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-á-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

LIST OF TABLES

Table | Page

LIST OF FIGURES

Figure Page

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 systens.

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 toque 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 Centeralized 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*

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

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

$$h_i(q) = m_i \qquad i = 1, ..., k < n$$

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 $q = [x \quad y \quad \theta]^T$, assuming the disk can only roll on the touching plane without slipping to the sides (i.e. there is no velocity

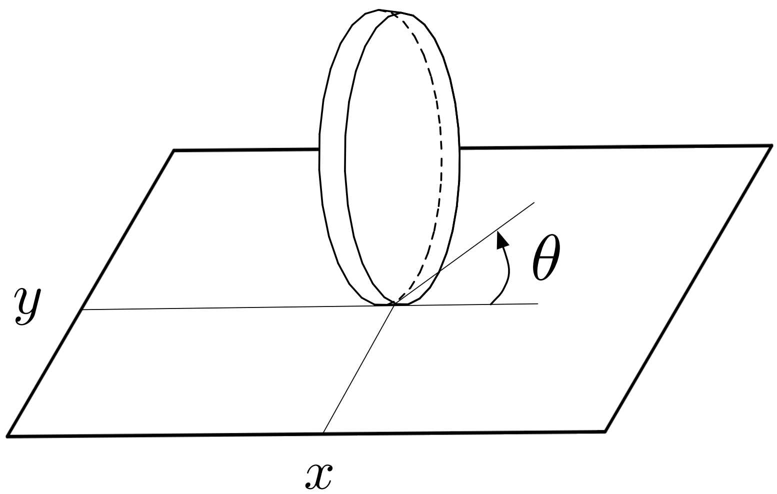


Figure 2.1: Pure rolling disk and its generalized coordinates in 2D plane

component for the contact point perpendicular to the plane containing the disk). This can be defined as:

$$\dot{x}\sin\theta - \dot{y}\cos\theta = 0 \tag{2.2}$$

Rewriting Equation 2.2 in *pfaffian form* will result in

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

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 $[b_1(q)...b_{n-k}(q)]$ a *kinematic model* of the constrained mechanical system is given by:

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

where $\mathbf{u} = [u_1 ... u_{n-k}]^T \in \mathbb{R}^{n-k}$ 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.*

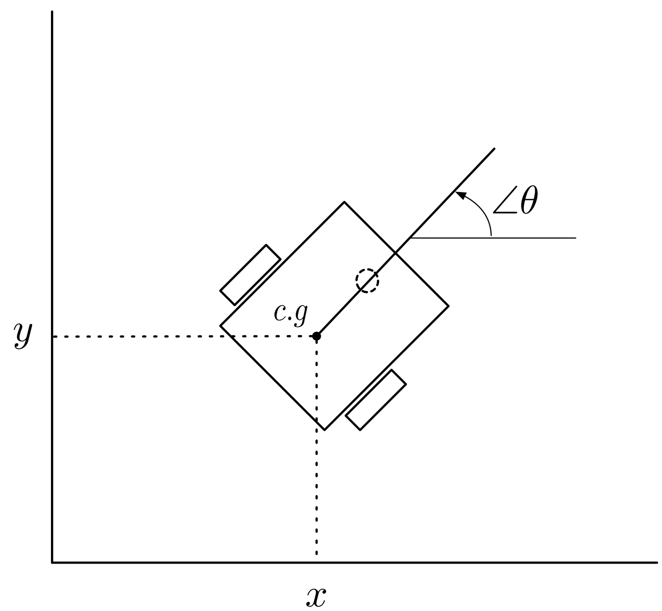


Figure 2.3: Generalized coordinates for a mobile robot

Consider the mobile robot in Figure 2.3. Using generalized coordinate vector $q = [x \quad y \quad \theta]$ 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

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

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

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

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

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

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:

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

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 kinamtic 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

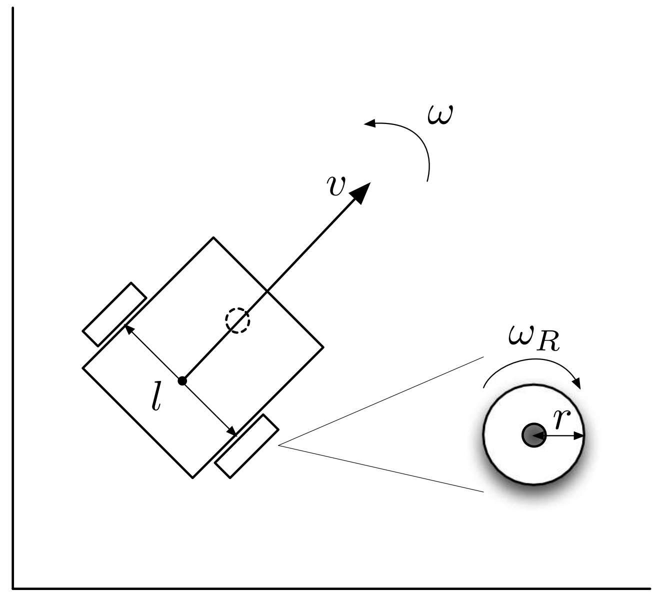


Figure 2.4: Linear and Angular velocity of the robot

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

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

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

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

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

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

where, $S(q)$ is a $(n \quad by \quad m)$ 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

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

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

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

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.

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

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

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

$$v = \sqrt{\dot{x}^2 + \dot{y}^2} \tag{2.17}$$

$$\omega = \dot{\theta} \tag{2.18}$$

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.

$$\dot{x} = v\cos\theta \tag{2.19}$$

$$\dot{y} = v\sin\theta \tag{2.20}$$

$$\dot{\theta} = \omega \tag{2.21}$$

$$\dot{v} = \frac{\tau_l}{m} \tag{2.22}$$

$$\dot{\omega} = \frac{\tau_a}{I} \tag{2.23}$$

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:

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

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: *Filed-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].

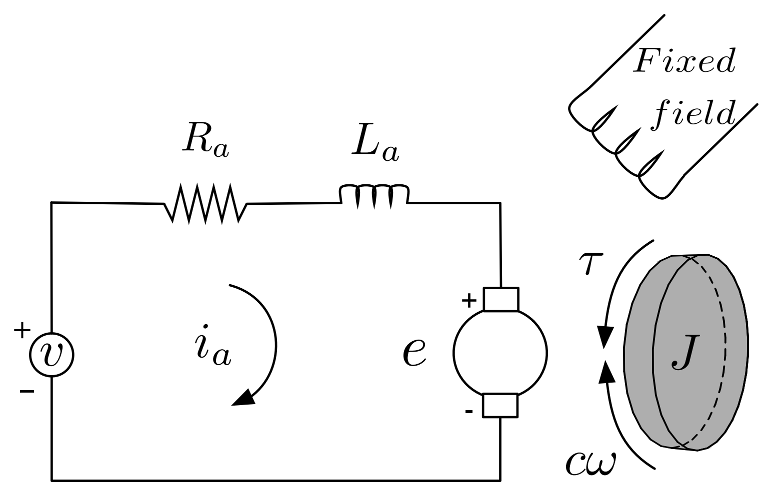


Figure 2.5: Circuit equivalent of a DC motor with a free body attached

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

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

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

$$v_a = v_r + v_L + v_b \tag{2.26}$$

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.

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

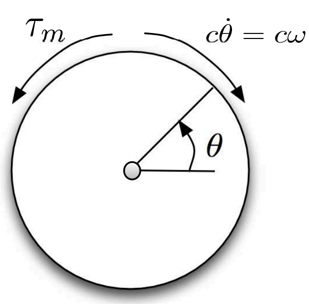


Figure 2.6: Torque applied to a free body

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

$$J\dot{\omega} + c\omega - \tau_m \tag{2.28}$$

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

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

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

$$\frac{\omega(s)}{V_a(s)} = \frac{K_m}{(L_a.s + R_a)(Js + c) + K_b K_m} \tag{2.30}$$

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 $\omega(s)$.

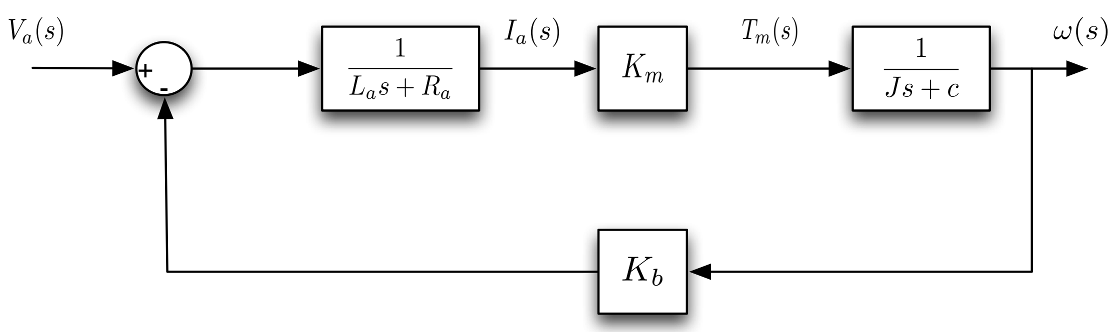


Figure 2.7: DC Motor block diagram

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.

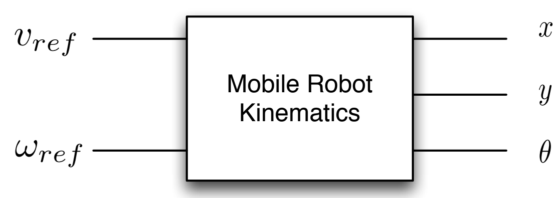


Figure 2.8: Block diagram of a mobile robot's kinematic

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:

$$
\begin{bmatrix} v \\ \omega \end{bmatrix} = M \begin{bmatrix} \omega_R \\ \omega_L \end{bmatrix} \tag{2.31}
$$

where M is a transformation matrix defined as:

$$
M = \begin{bmatrix} \frac{r}{2}, \frac{r}{2} \\ \frac{r}{l}, \frac{-r}{l} \end{bmatrix} \tag{2.32}
$$

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

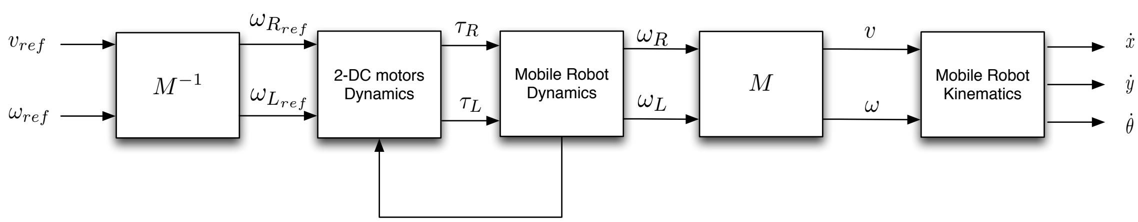


Figure 2.9: Block diagram of a mobile robot including actuator and body dynamics

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.

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

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

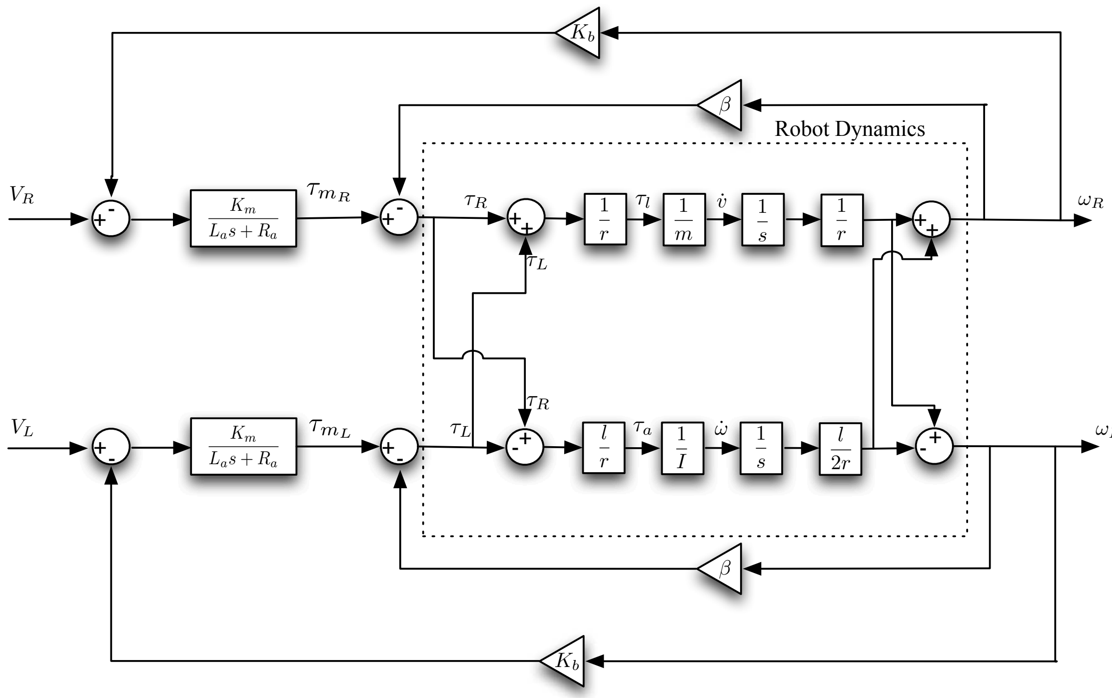


Figure 2.10: Actuator and body dynamics block diagram from $\omega_{R_{ref}}$ & $\omega_{L_{ref}}$ to ω_R & ω_L

