522     h6=tf(1,[(r^2)*m 0]);
523     h6.u='x4'; h6.y='vhat2';
524
525     h5=tf(L^2,[2*(r^2)*I 0]);
526     h5.u='x3'; h5.y='omegahat2';
527
528     h9=tf(beta,1);
529     h9.u='omegar2'; h9.y='tauf2';
530
531     h10=tf(kb,1);
532     h10.u='omegar2'; h10.y='vb2';
533
534
535     %sumblocks in channel 1
536     sum6= sumblk('e2=omegar2 - vb2');
537     sum7= sumblk('c2=tau2-tauf2');
538     sum8= sumblk('x3=tau1 - c2');
539     sum9= sumblk('x4=c2 + tau1');
540     sum10= sumblk('omegar2 = vhat2 - omegahat2');
541
542     %connect models
543     ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10
           ,sum1,sum2,sum3,sum4,sum5,sum6,sum7,sum8,sum9,
           sum10,{ 'omegar1', 'omegar2' },{ 'omegar1', 'omegar2'
           });
544     ML.statename={'ia1', 'x2', 'x3', 'ia2', 'x5', 'x6'};
545
546     MLmin=minreal(ML,[],0);
547     MLmin.u={'omegar1n', 'omegar2n'};
548     MLmin.y={'omegaln', 'omegan2n'};
549

```

```

550         Kp=append(C1,C2);
551         FF=MLmin*Kp;
552         robcl=feedback(FF,eye(2));
553
554         %3dB bandwidth
555         BWOL(j)=bandwidth(MLmin(1,1));
556         BWCL(i,j)=bandwidth(robcl(1,1));
557
558     end
559
560     figure(f6);
561     plot(mc,BWCL(i,:),lineorder{i});
562     hold on;
563
564     figure(f7);
565     plot(Power2,BWCL(i,:),lineorder{i});
566     hold on;
567 %
568     figure(f8);
569     plot(BWCL(i,:),pmr,lineorder{i});
570     hold on;
571
572
573
574 end
575
576
577 figure(f6);
578 plot(mc,BWOL(:),'b');
579 ylabel('3dB Bandwidth(rad/second)');
580 xlabel('Km');
581 title('3dB Bandwidth vs Torque constant, Variable
582         Proportional Controller');
583 grid minor;
584 tmc={'CL, kp= '};
585 leg=strcat(tmc,num2str(gain'));
586 legend(leg{:},'Open Loop');
587
588 figure(f7);
589 plot(Power2,BWOL(:),'b');
590 ylabel('3dB Bandwidth(rad/second)');
591 xlabel('Power(watts)');
592 title('3dB Bandwidth vs Power, Variable Proportional
593         Controller');
594 grid minor;
595 tmc={'CL, kp= '};
596 leg=strcat(tmc,num2str(gain'));

```

```

595     legend(leg{:}, 'Open Loop');
596
597     figure(f8);
598     plot(BWOL(:), pmr, 'b');
599     xlabel('3dB Bandwidth(rad/second)');
600     ylabel('Power/Mass(watts/Kg)');
601     title('3dB Bandwidth vs Power/Mass Ratio, Variable
        Proportional Controller');
602     grid minor;
603     tmc={'CL, kp= '};
604     leg=strcat(tmc, num2str(gain'));
605     legend(leg{:}, 'Open Loop');
606
607
608 %% CL(PI), Variable Controller gain, Variable Power, OL VS CL
609 % Properties of the plant
610 clearvars -EXCEPT tsrovc1 tsrovc1 magrat0rovc1 magrat0rovc1
        BWrovc1 BWrovc1 powerrov ppmrov MLminrovc1 rob;
611 close all;
612
613 mc=%motor torque constant;
614 mass=%system Mass;
615 reqbw=; %Required BW in rad/sec
616
617 lineorder={'b', 'g', 'r', 'c', 'm', 'k-', 'b—', 'r—', 'k—', 'b-.',
        'r-.', 'g—'};
618 pgain=%proportional gain;
619 igain=%integral gain;
620
621 for i=1:length(mc)
622
623 Km=mc(i);
624 kb=0.0847;
625 La=0.64*(10^-3);
626 Ra=0.27;
627 beta=0.021;
628 J=0.00057892;
629 dcm=tf(Km, [J*La J*Ra+beta*La beta*Ra+Km*kb]);
630 Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2); %Power in
        watts
631 Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
632 end
633
634
635 f4=figure;
636 f5=figure;
637 f6=figure;

```



```

638 f7=figure ;
639 f8=figure ;
640 f9=figure ;
641
642     for i=1:length(pgain)
643
644         C1=pid(pgain(i),0.1);
645         C2=C1;
646
647
648     for j=1:length(mc)
649
650         Km=mc(j);
651         kb=0.0847;
652         La=0.64*(10^-3);
653         Ra=0.27;
654         r=0.1;
655         m=mass;
656         L=0.5;
657         I=m*(L^2)/6; %moment of inertia for a cube with
658             width = length = L
659         beta=0.021;
660
661         pmr(j)=Power2(j)/mass;
662
663         %Transfer functions and their input output names
664             in chanel 1
665
666         h1=tf(Km,[La Ra]);
667         h1.u='e1'; h1.y='taul';
668
669         h2=tf(1,[(r^2)*m 0]);
670         h2.u='x1'; h2.y='vhat1';
671
672         h3=tf(L^2,[2*(r^2)*I 0]);
673         h3.u='x2'; h3.y='omegahat1';
674
675         h7=tf(beta,1);
676         h7.u='omegal'; h7.y='tauf1';
677
678         h8=tf(kb,1);
679         h8.u='omegal'; h8.y='vb1';
680
681         %sumblocks in channel 1
682         sum1= sumblk('e1=omegar1 - vb1');
683         sum2= sumblk('c1=taul-tauf1');
684         sum3= sumblk('x1=c1+tau2');

```

```

683 sum4= sumblk('x2=c1-tau2');
684 sum5= sumblk('omega1 = vhat1 + omegahat1');
685
686 %Transfer functions and their input output names
687     in channel 2
688
689 h4=tf(Km,[La Ra]);
690 h4.u='e2'; h4.y='tau2';
691
692 h6=tf(1,[(r^2)*m 0]);
693 h6.u='x4'; h6.y='vhat2';
694
695 h5=tf(L^2,[2*(r^2)*I 0]);
696 h5.u='x3'; h5.y='omegahat2';
697
698 h9=tf(beta,1);
699 h9.u='omega2'; h9.y='tauf2';
700
701 h10=tf(kb,1);
702 h10.u='omega2'; h10.y='vb2';
703
704
705 %sumblocks in channel 1
706 sum6= sumblk('e2=omegar2 - vb2');
707 sum7= sumblk('c2=tau2-tauf2');
708 sum8= sumblk('x3=taul - c2');
709 sum9= sumblk('x4=c2 + taul');
710 sum10= sumblk('omega2 = vhat2 - omegahat2');
711
712 %connect models
713 ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10
714     ,sum1,sum2,sum3,sum4,sum5,sum6,sum7,sum8,sum9,
715     sum10,{ 'omegar1', 'omegar2' },{ 'omega1', 'omega2'
716     });
717 ML.statename={ 'ia1', 'x2', 'x3', 'ia2', 'x5', 'x6' };
718
719 %Minimum realization Plant
720 MLmin=minreal(ML,[],0);
721 MLmin.u={ 'omegar1n', 'omegar2n' };
722 MLmin.y={ 'omegaln', 'omegan2n' };
723
724 %Minimum Realization Plots
725 %StepPlot
726 % figure(f3);

```