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:

$$\begin{bmatrix} \dot{v} \\ \dot{\omega} \end{bmatrix} = J_{m2\times2}T_{2\times2}K_{m2\times2}i - J_{m2\times2}T_{2\times2}\beta_{2\times2}\omega_w \tag{2.34}$$

$$\dot{i} = -L_{2\times2}R_{2\times2}i - L_{2\times2}K_{2\times2}\omega_w + L_{2\times2}V \tag{2.35}$$

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

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

$$R = \begin{bmatrix} R_a & 0 \\ 0 & R_a \end{bmatrix} \tag{2.37}$$

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

$$K = \begin{bmatrix} K_b & 0 \\ 0 & K_b \end{bmatrix} \tag{2.39}$$

$$\beta = \begin{bmatrix} \beta & 0 \\ 0 & \beta \end{bmatrix} \tag{2.40}$$

$$L = \begin{bmatrix} \frac{1}{L_a} & 0 \\ 0 & \frac{1}{L_a} \end{bmatrix} \tag{2.41}$$

$$i = \begin{bmatrix} i_{a1} \\ i_{a2} \end{bmatrix} \tag{2.42}$$

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

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

where,

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

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

Gains, poles and zeros are approximately located at

$$a = \frac{K_m}{JL_a} \tag{2.46}$$

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

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

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

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

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

$$J = \frac{J_1 J_2}{J_1 + J_2} = \frac{2Imr^2}{2I + l^2 m} \tag{2.51}$$

$$J_1 = mr^2 \tag{2.52}$$

$$J_2 = \frac{2r^2 I}{l^2} \tag{2.53}$$

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 Figure2.12, 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	$0.0487 \quad N.m/Amp$
L_a	Armature Inductance	$0.64 \times 10^{-3} \quad H$
R_a	Armature Resistance	$0.27 \quad ohm$
r	Wheel Raduis	$0.1 \quad m$
m	Mass	$30 \quad Kg$
I	Interia	$0.83 \quad Kg.m^2$
l	Distance between the wheels	$0.5 \quad m$
β	Friction Constant	$0.021 \quad N.m.s$
K_b	Back EMF Constant	$0.0487 \quad V/(rad/sec)$

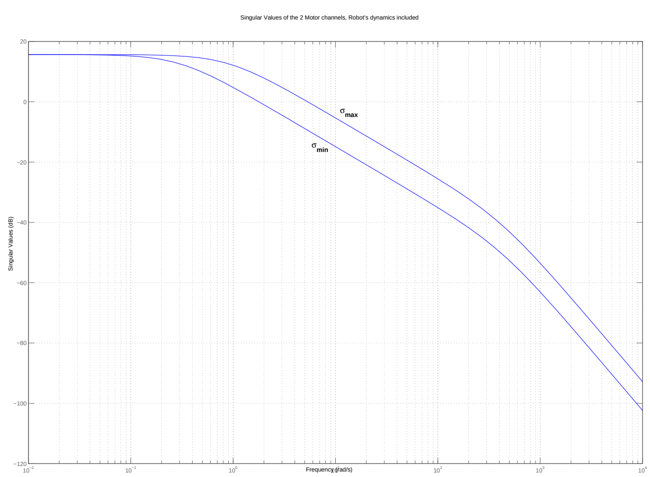


Figure 2.11: Singular Value plot of Mobile Robot Dynamics

previous discussions, we need this system to look close to I_{2x2}. This means there are two important factors to consider:

- *Bandwidth*

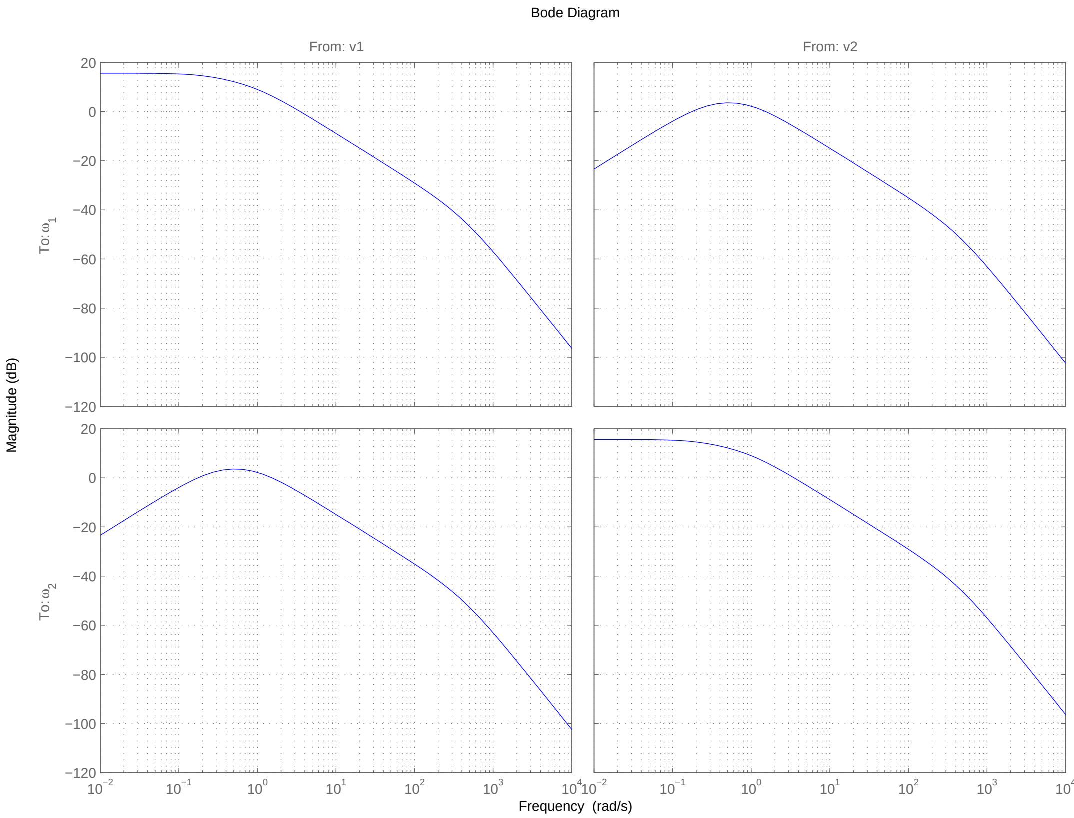


Figure 2.12: Bode Magnitude Plot of Mobile Robot Dynamics

- *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:

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

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.

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

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

$$P(0) = \tau(0)\omega(0) \tag{2.56}$$

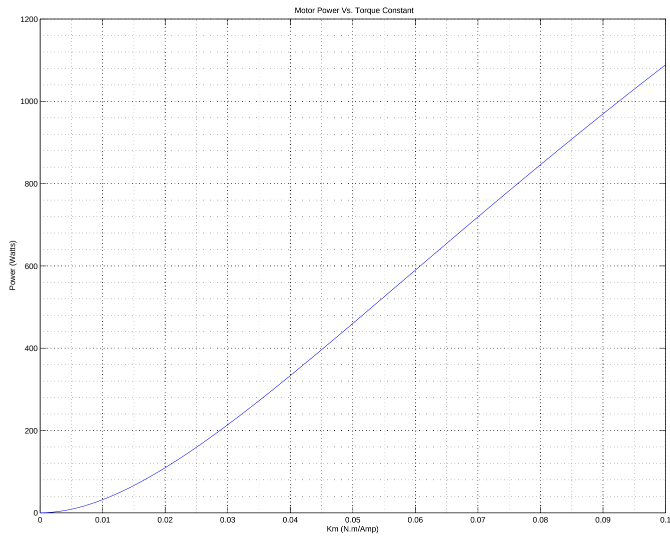


Figure 2.13: Variation of Power Vs. Km

where,

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

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

$\tau(0)$ 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 $m_2 >> m_1$. 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.

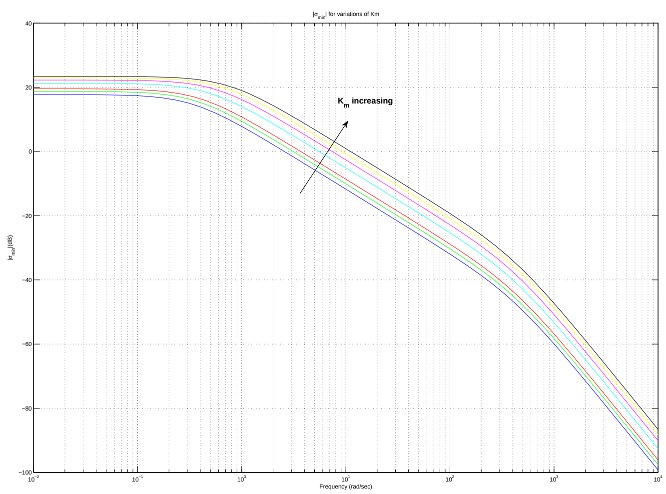


Figure 2.14: Magnitude of Minimum Singular Value for Variations of K_m

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,

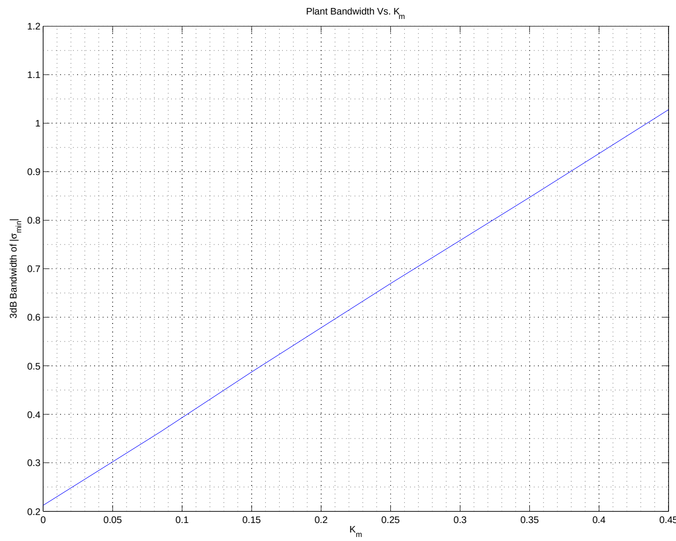


Figure 2.15: Open Loop Bandwidth Vs. K_m

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

$$BW_{3dB}(\sigma_{min}) \approx \frac{2(R_a\beta + K_b K_m)}{R_a J_1} \tag{2.59}$$

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

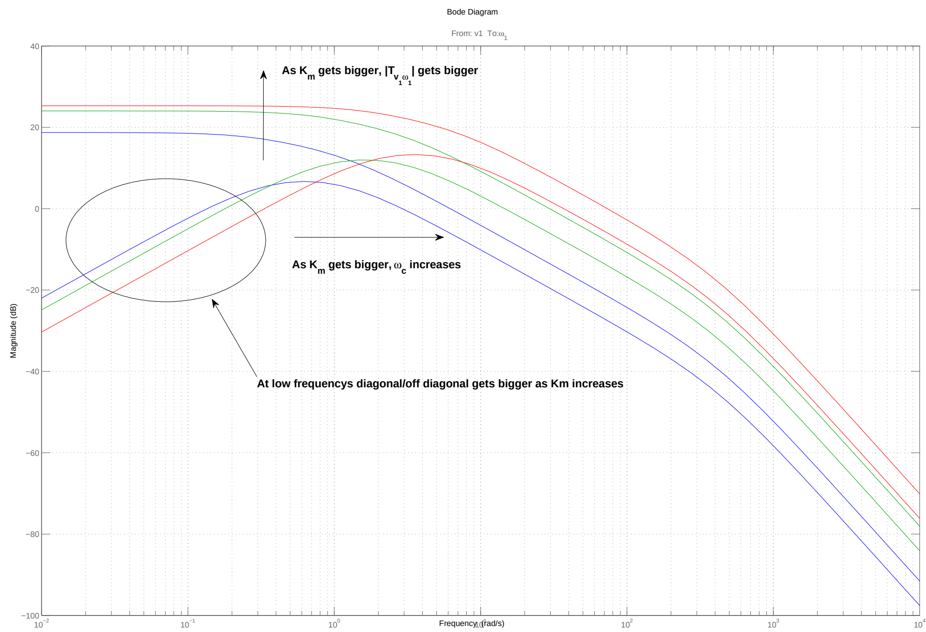


Figure 2.16: Magnitude of Diagonal and Off-Diagonal elements for Variations of K_m

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

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

$$|g_1| = \left|\frac{J_2 + J_1}{J_2 - J_1}\right| \tag{2.61}$$

where, the peak happens at ω_c .