```

726     % f3=stepplot(MLmin);
727     % grid on;
728     % title('Step response of the 2 Motor channels ,
       Robot''s dynamics included');

729
730     %Singular Value plot
731     %     figure(f4);
732     %     sigmaplot(MLmin,sopt ,lineorder{k});
733     % setoptions(f4 , 'FreqUnits' , 'Hz');
734     %     grid;
735     %     title('Singular Values of the 2 Motor
       channels , Robot''s dynamics included');
736     %     hold all;
737     %
738     % [Aol ,Bol , Col , Dol]=ssdata(MLmin);
739
740     %     figure(f5);
741     %     bodemag(MLmin,lineorder{k});
742     %     grid on;
743     %     title('Frequency Response of the Open
       Loop System');
744     %     hold all;
745
746
747     Kp=append(C1,C2);
748     FF=MLmin*Kp;
749     robcl=feedback(FF,eye(2));
750
751     %3dB bandwidth
752     BWOL(j)=bandwidth(MLmin(1,1));
753     BWCL(i,j)=bandwidth(robcl(1,1));
754
755     end
756
757     figure(f4);
758     plot(Power2,BWCL(i,:),lineorder{i});
759     hold on;
760     %
761     figure(f5);
762     plot(BWCL(i,:),pmr,lineorder{i});
763     hold on;
764
765     figure(f6);
766     plot(mc,BWCL(i,:),lineorder{i});
767     hold on;
768     %
769

```

```

770
771
772     end
773
774         figure(f4);
775         plot(Power2,BWOL(:),'b');
776         ylabel('3dB Bandwidth(rad/second)');
777         xlabel('Power(watts)');
778         title('3dB Bandwidth vs Power, PI controller W/ Variable
              Proportional Gain');
779         grid minor;
780         tmc={'CL, kp= '};
781         leg=strcat(tmc,num2str(pgain'));
782         legend(leg{:},'Open Loop');
783
784         figure(f5);
785         plot(BWOL(:),pmr,'b');
786         xlabel('3dB Bandwidth(rad/second)');
787         ylabel('Power/Mass(watts/Kg)');
788         title('3dB Bandwidth vs Power/Mass Ratio,PI controller W/
              Variable Proportional Gain');
789         grid minor;
790         tmc={'CL, kp= '};
791         leg=strcat(tmc,num2str(pgain'));
792         legend(leg{:},'Open Loop');
793
794         figure(f6);
795         plot(mc,BWOL(:),'b');
796         ylabel('3dB Bandwidth(rad/second)');
797         xlabel('Km');
798         title('3dB Bandwidth vs Torque constant,PI controller W/
              Variable Proportional Gain');
799         grid minor;
800         tmc={'CL, kp= '};
801         leg=strcat(tmc,num2str(pgain'));
802         legend(leg{:},'Open Loop');
803
804
805     for i=1:length(igain)
806
807         C1=pid(0.1,igain(i));
808         C2=C1;
809
810
811         for j=1:length(mc)
812
813             Km=mc(j);

```

```

814 kb=0.0847;
815 La=0.64*(10^-3);
816 Ra=0.27;
817 r=0.1;
818 m=mass;
819 L=0.5;
820 I=m*(L^2)/6; %moment of inertia for a cube with
      width = length = L
821 beta=0.021;
822
823
824 %Transfer functions and their input output names
      in channel 1
825
826 h1=tf(Km,[La Ra]);
827 h1.u='e1'; h1.y='taul';
828
829 h2=tf(1,[(r^2)*m 0]);
830 h2.u='x1'; h2.y='vhat1';
831
832 h3=tf(L^2,[2*(r^2)*I 0]);
833 h3.u='x2'; h3.y='omegahat1';
834
835 h7=tf(beta,1);
836 h7.u='omegal'; h7.y='tauf1';
837
838 h8=tf(kb,1);
839 h8.u='omegal'; h8.y='vb1';
840
841 %sumblocks in channel 1
842 sum1= sumblk('e1=omegar1 - vb1');
843 sum2= sumblk('c1=taul-tauf1');
844 sum3= sumblk('x1=c1+tau2');
845 sum4= sumblk('x2=c1-tau2');
846 sum5= sumblk('omegal = vhat1 + omegahat1');
847
848
849 %Transfer functions and their input output names
      in channel 2
850
851 h4=tf(Km,[La Ra]);
852 h4.u='e2'; h4.y='tau2';
853
854 h6=tf(1,[(r^2)*m 0]);
855 h6.u='x4'; h6.y='vhat2';
856
857 h5=tf(L^2,[2*(r^2)*I 0]);

```

```

858     h5.u='x3'; h5.y='omegahat2';
859
860     h9=tf(beta,1);
861     h9.u='omega2'; h9.y='tauf2';
862
863     h10=tf(kb,1);
864     h10.u='omega2'; h10.y='vb2';
865
866
867     %sumblocks in channel 1
868     sum6= sumblk('e2=omegar2 - vb2');
869     sum7= sumblk('c2=tau2-tauf2');
870     sum8= sumblk('x3=tau1 - c2');
871     sum9= sumblk('x4=c2 + tau1');
872     sum10= sumblk('omega2 = vhat2 - omegahat2');
873
874     %connect models
875     ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10
876             ,sum1,sum2,sum3,sum4,sum5,sum6,sum7,sum8,sum9,
877             sum10,{ 'omegar1', 'omegar2' },{ 'omega1', 'omega2'
878             });
879     ML.statename={ 'ia1', 'x2', 'x3', 'ia2', 'x5', 'x6' };
880
881     MLmin=minreal(ML,[],0);
882     MLmin.u={ 'omegar1n', 'omegar2n' };
883     MLmin.y={ 'omega1n', 'omega2n' };
884
885     Kp=append(C1,C2);
886     FF=MLmin*Kp;
887     robcl=feedback(FF,eye(2));
888
889     %3dB bandwidth
890     BWOL(j)=bandwidth(MLmin(1,1));
891     BWCL(i,j)=bandwidth(robcl(1,1));
892
893     end
894
895     figure(f7);
896     plot(mc,BWCL(i,:),lineorder{i});
897     hold on;
898
899     figure(f8);
900     plot(Power2,BWCL(i,:),lineorder{i});
901     hold on;
902
903     %
904     figure(f9);
905     plot(BWCL(i,:),pmr,lineorder{i});

```

```

902         hold on;
903
904
905     end
906
907     figure(f7);
908     plot(mc,BWOL(:), 'b');
909     ylabel('3dB Bandwidth(rad/second)');
910     xlabel('Km');
911     title('3dB Bandwidth vs Torque constant,PI controller W/
          Variable Integral Gain');
912     grid minor;
913     tmc={'CL, ki= '};
914     leg=strcat(tmc,num2str(igain'));
915     legend(leg{:}, 'Open Loop');
916
917     figure(f8);
918     plot(Power2,BWOL(:), 'b');
919     ylabel('3dB Bandwidth(rad/second)');
920     xlabel('Power(watts)');
921     title('3dB Bandwidth vs Power, PI controller W/ Variable
          Integral Gain');
922     grid minor;
923     tmc={'CL, ki= '};
924     leg=strcat(tmc,num2str(pgain'));
925     legend(leg{:}, 'Open Loop');
926
927     figure(f9);
928     plot(BWOL(:),pmr, 'b');
929     xlabel('3dB Bandwidth(rad/second)');
930     ylabel('Power/Mass(watts/Kg)');
931     title('3dB Bandwidth vs Power/Mass Ratio,PI controller W/
          Variable Integral Gain');
932     grid minor;
933     tmc={'CL, ki= '};
934     leg=strcat(tmc,num2str(pgain'));
935     legend(leg{:}, 'Open Loop');
936
937
938
939     %% CL, Sensitivity (P)
940     % Properties of the plant
941
942     clearvars -EXCEPT tsrovc1 tsrovc1 magrat0rovc1 magrat0rovc1
          BWrovc1 BWrovc1 powerrovc1 ppmrovc1 MLminrovc1 rob;
943     close all;
944

```

```

945 mc=%motor torque constant;
946 mass=%system Mass;
947 reqbw=; %Required BW in rad/sec
948
949 lineorder={'b','g','r','c','m','k-','b-','r-','k-','b-','r-','g-'};
950 lineorder={'b','g','r','c','m','k-','b-','r-','k-','b-','r-','g-'};
951
952
953 f2=figure;
954 f3=figure;
955 f4=figure;
956 f5=figure;
957
958
959
960     for g=1:length(gain)
961
962         C1=tf(gain(g),1);
963         C2=tf(gain(g),1);
964
965         Km=mc;
966         kb=0.0847;
967         La=0.64*(10^-3);
968         Ra=0.27;
969         r=0.1;
970         m=mass;
971         L=0.5;
972         I=m*(L^2)/6; %moment of inertia for a cube with
           width = length = L
973         beta=0.021;
974
975
976         %Transfer functions and their input output names
           in chanel 1
977
978         h1=tf(Km,[La Ra]);
979         h1.u='e1'; h1.y='taul';
980
981         h2=tf(1,[(r^2)*m 0]);
982         h2.u='x1'; h2.y='vhat1';
983
984         h3=tf(L^2,[2*(r^2)*I 0]);
985         h3.u='x2'; h3.y='omegahat1';
986
987         h7=tf(beta,1);

```



```

988     h7.u= 'omegar1'; h7.y='tauf1';
989
990     h8=tf(kb,1);
991     h8.u= 'omegar1'; h8.y='vb1';
992
993     %sumblocks in channel 1
994     sum1= sumblk('e1=omegar1 - vb1');
995     sum2= sumblk('c1=taul-tauf1');
996     sum3= sumblk('x1=c1+tau2');
997     sum4= sumblk('x2=c1-tau2');
998     sum5= sumblk('omegar1 = vhat1 + omegahat1');
999
1000
1001     %Transfer functions and their input output names
1002     %      in channel 2
1003
1004     h4=tf(Km,[La Ra]);
1005     h4.u='e2'; h4.y='tau2';
1006
1007     h6=tf(1,[(r^2)*m 0]);
1008     h6.u='x4'; h6.y='vhat2';
1009
1010     h5=tf(L^2,[2*(r^2)*I 0]);
1011     h5.u='x3'; h5.y='omegahat2';
1012
1013     h9=tf(beta,1);
1014     h9.u= 'omega2'; h9.y='tauf2';
1015
1016     h10=tf(kb,1);
1017     h10.u= 'omega2'; h10.y='vb2';
1018
1019     %sumblocks in channel 1
1020     sum6= sumblk('e2=omegar2 - vb2');
1021     sum7= sumblk('c2=tau2-tauf2');
1022     sum8= sumblk('x3=taul - c2');
1023     sum9= sumblk('x4=c2 + tau1');
1024     sum10= sumblk('omega2 = vhat2 - omegahat2');
1025
1026     %connect models
1027     ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10
1028             ,sum1,sum2,sum3,sum4,sum5,sum6,sum7,sum8,sum9,
1029             sum10,{ 'omegar1', 'omegar2'},{ 'omegar1', 'omegar2'
1030             });
1031     ML.statename={'ia1','x2','x3','ia2','x5','x6'};
1032
1033     MLmin=minreal(ML,[],0);

```

```

1031     MLmin.u={ 'omegar1n ', 'omegar2n ' };
1032     MLmin.y={ 'omega1n ', 'omega2n ' };
1033
1034     Kp=append(C1,C2);
1035     FF=MLmin*Kp;
1036     robcl=feedback(FF, eye(2));
1037
1038     Loopspid=loopsens(FF, eye(2));
1039
1040     figure(f2);
1041     bodemag(Loopspid.Si, lineorder{g});
1042     title('Sensitivity Bode Magnitude with P
           controller, variable K1, fixed K2');
1043     grid minor;
1044     hold all;
1045
1046     figure(f3);
1047     bodemag(Loopspid.Ti, lineorder{g});
1048     title('Complement Sensitivity Bode Magnitude with
           P controller, variable K1, fixed K2');
1049     grid minor;
1050     hold all;
1051
1052     figure(f5)
1053     bodemag(robcl, lineorder{g});
1054     title('Bode magnitude of the close loops system')
1055     ;
1056     grid minor;
1057     hold all;
1058
1059     %3dB bandwidth
1060     BWCL(g)=bandwidth(robcl(1,1));
1061
1062     end
1063
1064     tmc={'Kp= '};
1065     leg=strcat(tmc, num2str(gain '));
1066
1067     figure(f2);
1068     legend(leg{:});
1069
1070     figure(f3);
1071     legend(leg{:});
1072
1073     figure(f5);
1074     legend(leg{:});

```

```

1075 figure(f4);
1076 plot(gain,BWCL);
1077 title('3dB Bandwidth Vs. Controller gain');
1078 xlabel('Controller Gain');
1079 ylabel('3dB Bandwidth');
1080
1081 %% CL, Sensitivity (P VS PI)
1082 % Properties of the plant
1083
1084 clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
        BWrovcl BWrovol powerrov ppmrov MLminrover rob;
1085 close all;
1086
1087 mc=;
1088 mass=;
1089 gain=;
1090 Km=mc;
1091 kb=;
1092 La=;
1093 Ra=;
1094 r=;
1095 m=mass;
1096 L=;
1097 I=m*(L^2)/6; %moment of inertia for a cube with width =
        length = L
1098 beta=;
1099 %Transfer functions and their input output names in chanel 1
1100
1101 h1=tf(Km,[La Ra]);
1102 h1.u='e1'; h1.y='taul';
1103
1104 h2=tf(1,[(r^2)*m 0]);
1105 h2.u='x1'; h2.y='vhat1';
1106
1107 h3=tf(L^2,[2*(r^2)*I 0]);
1108 h3.u='x2'; h3.y='omegahat1';
1109
1110 h7=tf(beta,1);
1111 h7.u='omegal'; h7.y='tauf1';
1112
1113 h8=tf(kb,1);
1114 h8.u='omegal'; h8.y='vb1';
1115
1116 %sumblocks in channel 1
1117 sum1= sumblk('e1=omegal - vb1');
1118 sum2= sumblk('c1=taul-tauf1');
1119 sum3= sumblk('x1=c1+tau2');

```