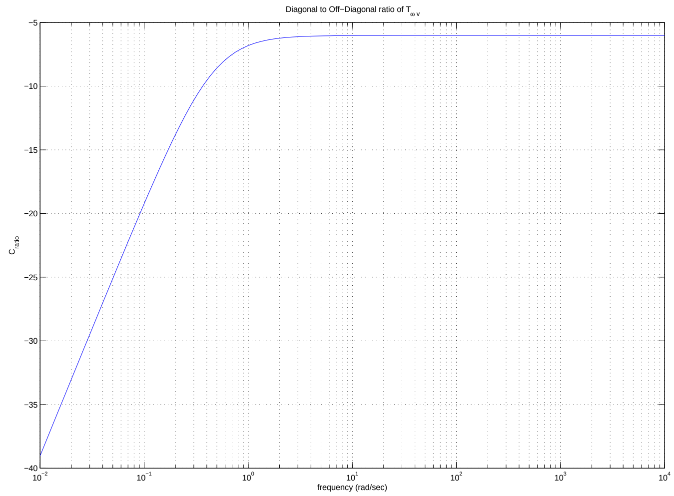


Figure 2.17: Magnitude of Off-Diagonal to Diagonal ratio

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

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

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

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

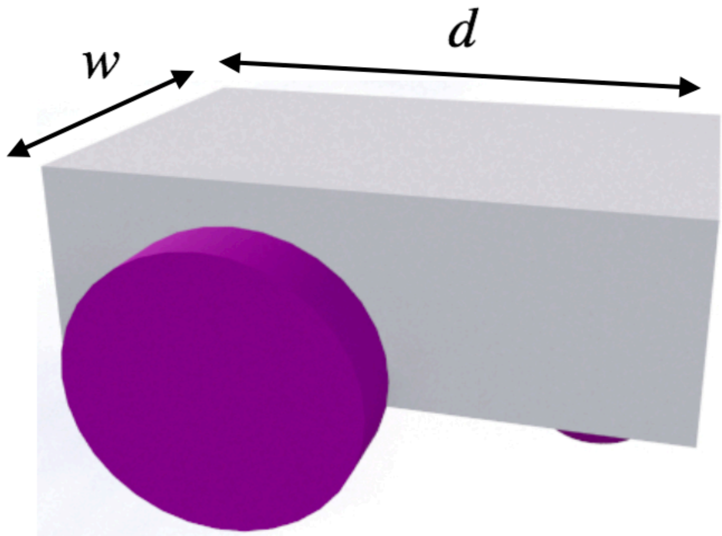


Figure 2.18: Cuboid Shape Mobile Robot

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 :

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

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 :

$$robot\, aspect\, ratio\, (RAR) = \frac{d}{w} \tag{2.65}$$

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.

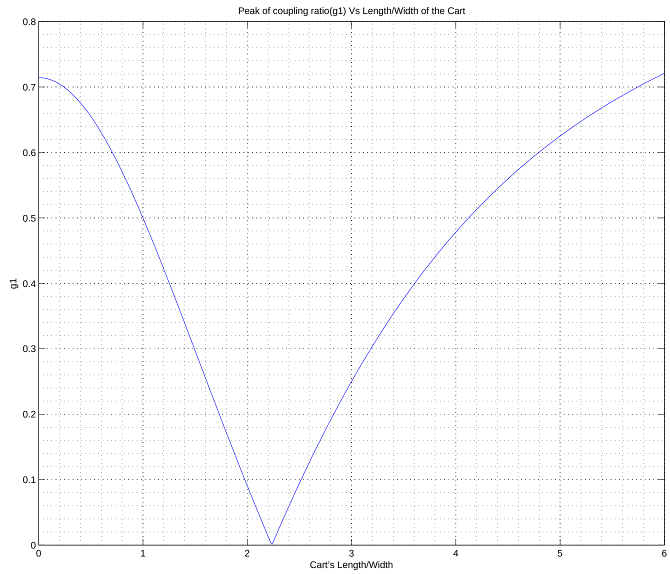


Figure 2.19: Peak coupling ratio behavior Vs. robot's aspect ratio

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

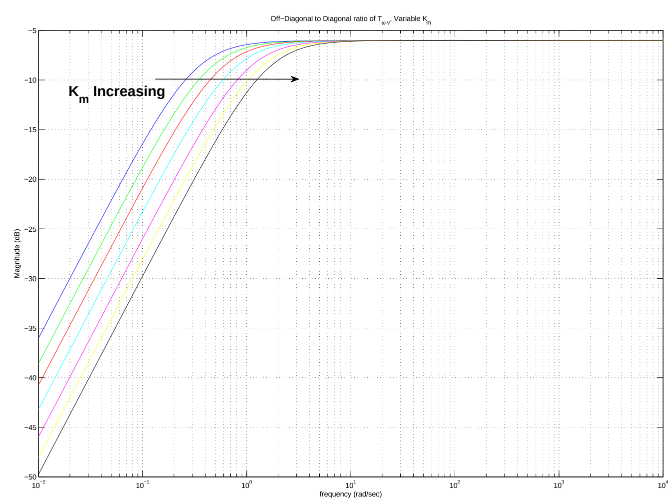


Figure 2.20: Magnitude of Off-Diagonal to Diagonal ratio for Variations of K_m

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).

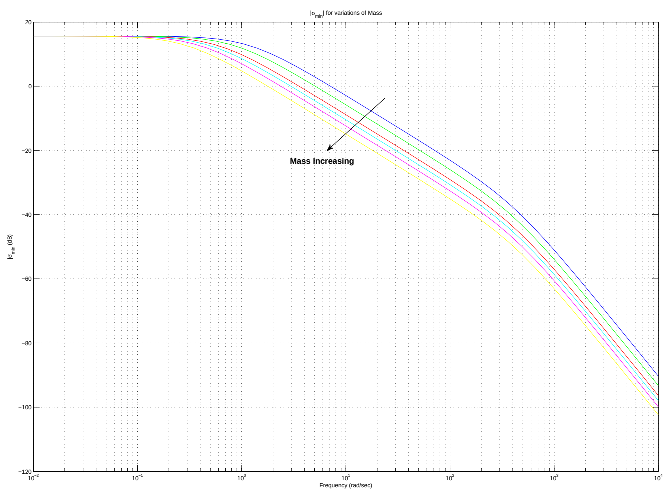


Figure 2.21: Magnitude of Minimum Singular Value for Variations of Mass

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.

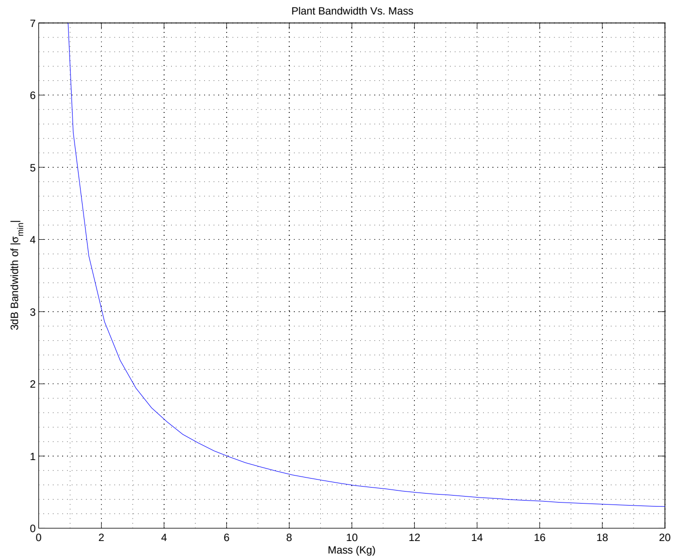


Figure 2.22: Open Loop Bandwidth Vs. *mass*

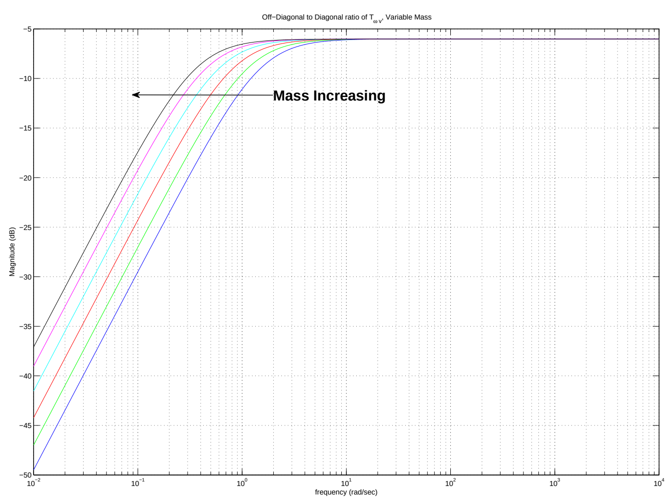


Figure 2.23: Magnitude of Diagonal and Off-Diagonal elements for Variations of Mass

- 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.

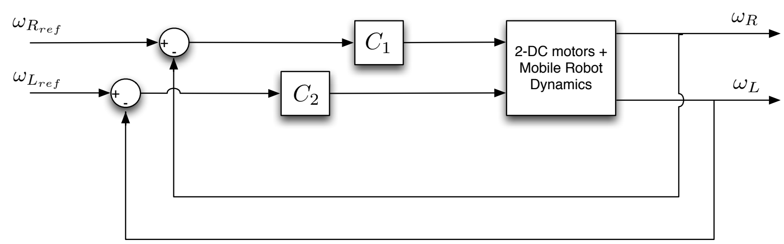


Figure 3.1: Decentralized Controller Architecture for Speed Control

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.

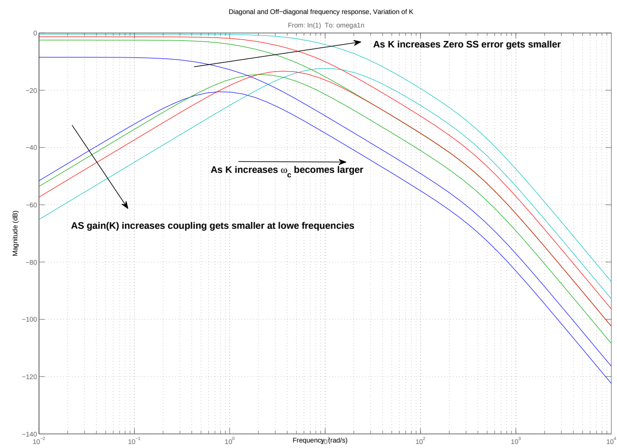


Figure 3.2: Magnitude of Diagonal and off-Diagonal elements for variations of K

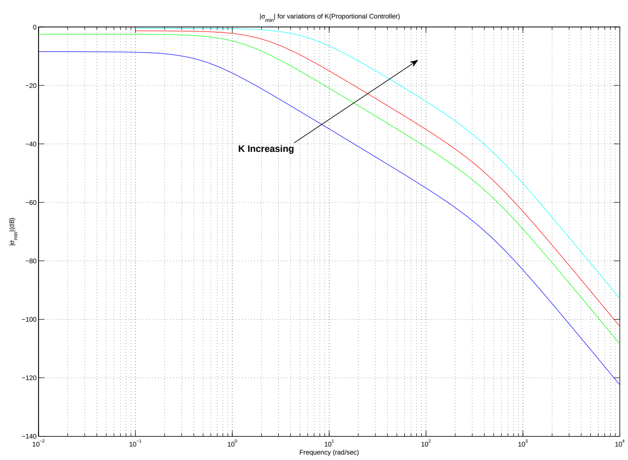


Figure 3.3: Minimum singular value for variations of K

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.