```

1120 sum4= sumblk('x2=c1-tau2');
1121 sum5= sumblk('omega1 = vhat1 + omegahat1');
1122
1123
1124 %Transfer functions and their input output names in channel 2
1125
1126 h4=tf(Km,[La Ra]);
1127 h4.u='e2'; h4.y='tau2';
1128
1129 h6=tf(1,[(r^2)*m 0]);
1130 h6.u='x4'; h6.y='vhat2';
1131
1132 h5=tf(L^2,[2*(r^2)*I 0]);
1133 h5.u='x3'; h5.y='omegahat2';
1134
1135 h9=tf(beta,1);
1136 h9.u='omega2'; h9.y='tauf2';
1137
1138 h10=tf(kb,1);
1139 h10.u='omega2'; h10.y='vb2';
1140
1141
1142 %sumblocks in channel 1
1143 sum6= sumblk('e2=omegar2 - vb2');
1144 sum7= sumblk('c2=tau2-tauf2');
1145 sum8= sumblk('x3=tau1 - c2');
1146 sum9= sumblk('x4=c2 + tau1');
1147 sum10= sumblk('omega2 = vhat2 - omegahat2');
1148
1149 %connect models
1150 ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10,sum1,sum2,
            sum3,sum4,sum5,sum6,sum7,sum8,sum9,sum10,{ 'omegar1', '
            omegar2'},{'omega1', 'omega2'});
1151 ML.statename={'ia1', 'x2', 'x3', 'ia2', 'x5', 'x6'};
1152
1153 %Minimum realization Plant
1154
1155 MLmin=minreal(ML,[],0);
1156 MLmin.u={'omegar1n', 'omegar2n'};
1157 MLmin.y={'omegaln', 'omega2n'};
1158
1159
1160 lineorder={'b', 'g', 'r', 'c', 'm', 'k'};
1161 lineorder2={'b—', 'g—', 'r—', 'c—', 'm—', 'k—'};
1162
1163
1164 f2=figure;

```

```

1165 f3=figure ;
1166 f4=figure ;
1167 f5=figure ;
1168
1169
1170
1171     for g=1:length(gain)
1172
1173         C1=tf(gain(g),1);
1174         C2=C1;
1175         CPI1=tf(gain(g),[1 0]);
1176         CPI2=CPI1;
1177
1178         Kp=append(C1,C2);
1179         FF=MLmin*Kp;
1180         robcl=feedback(FF,eye(2));
1181
1182         Loopspid=loopsens(FF,eye(2));
1183
1184         Kpi=append(CPI1,CPI2);
1185         FFpi=MLmin*Kpi;
1186         robclpi=feedback(FFpi,eye(2));
1187
1188         Loopspi=loopsens(FFpi,eye(2));
1189
1190         figure(f2);
1191         bodemag(Loopspid.Si, lineorder{g});
1192         hold all;
1193         bodemag(Loopspi.Si, lineorder2{g});
1194         title('Sensitivity Bode Magnitude with P
1195             controller, variable K1, fixed K2');
1196         grid minor;
1197         hold all;
1198
1199         figure(f3);
1200         bodemag(Loopspid.Ti, lineorder{g});
1201         hold all;
1202         bodemag(Loopspi.Ti, lineorder2{g});
1203         title('Complement Sensitivity Bode Magnitude with
1204             P controller, variable K1, fixed K2');
1205         grid minor;
1206         hold all;
1207
1208         figure(f5)
1209         bodemag(robcl, lineorder{g});
1210         hold all;
1211         bodemag(robclpi, lineorder2{g});

```

```

1210         title('Bode magnitude of the close loops system')
1211         ;
1212         grid minor;
1213         hold all;
1214
1215         %3dB bandwidth
1216         BWCL(g)=bandwidth(robcl(1,1));
1217         BWCLPi(g)=bandwidth(robclpi(1,1));
1218     end
1219
1220     tmc={'Kp= '};
1221     leg=strcat(tmc,num2str(gain'));
1222
1223     figure(f2);
1224     legend(leg{:});
1225
1226     figure(f3);
1227     legend(leg{:});
1228
1229     figure(f5);
1230     legend(leg{:});
1231
1232     figure(f4);
1233     plot(gain,BWCL);
1234     hold on;
1235     plot(gain,BWCLPi,'r');
1236     title('3dB Bandwidth Vs. Controller gain');
1237     xlabel('Controller Gain');
1238     ylabel('3dB Bandwidth');
1239     %% OPEN LOOP POWER + MASS PLOTS
1240     % Properties of the plant
1241     clearvars -EXCEPT tsrovl tsrovol magrat0rovl magrat0rovol
1242         BWrovl BWrovol powerrov ppmrov MLminrover rob;
1243     close all;
1244
1245     mc=%motor torque constant;
1246     mass=%system Mass;
1247     reqbw=; %Required BW in rad/sec
1248
1249     lineorder={'b','g','r','c','m','k-','b-','r-','k-','b-','r-','g-'};
1250     reqts=reqbw/5;
1251     reqrat=10; %Required diagonal/offdiagonal ratio
1252     %gray area calculation
1253

```

```

1254 tenpoffbw=0.9*reqbw;
1255 tenpoffrat=0.9*reqrat;
1256 tenpofffts=1.1*reqfts;
1257
1258
1259 for i=1:length(mc)
1260
1261 Km=mc(i);
1262 kb=0.0847;
1263 La=0.64*(10^-3);
1264 Ra=0.27;
1265 beta=0.021;
1266 J=0.00057892;
1267
1268
1269 h1= tf(Km,[La,Ra]);
1270 h1.u='e'; h1.y='tau';
1271
1272 h2= tf(1,[J,beta]);
1273 h2.u='tau'; h2.y='omega';
1274
1275 h3= tf(kb,1);
1276 h3.u='omega'; h3.y='vb';
1277
1278 sum1=sumblk('e=v-vb');
1279
1280 dcm=connect(ss(h1),h2,h3,sum1,'v',{ 'tau','omega'});
1281
1282 Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2); %Power in
watts
1283 Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
1284
1285 end
1286
1287 % f3=figure;
1288 % f4=figure;
1289 f5=figure;
1290 f6=figure;
1291 % f7=figure;
1292 % f8=figure;
1293 % f9=figure;
1294
1295
1296 k=1;
1297
1298
1299 for i=1:length(mass)

```