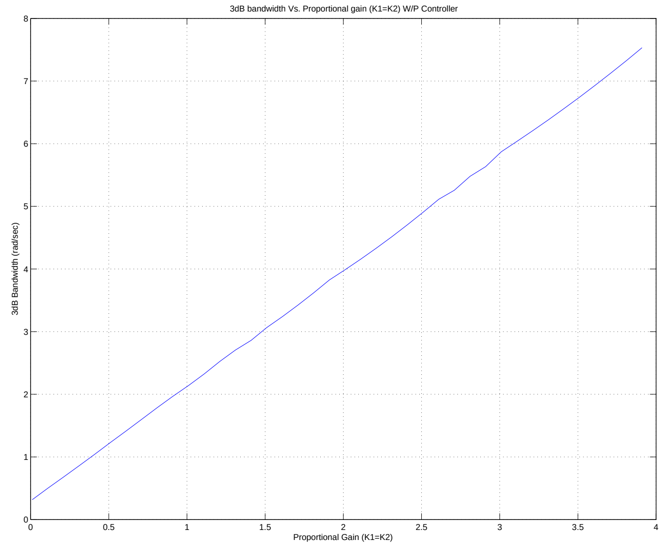


Figure 3.4: Bandwidth of the system Vs. Proportional gain (K)

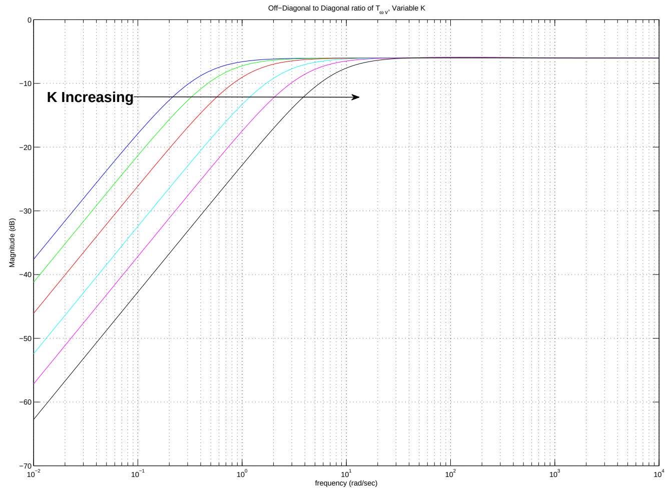


Figure 3.5: Decentralized Controller Architecture for Speed Control

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 :

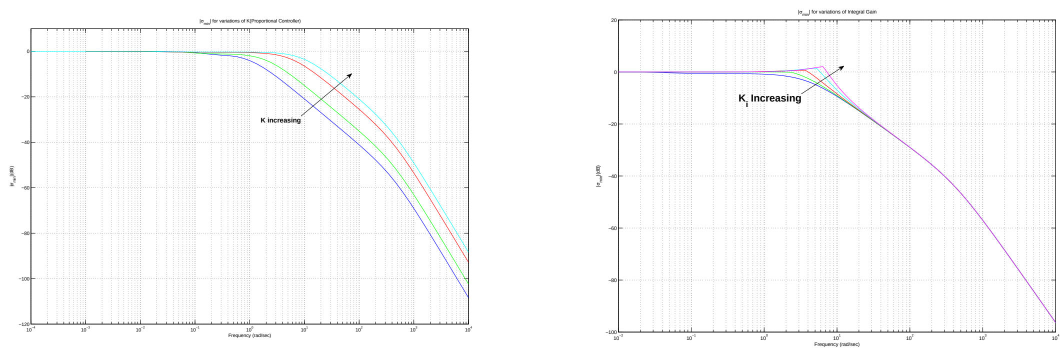
$$C_1 = C_2 = K_p + \frac{K_i}{s} \tag{3.1}$$

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.

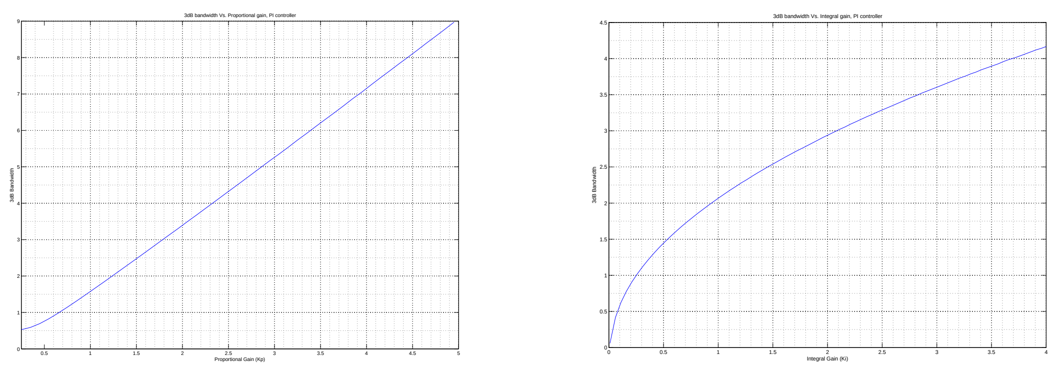
Similar to P controller, increasing K_p and K_i moves the coupling peak to higher



(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

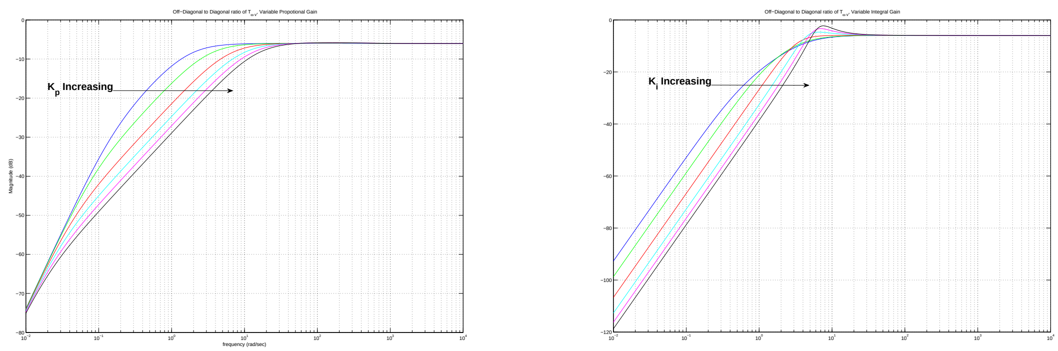
frequencies, as illustrated in Figure 3.8. However, there are two important facts to consider:



(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.



(a) *(b)*

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.

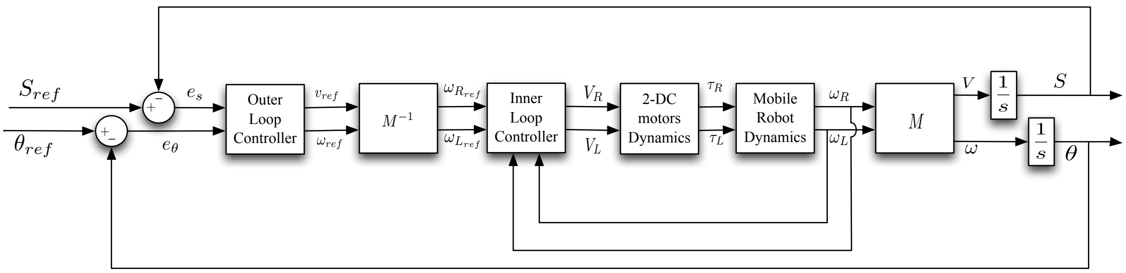


Figure 3.9: Displacement Control Block Diagram from S_{ref} and θ_{ref} to s and θ

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 $\Delta l \to 0$.

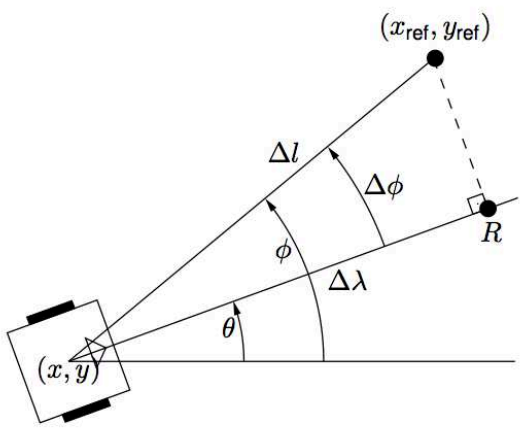


Figure 3.10: Mobile Robot in Cartesian Stabilization mode

In order for the robot to goes 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

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

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

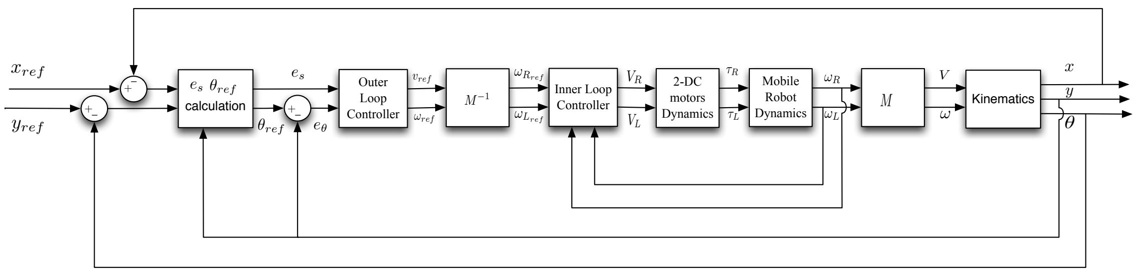


Figure 3.11: Positioning System (Displacement Control) Block Diagram

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

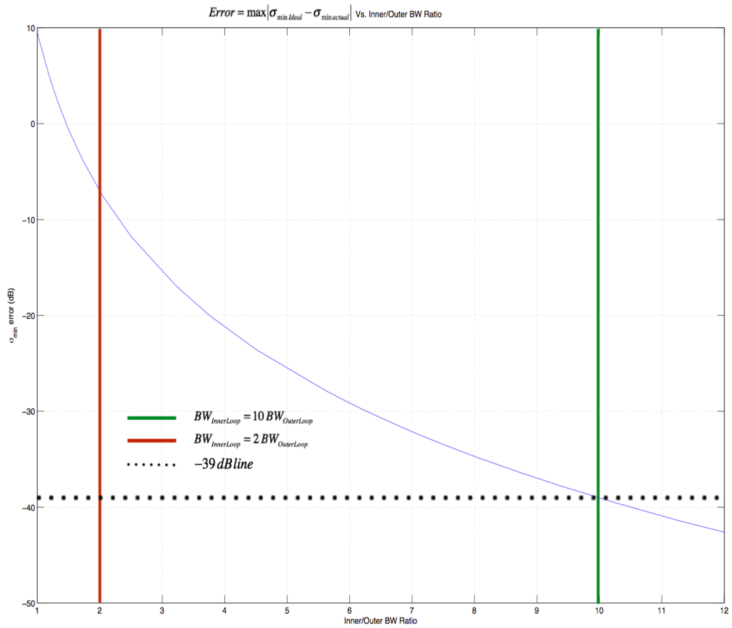


Figure 3.12: Error between ideal (Kinematic) and actual (Kinematic + Dynamics) system Vs. BW ratio

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

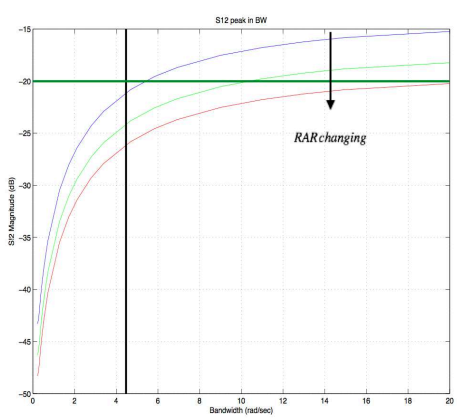
As a rule of thumb: $BW_{InnerLoop} \leq 2BW_{OuterLoop}$ (red line) can yield and error up to $10dB$

3.3 Decentralized Control Limitation

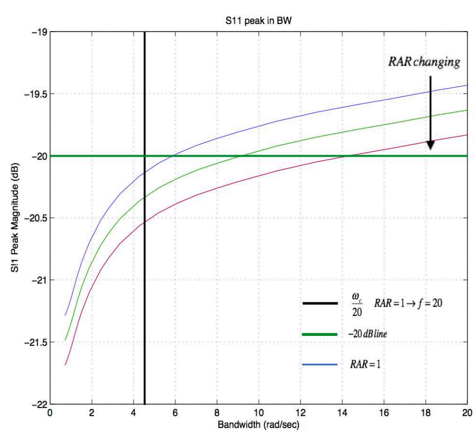
From previous discussions we know that making the inner loop fast is desirable, but of course operating at higher frequencies comes with a price, in our system this price is the sensitivity function. Defining the sensitivity as

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

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 that the diagonal element so that the cross coupling is minimum.



(a)



(b)

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.

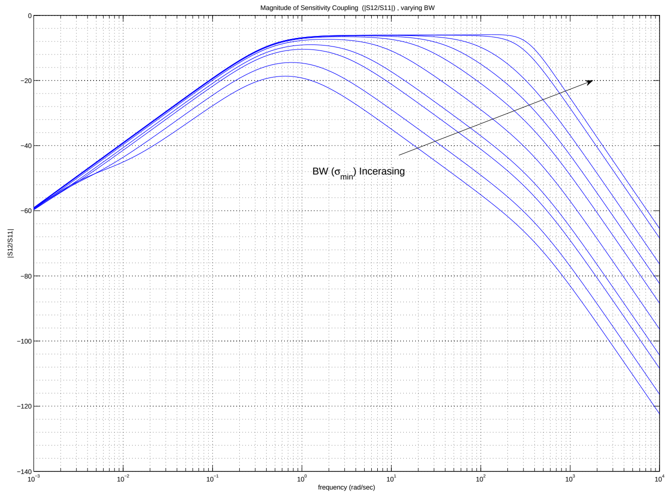


Figure 3.14: $|\frac{S_{12}}{S_{11}}|$

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 $|S_{11}| and |S_{12}| < -20dB$, $\left|\frac{S_{12}}{S_{11}}\right| < -20dB$

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.