```

1300
1301     for j=1:length(mc)
1302
1303         Km=mc(j);
1304         kb=0.0847;
1305         La=0.64*(10^-3);
1306         Ra=0.27;
1307         r=0.1;
1308         m=mass(i);
1309         L=0.5;
1310         I=m*(L^2)/6; %moment of inertia for a cube with width
           = length = L
1311         beta=0.021;
1312
1313         pmr(k)=Power(j)/mass(i); %Computing Power to Mass
           ratio
1314
1315
1316         %Transfer functions and their input output names in
           chanel 1
1317
1318         h1=tf(Km,[La Ra]);
1319         h1.u='e1'; h1.y='tau1';
1320
1321         h2=tf(1,[(r^2)*m 0]);
1322         h2.u='x1'; h2.y='vhat1';
1323
1324         h3=tf(L^2,[2*(r^2)*I 0]);
1325         h3.u='x2'; h3.y='omegahat1';
1326
1327         h7=tf(beta,1);
1328         h7.u='omega1'; h7.y='tauf1';
1329
1330         h8=tf(kb,1);
1331         h8.u='omega1'; h8.y='vb1';
1332
1333         %sumblocks in channel 1
1334         sum1= sumblk('e1=omegar1 - vb1');
1335         sum2= sumblk('c1=tau1-tauf1');
1336         sum3= sumblk('x1=c1+tau2');
1337         sum4= sumblk('x2=c1-tau2');
1338         sum5= sumblk('omega1 = vhat1 + omegahat1');
1339
1340
1341         %Transfer functions and their input output names in
           channel 2
1342

```



```

1343     h4=tf(Km,[La Ra]);
1344     h4.u='e2'; h4.y='tau2';
1345
1346     h6=tf(1,[r^2]*m 0]);
1347     h6.u='x4'; h6.y='vhat2';
1348
1349     h5=tf(L^2,[2*(r^2)*I 0]);
1350     h5.u='x3'; h5.y='omegahat2';
1351
1352     h9=tf(beta,1);
1353     h9.u='omega2'; h9.y='tauf2';
1354
1355     h10=tf(kb,1);
1356     h10.u='omega2'; h10.y='vb2';
1357
1358
1359     %sumblocks in channel 1
1360     sum6= sumblk('e2=omegar2 - vb2');
1361     sum7= sumblk('c2=tau2-tauf2');
1362     sum8= sumblk('x3=tau1 - c2');
1363     sum9= sumblk('x4=c2 + tau1');
1364     sum10= sumblk('omega2 = vhat2 - omegahat2');
1365
1366     %connect models
1367     ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10,
1368             sum1,sum2,sum3,sum4,sum5,sum6,sum7,sum8,sum9,sum10,
1369             {'omegar1','omegar2'},{'omegal','omega2'});
1370     ML.staname={'ia1','x2','x3','ia2','x5','x6'};
1371
1372     %Minimum realization Plant
1373     MLmin=minreal(ML,[],0);
1374     MLmin.u={'omegar1n','omegar2n'};
1375     MLmin.y={'omegaln','omega2n'};
1376
1377     BW(i,j)=bandwidth(MLmin(1,1)); % Motor Bandwidth
1378
1379     S=stepinfo(MLmin(1,1));
1380     ts(i,j)=S.SettlingTime;
1381
1382     k=k+1;
1383
1384     end
1385
1386     figure(f5);
1387     plot(Power,ts(i,:),lineorder{i});
1388     hold on;

```

```

1388
1389     figure(f6);
1390     plot(Power,BW(i,:),lineorder{i});
1391     hold on;
1392
1393 end
1394
1395 tmc={'Mass(Kg)= '};
1396 tcstr=num2str(mass');
1397 leg=strcat(tmc,tcstr);
1398 lege=strtrim(cellstr(leg));
1399
1400 figure(f5);
1401 ylabel('Settling Time (seconds)');
1402 xlabel('Power (hp)');
1403 title('Settling time Vs. Power for Open Loop systems with
1404         different Masses');
1404 grid on;
1405
1406 line([0 max(Power)],[reqts reqts],'color','r','LineStyle','—
1407         ') % Required Bandwidth Line
1407 line([0 max(Power)],[tenpoffts tenpoffts],'color',[0.5 0.5
1408         0.5],'LineStyle','—') %10% off Bandwidth Line
1408 plot(powerrov,tsrovol,'rO','MarkerFaceColor','r') % Rover
1409         Specification
1409
1410
1411 legend(lege{:},'Minimum Design Goal','10% Off Design Goal','
1412         Rover');
1412
1413
1414 figure(f6);
1415 ylabel('Bandwidth (radian/seconds)');
1416 xlabel('Power (hp)');
1417 title('Sysem Bandwidth Vs. Power for Open Loop systems with
1418         different Masses');
1418 grid on;
1419
1420 line([0 max(Power)],[reqbw reqbw],'color','r','LineStyle','—
1421         ') % Required Bandwidth Line
1421 line([0 max(Power)],[tenpoffbw tenpoffbw],'color',[0.5 0.5
1422         0.5],'LineStyle','—') %10% off Bandwidth Line
1422 plot(powerrov,BWrovol,'rO','MarkerFaceColor','r') % Rover
1423         Specification
1423
1424

```

```

1425 legend(lege{:}, 'Minimum Design Goal', '10% Off Design Goal', '
      Rover');
1426
1427 %% Closed Loop , POWER/MASS Plots
1428
1429 % Properties of the plant
1430 clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
      BWrovcl BWrovol powerrov ppmrov MLminrover rob;
1431 close all;
1432
1433 mc=%motor torque constant;
1434 mass=%system Mass;
1435 reqbw=; %Required BW in rad/sec
1436
1437 lineorder={'b', 'g', 'r', 'c', 'm', 'k-', 'b—', 'r—', 'k—', 'b-.',
      'r-.', 'g—'};
1438
1439 % Power Calculation
1440
1441 for i=1:length(mc)
1442
1443 Km=mc(i);
1444 kb=0.0487;
1445 La=0.64*(10^-3);
1446 Ra=0.27;
1447 beta=0.021;
1448 J=0.00057892;
1449
1450
1451 h1= tf(Km, [La, Ra]);
1452 h1.u='e'; h1.y='tau';
1453
1454 h2= tf(1, [J, beta]);
1455 h2.u='tau'; h2.y='omega';
1456
1457 h3= tf(kb, 1);
1458 h3.u='omega'; h3.y='vb';
1459
1460 sum1=sumblk('e=v-vb');
1461
1462 dcm=connect(ss(h1), h2, h3, sum1, 'v', {'tau', 'omega'});
1463
1464
1465 t = 0:1:24;
1466 u=t;
1467 [y, t]= lsim(dcm, u, t);
1468

```