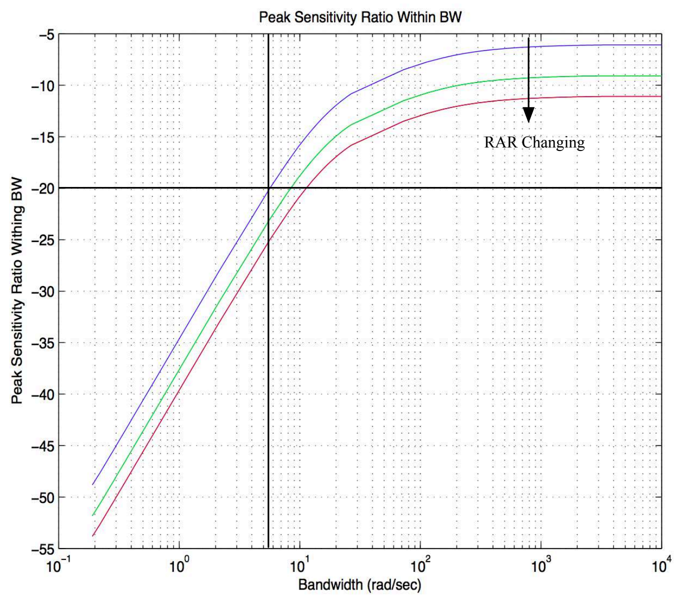


Figure 3.15: Peak $|\frac{S_{12}}{S_{11}}|$ within BW Vs BW

An important fact is that p_s depends on robot's structure, meaning as aspect ratio changes this peak will moves 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*

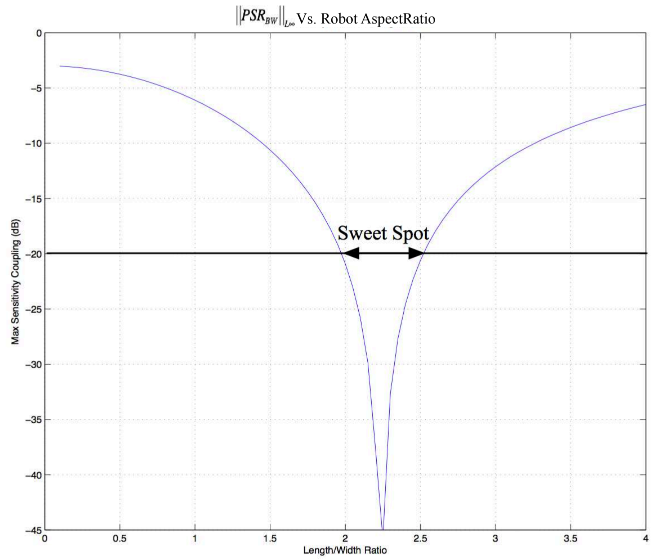


Figure 3.16: p_s Vs. Aspect Ratio

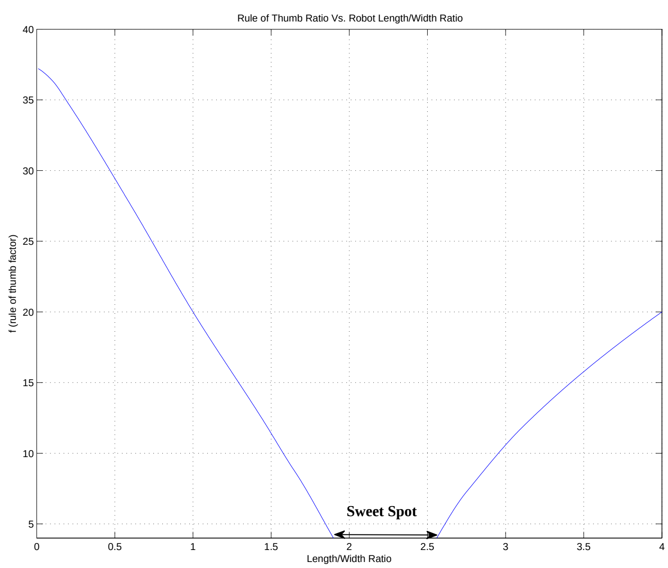


Figure 3.17: Rule of thumb Vs. Aspect Ratio

frequency noise, *sensor noise*, *saturation* and *non-linearties* 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:

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

where $\rho = 0.01$ and $Q = diag[1, 1, 1, 1, q_{I_{a1}}, q_{I_{a2}}, 2, 2]$. $q_{I_{a1}}$ 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.

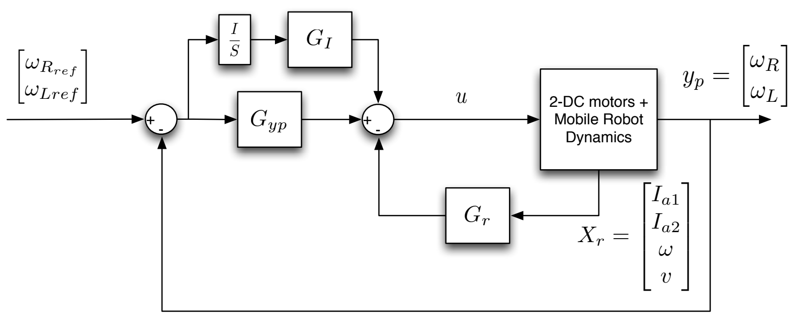


Figure 3.18: Dynamics Plant with a Linear Quadratic Regulator

As stated in section 3.3, the closed loop coupling ratio ($\left|\frac{T_{\omega\omega_{ref}12}}{T_{\omega\omega_{ref}11}}\right|$) 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.

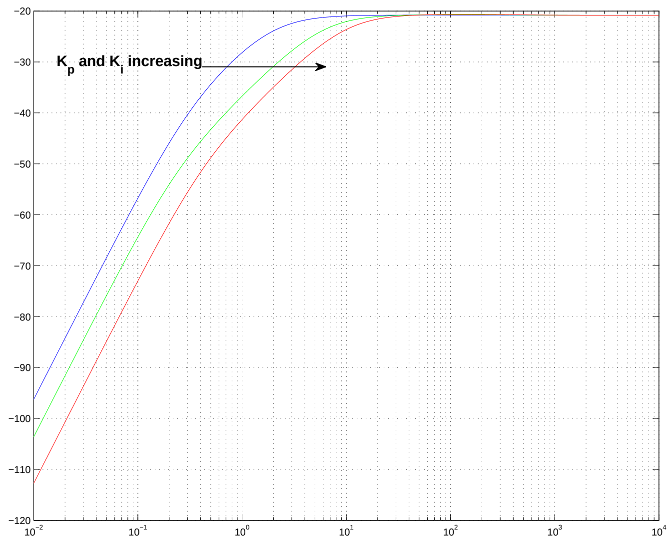


Figure 3.19: Closed loop coupling ratio with decentralized control

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:

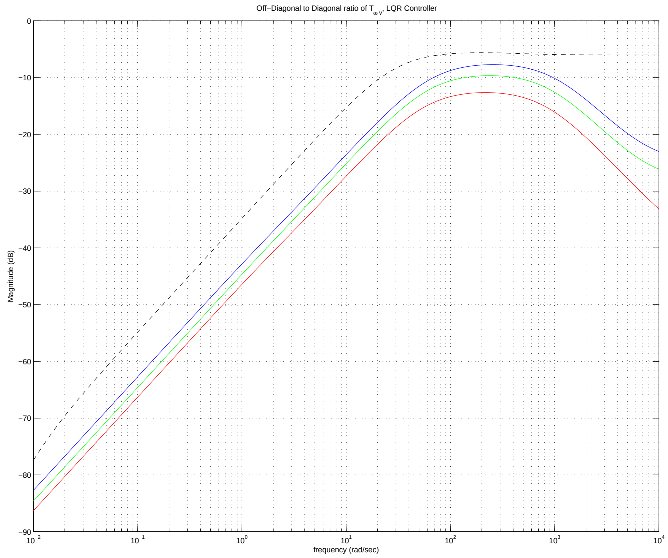


Figure 3.20: Closed loop coupling ratio with centralized control

" 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

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

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

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

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

$$V^T(q)q' = 0 \tag{4.3}$$

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

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

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

$$[\sin\theta \quad -\cos\theta \quad 0]q' = 0 \tag{4.5}$$

Therefore, all the admissible paths can be formulated as

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

where,

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

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

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

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

$$y = h(x) + d(x)u \tag{4.10}$$

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:

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

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

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

$$\theta(g) = atan2(y'(g), x'(g)) + k\pi \qquad k = 0, 1 \tag{4.13}$$

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

$$\begin{aligned} \hat{v}(g) &= \pm\sqrt{x'(g)^2 + y'(g)^2} && (4.14) \\ \hat{\omega}(g) &= \frac{y''(g)x'(g) - x''(g)y'(g)}{x'(g)^2 + y'(g)^2} && (4.15) \end{aligned}$$

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.

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

Chapter 5

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 nonholonomic 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 *Mechatronics 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 Franzi, 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

```matlab
%% MOBILE ROBOT PLANT SETUP AND CONTROLLER DESIGN
%
% In this Document and specific MR is modeled as a 2x2 plant, the plant is
% then analyzed and 2 controllers(LQR and PID) are desigend and compared

%% Plant Setup
% A 2x2 plant modeling two channels of DC motors connected to the wheels of
% a mobile robot, Inputs are voltages and outputs are angular velocity of
% each wheel

clear all;
close all;

%Motor Specification
Km=0.0487;
kb=Km;
La=0.64*(10^-3);
Ra=0.27;

%Robot Specification
r=; %wheel diameter in Meters
m=;  %Mass in Kg
L=; %Axis length in Meters
I=m*(L^2)/6; %moment of inertia for a cube with width = length = L
beta=; %Surface friction

h1=tf(Km,[La Ra]);
h1.u='e1'; h1.y='taum1';

h2=tf(1,[(r^2)*m 0]);
h2.u='x1'; h2.y='vhat1';

h3=tf(L^2,[2*(r^2)*I 0]);
h3.u='x2'; h3.y='omegahat1';

h7=tf(beta,1);
h7.u= 'omega1'; h7.y='tauf1';

h8=tf(kb,1);
h8.u= 'omega1'; h8.y='vb1';
sum1= sumblk('e1=omegar1 - vb1');
sum2= sumblk('tau1=taum1-tauf1');
```

```
sum3= sumblk('x1=tau1+tau2');
sum4= sumblk('x2=tau1-tau2');
sum5= sumblk('omega1 = vhat1 + omegahat1');


h4=tf(Km,[La Ra]);
h4.u='e2'; h4.y='taum2';

h6=tf(1,[(r^2)*m 0]);
h6.u='x4'; h6.y='vhat2';

h5=tf(L^2,[2*(r^2)*I 0]);
h5.u='x3'; h5.y='omegahat2';

h9=tf(beta,1);
h9.u= 'omega2'; h9.y='tauf2';

h10=tf(kb,1);
h10.u= 'omega2'; h10.y='vb2';

sum6= sumblk('e2=omegar2 - vb2');
sum7= sumblk('tau2=taum2-tauf2');
sum8= sumblk('x3=tau1 - tau2');
sum9= sumblk('x4=tau2 + tau1');
sum10= sumblk('omega2 = vhat2 - omegahat2');

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'});
ML.statename={'ia1','x2','x3','ia2','x5','x6'};

%Plant Plots

%StepPlot
f1=figure;
f1=stepplot(ML);
grid on;
title('Step response of the 2 Motor channels, Robot''s dynamics included');

%Singular Value plot
f2=figure;
f2=sigmaplot(ML,{10^-2,10^4});
setoptions(f2,'FreqUnits','Hz');
grid;
title('Singular Values of the plant');
```

```
MLmin=minreal(ML,[],0);
MLmin.u={'omegar1n','omegar2n'};
MLmin.y={'omega1n','omega2n'};

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

% * Singular Value plot
f4=figure;
f4=sigmaplot(MLmin);
setoptions(f4,'FreqUnits','Hz');
grid;
title('Singular Values of the 2 Motor channels, Robot''s dynamics included');

[Aol,Bol,Col,Dol]=ssdata(MLmin);


kff1=1/dcgain(MLmin(1,1));
kff2=1/dcgain(MLmin(2,2));

FFrob=MLmin*[kff1,0;0,kff2];

FFrob.u={'omegar1n','omegar2n'};
FFrob.y={'omega1n','omega2n'};

% %StepPlot
ff1=figure;
ff1=stepplot(FFrob);
grid on;
title('Step response of the 2 Motor channels Feed Forward Open Loop, Robot''s dynamics included');

% %Singular Value plot
ff2=figure;
ff2=sigmaplot(FFrob);
setoptions(ff2,'FreqUnits','Hz');
grid;
title('Singular Values of the 2 Motor channels Feed Forward Open Loop, Robot''s dynamics included');

%% LQR Design
```

```matlab
% In this section an LQR controller is designed for the plant and the
% closed loop responses are then compared to the open loop propertise
close all;
clear MLaug P P1 Z Q1 R1 Klqr P2 K OL CL roblqr

%augment each chanel with 1/s

h11=tf(1,[1 0]);
h11.u='omega1n'; h11.y='omega1/s';

h12=tf(1,[1 0]);
h12.u='omega2n'; h12.y='omega2/s';

%design plant with omega/s outputs

MLaug= connect(MLmin,h11,h12,{'omegar1n','omegar2n'},{'omega1/s','omega2/s'});

%Klqr design

P=augstate(MLaug); %Augment states with output
P1=P(3:8,1:2); %2 state are the same as the output which can be eliminited

Z=P1.c;

Q1=Z'*Z;

R1=0.001*eye(2);

Klqr=lqr(P1,Q1,R1);

%Prepearing the non-augmented system for simulation
P2=augstate(MLmin);

H=append(1,1,1,1,tf(1,[1 0]),tf(1,[1 0]));
K=Klqr*H; %putting integrator on the last two channels

FB=[0,0,1,0,0,0;
    0,0,0,1,0,0;
    0,0,0,0,1,0;
    0,0,0,0,0,1;
    1,0,0,0,0,0;
    0,1,0,0,0,0];