```

1469 Power(i)=y(24,1)*y(24,2)/746;
1470
1471 end
1472 f5=figure;
1473
1474 %rover bode
1475 bodemag(rob,'k-');
1476 h=findobj(gcf,'type','line');
1477 set(h,'linewidth',1.2);
1478 hold on;
1479
1480
1481 loops=loopsens(MLminrover,eye(2));
1482
1483 f3=figure;
1484
1485 bodemag(loops.Si,'k-');
1486 h=findobj(gcf,'type','line');
1487 set(h,'linewidth',1.2);
1488 hold on;
1489
1490 f4=figure;
1491
1492 bodemag(loops.Ti,'k-');
1493 h=findobj(gcf,'type','line');
1494 set(h,'linewidth',1.2);
1495 hold on;
1496
1497 % f6=figure;
1498 % f7=figure;
1499 % f8=figure;
1500 % f9=figure;
1501
1502
1503 k=1;
1504
1505
1506 for i=1:length(mc)
1507
1508     for j=1:length(mass)
1509
1510         Km=mc(i);
1511         kb=0.0487;
1512         La=0.64*(10^-3);
1513         Ra=0.27;
1514         r=0.1;
1515         m=mass(j);

```

```

1516 L=0.5;
1517 I=m*(L^2)/6; %moment of inertia for a cube with width
      = length = L
1518 beta=0.021;
1519
1520 pmr(k)=Power(i)/mass(j); %Computing Power to Mass
      ratio
1521
1522
1523 %Transfer functions and their input output names in
      chanel 1
1524
1525 h1=tf(Km,[La Ra]);
1526 h1.u='e1'; h1.y='tau1';
1527
1528 h2=tf(1,[(r^2)*m 0]);
1529 h2.u='x1'; h2.y='vhat1';
1530
1531 h3=tf(L^2,[2*(r^2)*I 0]);
1532 h3.u='x2'; h3.y='omegahat1';
1533
1534 h7=tf(beta,1);
1535 h7.u='omegal'; h7.y='tauf1';
1536
1537 h8=tf(kb,1);
1538 h8.u='omegal'; h8.y='vb1';
1539
1540 %sumblocks in channel 1
1541 sum1= sumblk('e1=omegar1 - vb1');
1542 sum2= sumblk('c1=tau1-tauf1');
1543 sum3= sumblk('x1=c1+tau2');
1544 sum4= sumblk('x2=c1-tau2');
1545 sum5= sumblk('omegal = vhat1 + omegahat1');
1546
1547
1548 %Transfer functions and their input output names in
      channel 2
1549
1550 h4=tf(Km,[La Ra]);
1551 h4.u='e2'; h4.y='tau2';
1552
1553 h6=tf(1,[(r^2)*m 0]);
1554 h6.u='x4'; h6.y='vhat2';
1555
1556 h5=tf(L^2,[2*(r^2)*I 0]);
1557 h5.u='x3'; h5.y='omegahat2';
1558

```

```

1559     h9=tf(beta,1);
1560     h9.u= 'omega2'; h9.y='tauf2';
1561
1562     h10=tf(kb,1);
1563     h10.u= 'omega2'; h10.y='vb2';
1564
1565
1566     %sumblocks in channel 1
1567     sum6= sumblk('e2=omegar2 - vb2');
1568     sum7= sumblk('c2=tau2-tauf2');
1569     sum8= sumblk('x3=tau1 - c2');
1570     sum9= sumblk('x4=c2 + tau1');
1571     sum10= sumblk('omega2 = vhat2 - omegahat2');
1572
1573     %connect models
1574     ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10,
1575             sum1,sum2,sum3,sum4,sum5,sum6,sum7,sum8,sum9,sum10,
1576             {'omegar1','omegar2'},{'omegal','omega2'});
1577     ML.staname={'ia1','x2','x3','ia2','x5','x6'};
1578
1579     %Minimum realization Plant
1580     MLmin=minreal(ML,[],0);
1581     MLmin.u={'omegar1n','omegar2n'};
1582     MLmin.y={'omegaln','omegan'};
1583
1584     %Minimum Realization Plots
1585
1586     %StepPlot
1587     % figure(f3);
1588     % f3=stepplot(MLmin);
1589     % grid on;
1590     % title('Step response of the 2 Motor channels , Robot
1591           ''s dynamics included');
1592
1593     %Singular Value plot
1594     % figure(f4);
1595     % sigmaplot(MLmin,sopt,lineorder{k});
1596     % setoptions(f4,'FreqUnits','Hz');
1597     % grid;
1598     % title('Singular Values of the 2 Motor channels ,
1599           Robot''s dynamics included');
1600     % hold all;
1601
1602     robcl=feedback(MLmin,eye(2));
1603
1604     figure(f5);

```

```

1602     bodemag(robcl , lineorder {k});
1603     grid on;
1604     title('Frequency Response of the Closed Loop System')
1605         ;
1605     hold all;
1606
1606     BW(k)=bandwidth(robcl(1,1)); % Motor Bandwidth
1607
1608     %evaluating the response at 0 rad/sec
1609     mag0=bode(robcl,0);
1610     magrat0(k)=mag0(1,1)/mag0(1,2);
1611
1612     %evaluating the response at OmegaBW
1613
1614
1615     magbw=bode(robcl , reqbw);
1616     magratbw(k)=magbw(1,1)/magbw(1,2);
1617
1618     S=stepinfo(robcl(1,1));
1619     ts(k)=S.SettlingTime;
1620
1621     %Sensetivity plots
1622     loops=loopsens(MLmin, eye(2));
1623
1624     figure(f3);
1625     bodemag(loops.Si, lineorder {k});
1626     title('Sensetivity Magnitude Closed loop System with
1627         K=I, Variable Power/Mass');
1628     grid on;
1629     hold all;
1630
1631     figure(f4);
1632     bodemag(loops.Ti, lineorder {k});
1633     title('Complement Magnitude Closed loop System with
1634         no K=I, Variable Power/Mass');
1635     grid on;
1636     hold all;
1637
1638     k=k+1;
1639
1640     end
1641 end
1642 tmc={'Power/Mass(hp/Kg) = '}; %adding Mass= to begining of
1643     each torque constant legend
1644 t cstr=num2str(pmr');

```

```

1645 leg=strcat(tmc,tcstr);
1646
1647 tmc2={' , BW= '};
1648 tcstr2=num2str(BW);
1649 leg2=strcat(tmc2,tcstr2);
1650
1651 % tmc3={' , Ts= '};
1652 % tcstr3=num2str(ts);
1653 % leg3=strcat(tmc3,tcstr3);
1654
1655 legen=strcat(leg,leg2);
1656
1657
1658 lege=strtrim(cellstr(legen));
1659
1660
1661 figure(f3);
1662 legend('Rover',lege{:});
1663
1664 figure(f4);
1665 legend('Rover',lege{:});
1666
1667 figure(f5);
1668 legend('Rover',lege{:});
1669
1670 figure(f6);
1671 plot(pmr,ts);
1672 ylabel('Settling Time (Seconds)');
1673 xlabel('Power per Kg (hp/Kg)');
1674 title('Settling time vs Power to Mass ratio plot');
1675
1676 figure(f7);
1677 plot(pmr,BW);
1678 ylabel('System Bandwidth (rad/sec)');
1679 xlabel('Power per Kg (hp/Kg)');
1680 title('Badnwidth vs Power to Mass ratio plot');
1681
1682 figure(f8);
1683 plot(pmr,magrat0);
1684 ylabel('diagonal DC gain / off diagonal DC gain');
1685 xlabel('Power per Kg (hp/Kg)');
1686 title('diagonal to off diagonal dc gain ratio vs Power to
      Mass ratio plot');
1687
1688 figure(f9);
1689 plot(pmr,magratbw);
1690 ylabel('diagonal amplitude / off diagonal amplitude');

```



```

1691 xlabel('Power per Kg (hp/Kg)');
1692 title('diagonal to off diagonal amplitude ratio @ bandwidth
        frequency vs Power to Mass ratio plot');
1693
1694 %% OL VS CL
1695 % Properties of the plant
1696 clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
        BWrovcl BWrovol powerrov ppmrov;
1697 close all;
1698
1699 mc=%motor torque constant;
1700 mass=%system Mass;
1701 reqbw=; %Required BW in rad/sec
1702
1703 lineorder={'b','g','r','c','m','k-','b—','r—','k—','b-.',
        'r-.','g—'};
1704 reqts=reqbw/5;
1705
1706 reqrat=10; %Required diagonal/offdiagonal ratio
1707
1708 %gray area calculation
1709
1710 tenpoffbw=0.9*reqbw;
1711 tenpoffrat=0.9*reqrat;
1712 tenpoffts=1.1*reqts;
1713
1714
1715 lineorder={'b','g','r','c','m','y','k','b—','r—','k—','b-.',
        ',','r-.','g—'};
1716
1717 Kminit=0.0487;
1718
1719 % Power Calculation @ 24 V
1720
1721 for i=1:length(mc)
1722
1723 Km=mc(i);
1724 kb=0.0847;
1725 La=0.64*(10^-3);
1726 Ra=0.27;
1727 beta=0.021;
1728 J=0.00057892;
1729
1730 dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);
1731
1732 vin=24; %input voltage
1733 Ts=(Km/Ra)*vin; %Stall Torque

```

```

1734 omega0=vin/kb; %No load speed
1735 Power2(i)=(Ts*omega0)/4;
1736 Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
1737
1738 end
1739
1740
1741 f6=figure;
1742 f7=figure;
1743 f8=figure;
1744 f9=figure;
1745
1746
1747 k=1;
1748
1749
1750 for i=1:length(mc)
1751     for j=1:length(mass)
1752         Km=mc(i);
1753         kb=0.0847;
1754         La=0.64*(10^-3);
1755         Ra=0.27;
1756         r=0.1;
1757         m=mass(j);
1758         L=0.5;
1759         I=m*(L^2)/6; %moment of inertia for a cube with width
1760             = length = L
1761         beta=0.021;
1762
1763         pmr(k)=Power(i)/mass(j); %Computing Power to Mass
1764             ratio
1765
1766
1767         %Transfer functions and their input output names in
1768             chanel 1
1769
1770         h1=tf(Km,[La Ra]);
1771         h1.u='e1'; h1.y='tau1';
1772
1773         h2=tf(1,[(r^2)*m 0]);
1774         h2.u='x1'; h2.y='vhat1';
1775
1776         h3=tf(L^2,[2*(r^2)*I 0]);
1777         h3.u='x2'; h3.y='omegahat1';

```

```

1778     h7=tf(beta,1);
1779     h7.u='omegar1'; h7.y='tauf1';
1780
1781     h8=tf(kb,1);
1782     h8.u='omegar1'; h8.y='vb1';
1783
1784     %sumblocks in channel 1
1785     sum1= sumblk('e1=omegar1 - vb1');
1786     sum2= sumblk('c1=tau1-tauf1');
1787     sum3= sumblk('x1=c1+tau2');
1788     sum4= sumblk('x2=c1-tau2');
1789     sum5= sumblk('omegar1 = vhat1 + omegahat1');
1790
1791
1792     %Transfer functions and their input output names in
1793         channel 2
1794
1795     h4=tf(Km,[La Ra]);
1796     h4.u='e2'; h4.y='tau2';
1797
1798     h6=tf(1,[(r^2)*m 0]);
1799     h6.u='x4'; h6.y='vhat2';
1800
1801     h5=tf(L^2,[2*(r^2)*I 0]);
1802     h5.u='x3'; h5.y='omegahat2';
1803
1804     h9=tf(beta,1);
1805     h9.u='omegar2'; h9.y='tauf2';
1806
1807     h10=tf(kb,1);
1808     h10.u='omegar2'; h10.y='vb2';
1809
1810     %sumblocks in channel 1
1811     sum6= sumblk('e2=omegar2 - vb2');
1812     sum7= sumblk('c2=tau2-tauf2');
1813     sum8= sumblk('x3=tau1 - c2');
1814     sum9= sumblk('x4=c2 + tau1');
1815     sum10= sumblk('omegar2 = vhat2 - omegahat2');
1816
1817     %connect models
1818     ML=connect(ss(h1),h2,h3,ss(h4),h5,h6,h7,h8,h9,h10,
1819         sum1,sum2,sum3,sum4,sum5,sum6,sum7,sum8,sum9,sum10,
1820         {'omegar1','omegar2'},{'omegar1','omegar2'});
1821     ML.staname={'ia1','x2','x3','ia2','x5','x6'};
1822
1823     %Minimum realization Plant

```

```

1822
1823     MLmin=minreal(ML,[],0);
1824     MLmin.u={'omegar1n','omegar2n'};
1825     MLmin.y={'omega1n','omega2n'};
1826
1827     robcl=feedback(MLmin,eye(2));
1828
1829     BWol(k)=bandwidth(MLmin(1,1)); % System Bandwidth
1830
1831     BWcl(k)=bandwidth(robcl(1,1)); % System Bandwidth
1832
1833     %evaluating the response at 0 rad/sec
1834     mag0ol=bode(MLmin,0);
1835     magrat0ol(k)=mag0ol(1,1)/mag0ol(1,2);
1836
1837     mag0cl=bode(robcl,0);
1838     magrat0cl(k)=mag0cl(1,1)/mag0cl(1,2);
1839
1840     %evaluating the response at OmegaBW
1841     magbwol=bode(MLmin,reqbw);
1842     magratbwol(k)=magbwol(1,1)/magbwol(1,2);
1843
1844     magbwcl=bode(robcl,reqbw);
1845     magratbwcl(k)=magbwcl(1,1)/magbwcl(1,2);
1846
1847     Sol=stepinfo(MLmin(1,1));
1848     tsol(k)=Sol.SettlingTime;
1849
1850     Scl=stepinfo(robcl(1,1));
1851     tscl(k)=Scl.SettlingTime;
1852
1853
1854     k=k+1;
1855
1856
1857     end
1858 end
1859
1860
1861
1862 figure(f6);
1863 plot(pmr,tsol);
1864 ylabel('Settling Time (Seconds)');
1865 xlabel('Power per Kg (hp/Kg)');
1866 title('Settling time vs Power to Mass ratio plot');
1867 hold on;
1868 plot(pmr,tscl,'g');

```