```

```
OL=FB*P2*K;
CL=feedback(OL,eye(6),1:6,1:6);

roblqr=CL([5 6],[5 6]);
roblqr.u={'Omegar1','Omegar2'}; roblqr.y={'Omega1','Omega2'};

f5=figure;
f5=stepplot(roblqr);
title('Closed Loop Step response of a 2 channel motor/Robot with LQR controller');
grid on;

w=logspace(-2,4,5000);

a=bode(roblqr(1,1),w);
b=bode(roblqr(1,2),w);
rat=b ./ a;
ratio=zeros(length(rat(1,1,:)),1);
ratio(:)=rat(1,1,:);

figure;
semilogx(w,20*log10(ratio));
hold on;
title('Off-Diagonal to Diagonal ratio of T_{\omega v}, LQR Controller');
xlabel('frequency (rad/sec)');
ylabel('Magnitude (dB)');
grid on;
%
% figure;
% bodemag(roblqr);

[Alqr,Blqr,Clqr,Dlqr]=ssdata(roblqr);
%%
close all;

t=0:0.1:15;
Td=-0.5 *(t>5 & t<10);   %- 0.5 disturbance between 5s to 10s
u=[ones(size(t));Td];    % augmenting step of size one and Td to u as input

f6=figure;
f6=lsimplot(FFrob(1,[1 2]),roblqr(1,[1 2]),'--',u,t);
grid on;
title('Motor 1 Step response, and reaction to a input disturbance caused by coupling of motor 2');
legend('OpenLoop','LQR Controller');
```

```
u1=[Td;ones(size(t))];
f7=figure;
f7=lsimplot(FFrob(2,[1 2]),roblqr(2,[1 2]),'--',u1,t);
grid on;
title('Motor 2 Step response, and reaction to a input
    disturbance caused by coupling of motor 1');
legend('OpenLoop','LQR Controller');
%ML_aug= connect(ML,h11,h12,{'omegar1','omegar2'},{'omega1','
    omega1/s','omega2','omega2/s'});

f8=figure;
f8=stepplot(FFrob,roblqr,'--');
grid on;
title('step response of the Openloop VS Closed Loop');
legend('Open Loop','LQR Controller');

f9=figure;
f9=sigmaplot(FFrob);
hold on;
sigmaplot(roblqr,'--');
grid on;
legend('Open Loop','LQR Controller');

f10=figure;
f10=lsimplot(roblqr,u,t);
grid on;
title('Motor 1 Step response, and reaction to a input
    disturbance caused by coupling of motor 2');
legend('LQR Controller');


f11=figure;
f11=lsimplot(roblqr,u1,t);
grid on;
title('Motor 2 Step response, and reaction to a input
    disturbance caused by coupling of motor 1');
legend('LQR Controller');

%% PI Controller
% In this section a PI controller is desigend for each
    channel

C1=pidtune(MLmin(1,1),'pi');
C2=pidtune(MLmin(2,2),'pi');
% C1.ki=1;
% C2.ki=1;
```

```
C1.kp=10;
C2.kp=10;
KPID=append(C1,C2);

FF=MLmin*KPID;

robpid=feedback(FF,eye(2));

w=logspace(-2,4,100);

a=bode(robpid(1,1),w);
b=bode(robpid(1,2),w);
rat=b ./ a;
ratio=zeros(length(rat(1,1,:)),1);
ratio(:)=rat(1,1,:);

semilogx(w,20*log10(ratio),'k--');

% stepplot(robpid);
%% PID vs. LQR
% The following plots provide a comparison between PI and LQR controllers
% for the same plant

close all;

t=0:0.1:15;
Td=-0.5 *(t>5 & t<10);    %- 0.5 disturbance between 5s to 10s
u=[ones(size(t));Td];     % augmenting step of size one and Td to u as input

f6=figure;
f6=lsimplot(robpid(1,[1 2]),roblqr(1,[1 2]),'--',u,t);
grid on;
title('Motor 1 Step response, and reaction to a input disturbance caused by coupling of motor 2');
legend('PID Controller','LQR Controller');

u1=[Td;ones(size(t))];
f7=figure;
f7=lsimplot(robpid(2,[1 2]),roblqr(2,[1 2]),'--',u1,t);
grid on;
title('Motor 2 Step response, and reaction to a input disturbance caused by coupling of motor 1');
legend('PID Controller','LQR Controller');
%ML_aug= connect(ML,h11,h12,{'omegar1','omegar2'},{'omega1','omega1/s','omega2','omega2/s'});
```

```
f8=figure;
f8=stepplot(robpid,roblqr,'--');
grid on;
title('step response of the Openloop VS Closed Loop');
legend('PID Controller','LQR Controller');

f9=figure;
f9=sigmaplot(robpid);
hold on;
sigmaplot(roblqr,'--');
grid on;
legend('PID Controller','LQR Controller');

f10=figure;
f10=lsimplot(roblqr,u,t);
grid on;
title('Motor 1 Step response, and reaction to a input
    disturbance caused by coupling of motor 2');
legend('LQR Controller');


f11=figure;
f11=lsimplot(roblqr,u1,t);
grid on;
title('Motor 2 Step response, and reaction to a input
    disturbance caused by coupling of motor 1');
legend('LQR Controller');

f10=figure;
f10=lsimplot(robpid,u,t);
grid on;
title('Motor 1 Step response, and reaction to a input
    disturbance caused by coupling of motor 2');
legend('PID Controller');


f11=figure;
f11=lsimplot(robpid,u1,t);
grid on;
title('Motor 2 Step response, and reaction to a input
    disturbance caused by coupling of motor 1');
legend('PID Controller');


%% Sensitivity analysis
```

```
% This section Sensetivity and Complement sensitivity of
    formerly designed
% controllers are plotted and compared

close all;

Plant=FB*P2;
Controller=ss(K);
Loopslqr=loopsens(Plant,Controller);

Loopspid=loopsens(MLmin,KPID);

figure;
bodemag(Loopslqr.Si,Loopspid.Si);
title('Sensitivity Bode Magnitude');
legend('LQR Controller','PID Controllers');
grid on;

figure;
f19=bodeplot(Loopslqr.Si,Loopspid.Si);
title('Sensitivity Bode Phase');
legend('LQR Controller','PID Controllers');
grid on;
setoptions(f19,'MagVisible','off');


figure;
bodemag(Loopslqr.Ti,Loopspid.Ti);
title('Complement Sensitivity Bode Magnitude');
legend('LQR Controller','PID Controller');
grid on;

figure;
f20=bodeplot(Loopslqr.Ti,Loopspid.Ti);
title('Complement Sensitivity Bode Phase');
legend('LQR Controller','PID Controllers');
grid on;
setoptions(f20,'MagVisible','off');

figure;
f22=bodeplot(Loopslqr.Si,Loopspid.Si);
title('Sensitivity Bode');
legend('LQR Controller','PID Controllers');
grid on;

figure;
f21=bodeplot(Loopslqr.Ti,Loopspid.Ti);
```

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

```
%%
clear all;
close all;
f1=figure;

tmc= %motor torque constant;

for i=1:length(tmc)

%motor specs
Km=tmc(i);
kb=;
La=;
Ra=;
beta=;
J=;


h1= tf(Km,[La,Ra]);
h1.u='e'; h1.y='tau';

h2= tf(1,[J,beta]);
h2.u='tau'; h2.y='omega';

h3= tf(kb,1);
h3.u='omega'; h3.y='vb';

sum1=sumblk('e=v-vb');

dcm=connect(ss(h1),h2,h3,sum1,'v',{'tau','omega'});

%
t=0:1:24;
u=t;
[y,t]= lsim(dcm,u,t);
%
Power(i)=y(24,1)*y(24,2);
Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; %Power in watts
Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
dc=Km/(beta*Ra+Km*kb);

end

```

```
plot(tmc,Power);
hold on;

plot(tmc,Power,'r');

%%
clear all;
close all;
f1=figure;

R=%armature resistance;

for i=1:length(R)

Km=;
kb=;
La=;
Ra=R(i);
beta=;
J=;


h1= tf(Km,[La,Ra]);
h1.u='e'; h1.y='tau';

h2= tf(1,[J,beta]);
h2.u='tau'; h2.y='omega';

h3= tf(kb,1);
h3.u='omega'; h3.y='vb';

sum1=sumblk('e=v-vb');

dcm=connect(ss(h1),h2,h3,sum1,'v',{'tau','omega'});


Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; %Power in watts
Power(i)=Power2(i)*(1.341*10^-3); %Power in hp

end
%
% plot(tmc,Power);
% hold on;

plot(R,Power,'r');

```

```matlab

%% DC motor bandwidth req

clear all;
close all;
f1=figure;

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.',
    'r-.','g--'};

tmc=[0.01 0.05 0.08 0.1 0.3 0.4 0.6 0.7 0.9 2 3 4] ;

f2=figure;
f3=figure;
f4=figure;
f5=figure;

for i=1:length(tmc)

Km=tmc(i);
kb=0.0847;
La=0.64*(10^-3);
Ra=0.27;
beta=0.021;
J=0.00057892;


h1= tf(Km,[La,Ra]);
h1.u='e'; h1.y='tau';

h2= tf(1,[J,beta]);
h2.u='tau'; h2.y='omega';

h3= tf(kb,1);
h3.u='omega'; h3.y='vb';

sum1=sumblk('e=v-vb');

dcm=connect(ss(h1),h2,h3,sum1,'v',{'tau','omega'});
figure(f1);
stepplot(dcm,lineorder{i});
hold on;

figure(f2);
bodemag(dcm,lineorder{i});
hold on;
grid on;
```

```
bw(i)=bandwidth(dcm(2,1));

Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2)  ; %Power in watts
Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
% dc=Km/(beta*Ra+Km*kb);

figure(f4);
pzmap(dcm,lineorder{i});
hold on;

S=stepinfo(dcm);
settime(i)=S(2).SettlingTime;


end
%
% plot(tmc,Power);
% hold on;

% plot(tmc,Power,'r');
figure(f3);
plot(Power,bw);
grid on;
xlabel('Power');
ylabel('Bandwidth');

figure(f5);
plot(tmc,settime);

%% DC motor + variable inertia plots

clear all;
close all;
f1=figure;

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.','r-.','g--'};

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];
inertia=(10^-5)*[10 40 60 70 80 90];

f2=figure;
f3=figure;
f4=figure;
```

```
for j=1:length(inertia)

    for i=1:length(tmc)

        Km=tmc(i);
        kb=0.0847;
        La=0.64*(10^-3);
        Ra=0.27;
        beta=0.021;
        J=inertia(j);


        h1= tf(Km,[La,Ra]);
        h1.u='e'; h1.y='tau';

        h2= tf(1,[J,beta]);
        h2.u='tau'; h2.y='omega';

        h3= tf(kb,1);
        h3.u='omega'; h3.y='vb';

        sum1=sumblk('e=v-vb');

        dcm=connect(ss(h1),h2,h3,sum1,'v',{'tau','omega'});

%          figure(f2);
%          bodemag(dcm,lineorder{i});
%          hold on;
%          grid on;
%
        bw(j,i)=bandwidth(dcm(2,1));

        Power2(j,i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; % Power in watts
        Power(j,i)=Power2(i)*(1.341*10^-3); %Power in hp

    end
figure(f2);
plot(Power2(j,:),bw(j,:),lineorder{j});
hold on;
grid on;

end

%
% plot(tmc,Power);
% hold on;
```

```
% plot(tmc,Power,'r');
% figure(f3);
% plot(Power,bw);
% grid on;
% xlabel('Power');
% ylabel('Bandwidth');
%% PURE TF analysis of the motor
%
clear all;
close all;
f1=figure;

% for k=1:40

tmc=;
for i=1:length(tmc)

Km=tmc(i);

kb=;
La=;
Ra=;
beta=;
J=;

dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);

figure(f1);
pzmap(dcm);
hold on;

dcg(i)=dcgain(dcm);
end

%% PURE TF analysis of the motor
%
clear all;
close all;

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.',
    'r-.','g--'};
tmc=%motor torque constant;

f1=figure;
f2=figure;
f3=figure;
```

```
f4=figure;
f5=figure;

% for k=1:40


for i=1:length(tmc)

Km=tmc(i);
kb=0.0847;
La=0.64*(10^-3);
Ra=0.27;
beta=0.021;
J=0.00057892;

dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);
%
figure(f1);
pzmap(dcm);
hold on;
%
figure(f2);
bodemag(dcm,lineorder{i});
hold on;
%
figure(f3);
step(dcm,lineorder{i});
hold on;
%
S=stepinfo(dcm);
settime(i)=S.SettlingTime;

% bw(i)=bandwidth(dcm);

vin=; %input voltage
Ts=(Km/Ra)*vin; %Stall Torque
omega0=vin/kb; %No load speed
powermax(i)=(Ts*omega0)/4;
end

figure(f4);
plot(tmc,powermax);

figure(f5);
plot(tmc,settime);
```

```
hold on;

%% 2 dc motors comparison
close all;
clear all;


lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.',
    'r-.','g--'};
tmc=[1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000
    12000];

f1=figure;
f2=figure;
f3=figure;
f4=figure;
f5=figure;

% for k=1:40


for i=1:length(tmc)

Km2=tmc(i);
kb2=;
La2=;
Ra2=;
beta2=;
J2=;


dcm=tf(Km2,[J2*La2 J2*Ra2+beta2*La2 beta2*Ra2+Km2*kb2]);
%
figure(f1);
pzmap(dcm);
hold on;
%
figure(f2);
bodemag(dcm,lineorder{i});
hold on;
%
figure(f3);
step(dcm,lineorder{i});
hold on;
%
S=stepinfo(dcm);
settime(i)=S.SettlingTime;
```

```
% bw(i)=bandwidth(dcm);

vin=24; %input voltage
Ts=(Km2/Ra2)*(vin^2); %Stall Torque
omega0=vin/kb2; %No load speed
powermax(i)=(Ts*omega0)/4;
Power2(i)=(24^2)*beta2*((Km2/(beta2*Ra2+Km2*kb2))^2) ; %Power in watts
end

figure(f4);
plot(tmc,powermax);
hold on;
plot(tmc,Power2,'r');

figure(f5);
plot(tmc,settime);

%% 2 dc motors comparison
close all;
clear all;

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.','r-.','g--'};
tmc=[0.01 0.05 0.1 0.2 0.3 0.5 1];

f1=figure;
f2=figure;
f3=figure;
f4=figure;
f5=figure;

% for k=1:40

for i=1:length(tmc)

Km2=tmc(i);
kb2=;
La2=;
Ra2=;
beta2=;
J2=;


dcm=tf(Km2,[J2*La2 J2*Ra2+beta2*La2 beta2*Ra2+Km2*kb2]);
%
```

```
figure(f1);
pzmap(dcm);
hold on;
%
figure(f2);
bodemag(dcm,lineorder{i});
hold on;
%
figure(f3);
step(dcm,lineorder{i});
hold on;
%
S=stepinfo(dcm);
settime(i)=S.SettlingTime;

% bw(i)=bandwidth(dcm);

vin=24; %input voltage
Ts=(Km2/Ra2)*(vin^2); %Stall Torque
omega0=vin/kb2; %No load speed
powermax(i)=(Ts*omega0)/4;
Power2(i)=(24^2)*beta2*((Km2/(beta2*Ra2+Km2*kb2))^2) ; %Power
    in watts
end

figure(f4);
plot(tmc,powermax);
hold on;
plot(tmc,Power2,'r');

figure(f5);
plot(tmc,settime);
```

```
%% ROVER
% Properties of the plant
clear all;
close all;

%Motor Specification
Km=%torque constant;
kb=Km;
La=%armature inductance;
Ra=%armature resistance;

%Robot Specification
r=; %wheel diameter in Meters
m=;  %Mass in Kg
L=; %Axis length in Meters
```

```
I=m*(L^2)/6; %moment of inertia for a cube with width =
    length = L
beta=; %Surface friction

h1= tf(Km,[La,Ra]);
h1.u='e'; h1.y='tau';

h2= tf(1,[J,beta]);
h2.u='tau'; h2.y='omega';

h3= tf(kb,1);
h3.u='omega'; h3.y='vb';

sum1=sumblk('e=v-vb');

dcm=connect(ss(h1),h2,h3,sum1,'v',{'tau','omega'});

Power2=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; %Power in watts
powerrov=Power2*(1.341*10^-3); %Power in hp

ppmrov=powerrov/m;

h1=tf(Km,[La Ra]);
h1.u='e1'; h1.y='tau1';

h2=tf(1,[(r^2)*m 0]);
h2.u='x1'; h2.y='vhat1';

h3=tf(L^2,[2*(r^2)*I 0]);
h3.u='x2'; h3.y='omegahat1';

h7=tf(beta,1);
h7.u= 'omega1'; h7.y='tauf1';

h8=tf(kb,1);
h8.u= 'omega1'; h8.y='vb1';

%sumblocks in channel 1
sum1= sumblk('e1=omegar1 - vb1');
sum2= sumblk('c1=tau1-tauf1');
sum3= sumblk('x1=c1+tau2');
sum4= sumblk('x2=c1-tau2');
sum5= sumblk('omega1 = vhat1 + omegahat1');


%Transfer functions and their input output names in channel 2