```

1869
1870 grid on;
1871
1872 %Settling Time
1873 pmtscl=interp1 (tscl ,pmr ,reqts );
1874 pmtsol=interp1 (tsol ,pmr ,reqts );
1875
1876 pmtscl2=interp1 (tscl ,pmr ,tenpoffts );
1877 pmtsol2=interp1 (tsol ,pmr ,tenpoffts );
1878
1879 line ([0 max(pmr)],[reqts reqts], 'color','r','LineStyle','—')
      % Required Bandwidth Line
1880
1881 line ([0 max(pmr)],[tenpoffts tenpoffts], 'color',[0.5 0.5
      0.5], 'LineStyle','—') %10% off Bandwidth Line
1882
1883 plot (ppmrov,tsrovcl,'rO','MarkerFaceColor','r') % Rover
      Specification
1884
1885 plot (ppmrov,tsrovol,'rO','MarkerFaceColor','r') % Rover
      Specification
1886
1887
1888 if ~isnan (pmtsol)
1889 line ([pmtsol pmtsol],[0 reqts], 'color','r','LineStyle','—');
1890 end
1891
1892 if ~isnan (pmtscl)
1893 line ([pmtscl pmtscl],[0 reqts], 'color','r','LineStyle','—');
1894 end
1895
1896 if ~isnan (pmtsol2)
1897 line ([pmtsol2 pmtsol2],[0 tenpoffts], 'color',[0.5 0.5 0.5], '
      LineStyle','—');
1898 end
1899
1900 if ~isnan (pmtscl2)
1901 line ([pmtscl2 pmtscl2],[0 tenpoffts], 'color',[0.5 0.5 0.5], '
      LineStyle','—');
1902 end
1903
1904 legend ('Open Loop System','Closed Loop System','Minimum
      Design Goal','10% Off Design Goal','Rover');
1905
1906
1907 figure (f7);
1908 plot (pmr,BWol);

```

```

1909 ylabel('System Bandwidth (rad/sec)');
1910 xlabel('Power per Kg (hp/Kg)');
1911 title('Badnwidth vs Power to Mass ratio plot');
1912 hold on;
1913 plot(pmr,BWcl,'g');
1914 grid on;
1915
1916 %Bandwidth
1917 pmcl=interp1(BWcl,pmr,reqbw);
1918 pmol=interp1(BWol,pmr,reqbw);
1919
1920 pmcl2=interp1(BWcl,pmr,tenpoffbw);
1921 pmol2=interp1(BWol,pmr,tenpoffbw);
1922
1923 line([0 max(pmr)],[reqbw reqbw],'color','r','LineStyle','—')
1924     % Required Bandwidth Line
1925 line([0 max(pmr)],[tenpoffbw tenpoffbw],'color',[0.5 0.5
1926     0.5],'LineStyle','—') %10% off Bandwidth Line
1927
1928 plot(ppmrov,BWrovcl,'rO','MarkerFaceColor','r') % Rover
1929     Specification
1930 plot(ppmrov,BWrovol,'rO','MarkerFaceColor','r') % Rover
1931     Specification
1932
1933 if ~isnan(pmol)
1934     line([pmol pmol],[0 reqbw],'color','r','LineStyle','—');
1935 end
1936
1937 if ~isnan(pmcl)
1938     line([pmcl pmcl],[0 reqbw],'color','r','LineStyle','—');
1939 end
1940
1941 if ~isnan(pmol2)
1942     line([pmol2 pmol2],[0 tenpoffbw],'color',[0.5 0.5 0.5],
1943         'LineStyle','—');
1944 end
1945
1946 if ~isnan(pmcl2)
1947     line([pmcl2 pmcl2],[0 tenpoffbw],'color',[0.5 0.5 0.5],
1948         'LineStyle','—');
1949 end
1950
1951 legend('Open Loop System','Closed Loop System','Minimum
1952     Design Goal','10% Off Design Goal','Rover');
1953
1954 %Diagonal/Off Diagonal

```

```

1949
1950 figure(f8);
1951 plot(pmr, magrat0ol);
1952 ylabel('diagonal DC gain / off diagonal DC gain');
1953 xlabel('Power per Kg (hp/Kg)');
1954 title('diagonal to off diagonal dc gain ratio vs Power to
        Mass ratio plot');
1955 hold on;
1956 plot(pmr, magrat0cl, 'g');
1957 grid on;
1958
1959 %interpolate data
1960 pmratcl=interp1(magrat0cl, pmr, reqrat);
1961 pmratol=interp1(magrat0ol, pmr, reqrat);
1962
1963 pmratcl2=interp1(magrat0cl, pmr, tenpoffrat);
1964 pmratol2=interp1(magrat0ol, pmr, tenpoffrat);
1965
1966 line([0 max(pmr)], [reqrat reqrat], 'color', 'r', 'LineStyle', '—
        ') % Required Bandwidth Line
1967 line([0 max(pmr)], [tenpoffrat tenpoffrat], 'color', [0.5 0.5
        0.5], 'LineStyle', '—') % 10% off Bandwidth Line
1968
1969 plot(ppmrov, magrat0rovol, 'rO', 'MarkerFaceColor', 'r') % Rover
        Specification
1970
1971 plot(ppmrov, magrat0rovcl, 'rO', 'MarkerFaceColor', 'r') % Rover
        Specification
1972
1973 if ~isnan(pmratol)
1974 line([pmratol pmratol], [0 reqrat], 'color', 'r', 'LineStyle', '—
        ');
1975 end
1976
1977 if ~isnan(pmratcl)
1978 line([pmratcl pmratcl], [0 reqrat], 'color', 'r', 'LineStyle', '—
        ');
1979 end
1980
1981 if ~isnan(pmratol2)
1982 line([pmratol2 pmratol2], [0 tenpoffrat], 'color', [0.5 0.5
        0.5], 'LineStyle', '—');
1983 end
1984
1985 if ~isnan(pmratcl2)
1986 line([pmratcl2 pmratcl2], [0 tenpoffrat], 'color', [0.5 0.5
        0.5], 'LineStyle', '—');

```

```

1987 end
1988 legend('Open Loop System','Closed Loop System','Minimum
          Design Goal','10% Off Design Goal','Rover');
1989
1990
1991 figure(f9);
1992 plot(pmr,magratbwol);
1993 ylabel('diagonal amplitude / off diagonal amplitude');
1994 xlabel('Power per Kg (hp/Kg)');
1995 title('diagonal to off diagonal amplitude ratio @ bandwidth
          frequency vs Power to Mass ratio plot');
1996 hold on;
1997 plot(pmr,magratbwcl,'g');
1998 legend('Open Loop System','Closed Loop System');
1999
2000 %interpolate data
2001 pmratbwcl=interp1(magratbwcl,pmr,reqrat,'pchip');
2002 pmratbwol=interp1(magratbwol,pmr,reqrat,'pchip');
2003
2004 line([0 max(pmr)],[reqrat reqrat],'color','r','LineStyle','—
          ') % Required Bandwidth Line
2005 line([pmratbwol pmratbwol],[0 reqrat],'color','r','LineStyle'
          , '—');
2006 line([pmratbwcl pmratbwcl],[0 reqrat],'color','r','LineStyle'
          , '—');

```