```

```
h4=tf(Km,[La Ra]);
h4.u='e2'; h4.y='tau2';

h6=tf(1,[(r^2)*m 0]);
h6.u='x4'; h6.y='vhat2';

h5=tf(L^2,[2*(r^2)*I 0]);
h5.u='x3'; h5.y='omegahat2';

h9=tf(beta,1);
h9.u= 'omega2'; h9.y='tauf2';

h10=tf(kb,1);
h10.u= 'omega2'; h10.y='vb2';

sum6= sumblk('e2=omegar2 - vb2');
sum7= sumblk('c2=tau2-tauf2');
sum8= sumblk('x3=tau1 - c2');
sum9= sumblk('x4=c2 + tau1');
sum10= sumblk('omega2 = vhat2 - omegahat2');

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'});
ML.statename={'ia1','x2','x3','ia2','x5','x6'};

MLminrover=minreal(ML,[],0);
MLminrover.u={'omegar1n','omegar2n'};
MLminrover.y={'omega1n','omega2n'};

BWrovol=bandwidth(MLminrover(1,1)); % Motor Bandwidth

%evaluating the response at 0 rad/sec
mag0rov=bode(MLminrover,0);
magrat0rovol=mag0rov(1,1)/mag0rov(1,2);

%evaluating the response at OmegaBW
%
magbw=bode(MLmin,reqbw);
magratbw(k)=magbw(1,1)/magbw(1,2);

S=stepinfo(MLminrover(1,1));
tsrovol=S.SettlingTime;

rob=feedback(MLminrover,eye(2));

BWrovcl=bandwidth(rob(1,1)); % Motor Bandwidth
```

```
%evaluating the response at 0 rad/sec
mag0rov=bode(rob,0);
magrat0rovcl=mag0rov(1,1)/mag0rov(1,2);

%evaluating the response at OmegaBW
%
magbw=bode(MLmin,reqbw);
magratbw(k)=magbw(1,1)/magbw(1,2);

S=stepinfo(rob(1,1));
tsrovcl=S.SettlingTime;

%% Open Loop , POWER/MASS Plots
% Properties of the plant
clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol BWrovcl BWrovol powerrov ppmrov MLminrover rob;
close all;

mc=%motor torque constant;
mass=%system Mass;
reqbw=; %Required BW in rad/sec

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.','r-.','g--'};

%Power
for i=1:length(mc)

Km=mc(i);
kb=0.0847;
La=0.64*(10^-3);
Ra=0.27;
beta=0.021;
J=0.00057892;

dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);

Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; %Power in watts
Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
end

f5=figure;
plot(mc,Power2);
title('Motor Power Vs. Torque Constant');
xlabel('Km (N.m/Amp)');
```

```
ylabel('Power (Watts)');
grid minor;

 f6=figure;
 f7=figure;
 f8=figure;
 f9=figure;
 f10=figure;
 f11=figure;
 f12=figure;
 f13=figure;
 f14=figure;
 f15=figure;

% rover bode
% bodemag(MLminrover,'k-');
% h=findobj(gcf,'type','line');
% set(h,'linewidth',1.1);
% hold on;


k=1;


for i=1:length(mass)

    for j=1:length(mc)

        Km=mc(j);
        kb=0.0847;
        La=0.64*(10^-3);
        Ra=0.27;
        r=0.1;
        m=mass(i);
        L=0.5;
        I=m*(L^2)/6; %moment of inertia for a cube with width
            = length = L
        beta=0.021;

        pmr(k)=Power(j)/mass(i); %Computing Power to Mass
            ratio
        pmr1(i,j)=Power2(j)/mass(i);

        %Transfer functions and their input output names in
            chanel 1

        h1=tf(Km,[La Ra]);
```

```
h1.u='e1'; h1.y='tau1';

h2=tf(1,[(r^2)*m 0]);
h2.u='x1'; h2.y='vhat1';

h3=tf(L^2,[2*(r^2)*I 0]);
h3.u='x2'; h3.y='omegahat1';

h7=tf(beta,1);
h7.u= 'omega1'; h7.y='tauf1';

h8=tf(kb,1);
h8.u= 'omega1'; h8.y='vb1';

%sumblocks in channel 1
sum1= sumblk('e1=omegar1 - vb1');
sum2= sumblk('c1=tau1-tauf1');
sum3= sumblk('x1=c1+tau2');
sum4= sumblk('x2=c1-tau2');
sum5= sumblk('omega1 = vhat1 + omegahat1');


%Transfer functions and their input output names in
   channel 2

h4=tf(Km,[La Ra]);
h4.u='e2'; h4.y='tau2';

h6=tf(1,[(r^2)*m 0]);
h6.u='x4'; h6.y='vhat2';

h5=tf(L^2,[2*(r^2)*I 0]);
h5.u='x3'; h5.y='omegahat2';

h9=tf(beta,1);
h9.u= 'omega2'; h9.y='tauf2';

h10=tf(kb,1);
h10.u= 'omega2'; h10.y='vb2';


%sumblocks in channel 1
sum6= sumblk('e2=omegar2 - vb2');
sum7= sumblk('c2=tau2-tauf2');
sum8= sumblk('x3=tau1 - c2');
sum9= sumblk('x4=c2 + tau1');
sum10= sumblk('omega2 = vhat2 - omegahat2');
```

```
        %connect models
        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'});
        ML.statename={'ia1','x2','x3','ia2','x5','x6'};

        %Minimum realization Plant

        MLmin=minreal(ML,[],0);
        MLmin.u={'omegar1n','omegar2n'};
        MLmin.y={'omega1n','omega2n'};

        %3dB bandwidth
        BW(k)=bandwidth(MLmin(1,1));
        BW1(i,j)=bandwidth(MLmin(1,1));

        %Max Transient frequency
          [mag,phase,w]=bode(MLmin(1,2));
          [Y,I]=max(mag);

        maxoffdiagmag(i,j)= Y;
        maxoffdiagfreq(i,j)=w(I);

        %Transient Mag / DC gain

        TMDC(i,j)=abs(Y/dcgain(MLmin(1,2)));

        %evaluating the response at 0 rad/sec
        mag0=bode(MLmin,0);
        magrat0(k)=mag0(1,1)/mag0(1,2);

        %evaluating the response at OmegaBW

        magbw=bode(MLmin,reqbw);
        magratbw(k)=magbw(1,1)/magbw(1,2);

        S=stepinfo(MLmin(1,1));
        ts(k)=S.SettlingTime;

        k=k+1;


    end

    figure(f6);
    plot(mc,BW1(i,:),lineorder{i});
```

```
    hold on;

    figure(f7);
    plot(mc,maxoffdiagmag(i,:),lineorder{i});
    hold on;

    figure(f8);
    plot(mc,TMDC(i,:),lineorder{i});
    hold on;

    figure(f9);
    plot(mc, maxoffdiagfreq(i,:),lineorder{i});
    hold on;

    figure(f10);
    plot(Power2,BW1(i,:),lineorder{i});
    hold on;

    figure(f11);
    plot(Power2,TMDC(i,:),lineorder{i});
    hold on;

    figure(f12);
    plot(Power2,maxoffdiagfreq(i,:),lineorder{i});
    hold on;

    figure(f13);
    plot(pmr1(i,:),BW1(i,:),lineorder{i});
    hold on;

    figure(f15);
    plot(BW1(i,:),pmr1(i,:),lineorder{i});
    hold on;

    figure(f14);
    plot(pmr1(i,:),TMDC(i,:),lineorder{i});
    hold on;



end


leg=strcat(tmc,tcstr);

tmc2={', BW= '};
tcstr2=num2str(BW');
```

```
leg2=strcat(tmc2,tcstr2);

tmc3={', Ts= '};
tcstr3=num2str(ts');
leg3=strcat(tmc3,tcstr3);

legen=strcat(leg,leg2);


lege=strtrim(cellstr(legen));

figure(f5);
legend('Rover',lege{:});

massstr={'Mass(Kg)= '}; %adding Mass= to begining of each torque constant legend
mass1str=num2str(mass');
mass2str=strcat(massstr,mass1str);

line([0 max(pmr)],[reqbw reqbw],'color','r','LineStyle','--') % Required Bandwidth Line

 figure(f6);
 ylabel('3dB Bandwidth(rad/second)');
 xlabel('Km (N.m/Amp)');
 title('3dB Bandwidth vs torque constant');
 legend(num2str(mass'));
 grid minor;

 figure(f7);
 title('Off Diagonal Peak bode magnitude vs torque constant');
 grid minor;
 xlabel('Km (N.m/Amp)');
 ylabel('Maximum Off diagonal transient value');
 legend(num2str(mass'));

 figure(f8);
 title('Off Diagonal Transient Peak Magnitude / DC gain Vs. torque constant');
 grid minor;
 xlabel('Km (N.m/Amp)');
 ylabel('Transient Peak Magnitude / DC gain');
 legend(num2str(mass'));

 figure(f9);
```

```
title('Off Diagonal Peak Transient Frequency vs Torque
    constant');
grid minor;
xlabel('Km (N.m/Amp)');
ylabel('Peak Transient Frequency (rad/sec)');
legend(num2str(mass'));

figure(f10);
ylabel('3dB Bandwidth(rad/second)');
xlabel('Power (Watts)');
title('3dB Bandwidth vs Power');
legend(num2str(mass'));
grid minor;

figure(f11);
title('(Off Diagonal Transient Peak Magnitude / DC gain) Vs.
    Power');
grid minor;
xlabel('Power (Watts)');
ylabel('Transient Peak Magnitude / DC gain');
legend(num2str(mass'));

figure(f12);
title('Off Diagonal Peak Transient Frequency vs Power');
grid minor;
xlabel('Power (Watts)');
ylabel('Peak Transient Frequency (rad/sec)');
legend(num2str(mass'));

figure(f13);
grid minor;
ylabel('3dB Bandwidth(rad/second)');
xlabel('Power/Mass(Watts/Kg)');
title('3dB Bandwidth vs Power/Mass');
legend(num2str(mass'));

figure(f13);
ylabel('3dB Bandwidth(rad/second)');
xlabel('Power/Mass(Watts/Kg)');
title('3dB Bandwidth vs Power/Mass');
legend(num2str(mass'));
grid minor;

figure(f15);
xlabel('3dB Bandwidth(rad/second)');
ylabel('Power/Mass(Watts/Kg)');
title('Power/Mass Vs. 3dB Bandwidth');
```

```
legend(num2str(mass'));
grid minor;
xlim([0 15]);
line([0 max(Pmr1(1,:))],[reqbw reqbw],'color','r','LineStyle','--') % Required Bandwidth Line


figure(f14);
ylabel('Off Diagonal Transient Peak Magnitude / DC gain');
xlabel('Power/Mass(Watts/Kg)');
title('(Off Diagonal Transient Peak Magnitude / DC gain) Vs. Power/Mass');
legend(num2str(mass'));
grid minor;

%% CL(P), Variable Controller gain, Variable Power, OL VS CL
% Properties of the plant
clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol BWrovcl BWrovol powerrov ppmrov MLminrover rob;
close all;

mc=%motor torque constant;
mass=%system Mass;
reqbw=; %Required BW in rad/sec

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.','r-.','g--'};

gain=%proportional gains;


for i=1:length(mc)

Km=mc(i);
kb=0.0847;
La=0.64*(10^-3);
Ra=0.27;
beta=0.021;
J=0.00057892;

dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);




Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; %Power in watts
```

```
Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
end


f6=figure;
f7=figure;
f8=figure;


    for i=1:length(gain)

        C1=tf(gain(i),1);
        C2=C1;


        for j=1:length(mc)


            Km=mc(j);
            kb=0.0847;
            La=0.64*(10^-3);
            Ra=0.27;
            r=0.1;
            m=mass;
            L=0.5;
            I=m*(L^2)/6; %moment of inertia for a cube with
               width = length = L
            beta=0.021;

            pmr(j)=Power2(j)/m;

            %Transfer functions and their input output names
               in chanel 1

            h1=tf(Km,[La Ra]);
            h1.u='e1'; h1.y='tau1';

            h2=tf(1,[(r^2)*m 0]);
            h2.u='x1'; h2.y='vhat1';

            h3=tf(L^2,[2*(r^2)*I 0]);
            h3.u='x2'; h3.y='omegahat1';

            h7=tf(beta,1);
            h7.u= 'omega1'; h7.y='tauf1';

            h8=tf(kb,1);
```

```
    h8.u= 'omega1'; h8.y='vb1';

    %sumblocks in channel 1
    sum1= sumblk('e1=omegar1 - vb1');
    sum2= sumblk('c1=tau1-tauf1');
    sum3= sumblk('x1=c1+tau2');
    sum4= sumblk('x2=c1-tau2');
    sum5= sumblk('omega1 = vhat1 + omegahat1');


    %Transfer functions and their input output names
       in channel 2

    h4=tf(Km,[La Ra]);
    h4.u='e2'; h4.y='tau2';

    h6=tf(1,[(r^2)*m 0]);
    h6.u='x4'; h6.y='vhat2';

    h5=tf(L^2,[2*(r^2)*I 0]);
    h5.u='x3'; h5.y='omegahat2';

    h9=tf(beta,1);
    h9.u= 'omega2'; h9.y='tauf2';

    h10=tf(kb,1);
    h10.u= 'omega2'; h10.y='vb2';


    %sumblocks in channel 1
    sum6= sumblk('e2=omegar2 - vb2');
    sum7= sumblk('c2=tau2-tauf2');
    sum8= sumblk('x3=tau1 - c2');
    sum9= sumblk('x4=c2 + tau1');
    sum10= sumblk('omega2 = vhat2 - omegahat2');

    %connect models
    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'
       });
    ML.statename={'ia1','x2','x3','ia2','x5','x6'};

    MLmin=minreal(ML,[],0);
    MLmin.u={'omegar1n','omegar2n'};
    MLmin.y={'omega1n','omega2n'};
```

```
            Kp=append(C1,C2);
            FF=MLmin*Kp;
            robcl=feedback(FF,eye(2));

            %3dB bandwidth
            BWOL(j)=bandwidth(MLmin(1,1));
            BWCL(i,j)=bandwidth(robcl(1,1));

        end

        figure(f6);
        plot(mc,BWCL(i,:),lineorder{i});
        hold on;

        figure(f7);
        plot(Power2,BWCL(i,:),lineorder{i});
        hold on;
%
        figure(f8);
        plot(BWCL(i,:),pmr,lineorder{i});
        hold on;



    end


    figure(f6);
    plot(mc,BWOL(:),'b');
    ylabel('3dB Bandwidth(rad/second)');
    xlabel('Km');
    title('3dB Bandwidth vs Torque constant, Variable
       Proportional Controller');
    grid minor;
    tmc={'CL, kp= '};
    leg=strcat(tmc,num2str(gain'));
    legend(leg{:},'Open Loop');

    figure(f7);
    plot(Power2,BWOL(:),'b');
    ylabel('3dB Bandwidth(rad/second)');
    xlabel('Power(watts)');
    title('3dB Bandwidth vs Power, Variable Proportional
       Controller');
    grid minor;
    tmc={'CL, kp= '};
    leg=strcat(tmc,num2str(gain'));
```

```
    legend(leg{:},'Open Loop');

    figure(f8);
    plot(BWOL(:),pmr,'b');
    xlabel('3dB Bandwidth(rad/second)');
    ylabel('Power/Mass(watts/Kg)');
    title('3dB Bandwidth vs Power/Mass Ratio, Variable Proportional Controller');
    grid minor;
    tmc={'CL, kp= '};
    leg=strcat(tmc,num2str(gain'));
    legend(leg{:},'Open Loop');


%% CL(PI), Variable Controller gain, Variable Power, OL VS CL
% Properties of the plant
clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol BWrovcl BWrovol powerrov ppmrov MLminrover rob;
close all;

mc=%motor torque constant;
mass=%system Mass;
reqbw=; %Required BW in rad/sec

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.','r-.','g--'};
pgain=%proportional gain;
igain=%integral gain;

for i=1:length(mc)

Km=mc(i);
kb=0.0847;
La=0.64*(10^-3);
Ra=0.27;
beta=0.021;
J=0.00057892;
dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);
Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; %Power in watts
Power(i)=Power2(i)*(1.341*10^-3); %Power in hp
end


f4=figure;
f5=figure;
f6=figure;
```

```matlab
f7=figure;
f8=figure;
f9=figure;

    for i=1:length(pgain)

        C1=pid(pgain(i),0.1);
        C2=C1;


        for j=1:length(mc)

            Km=mc(j);
            kb=0.0847;
            La=0.64*(10^-3);
            Ra=0.27;
            r=0.1;
            m=mass;
            L=0.5;
            I=m*(L^2)/6; %moment of inertia for a cube with width = length = L
            beta=0.021;

            pmr(j)=Power2(j)/mass;

            %Transfer functions and their input output names in chanel 1

            h1=tf(Km,[La Ra]);
            h1.u='e1'; h1.y='tau1';

            h2=tf(1,[(r^2)*m 0]);
            h2.u='x1'; h2.y='vhat1';

            h3=tf(L^2,[2*(r^2)*I 0]);
            h3.u='x2'; h3.y='omegahat1';

            h7=tf(beta,1);
            h7.u= 'omega1'; h7.y='tauf1';

            h8=tf(kb,1);
            h8.u= 'omega1'; h8.y='vb1';

            %sumblocks in channel 1
            sum1= sumblk('e1=omegar1 - vb1');
            sum2= sumblk('c1=tau1-tauf1');
            sum3= sumblk('x1=c1+tau2');
```

```
sum4= sumblk('x2=c1-tau2');
sum5= sumblk('omega1 = vhat1 + omegahat1');


%Transfer functions and their input output names
   in channel 2

h4=tf(Km,[La Ra]);
h4.u='e2'; h4.y='tau2';

h6=tf(1,[(r^2)*m 0]);
h6.u='x4'; h6.y='vhat2';

h5=tf(L^2,[2*(r^2)*I 0]);
h5.u='x3'; h5.y='omegahat2';

h9=tf(beta,1);
h9.u= 'omega2'; h9.y='tauf2';

h10=tf(kb,1);
h10.u= 'omega2'; h10.y='vb2';


%sumblocks in channel 1
sum6= sumblk('e2=omegar2 - vb2');
sum7= sumblk('c2=tau2-tauf2');
sum8= sumblk('x3=tau1 - c2');
sum9= sumblk('x4=c2 + tau1');
sum10= sumblk('omega2 = vhat2 - omegahat2');

%connect models
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'
   });
ML.statename={'ia1','x2','x3','ia2','x5','x6'};

%Minimum realization Plant

MLmin=minreal(ML,[],0);
MLmin.u={'omegar1n','omegar2n'};
MLmin.y={'omega1n','omega2n'};

%Minimum Realization Plots

%StepPlot
% figure(f3);
```

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

            %Singular Value plot
            %         figure(f4);
            %         sigmaplot(MLmin,sopt,lineorder{k});
            % setoptions(f4,'FreqUnits','Hz');
            %         grid;
            %         title('Singular Values of the 2 Motor
              channels, Robot''s dynamics included');
            %         hold all;
            %
            % [Aol,Bol,Col,Dol]=ssdata(MLmin);

            %         figure(f5);
            %         bodemag(MLmin,lineorder{k});
            %         grid on;
            %         title('Frequency Response of the Open
              Loop System');
            %         hold all;


            Kp=append(C1,C2);
            FF=MLmin*Kp;
            robcl=feedback(FF,eye(2));

            %3dB bandwidth
            BWOL(j)=bandwidth(MLmin(1,1));
            BWCL(i,j)=bandwidth(robcl(1,1));

        end

        figure(f4);
        plot(Power2,BWCL(i,:),lineorder{i});
        hold on;
%
        figure(f5);
        plot(BWCL(i,:),pmr,lineorder{i});
        hold on;

        figure(f6);
        plot(mc,BWCL(i,:),lineorder{i});
        hold on;
%

```

```
    end

     figure(f4);
    plot(Power2,BWOL(:),'b');
    ylabel('3dB Bandwidth(rad/second)');
    xlabel('Power(watts)');
    title('3dB Bandwidth vs Power, PI controller W/ Variable
       Proportional Gain');
    grid minor;
    tmc={'CL, kp= '};
    leg=strcat(tmc,num2str(pgain'));
    legend(leg{:},'Open Loop');

    figure(f5);
    plot(BWOL(:),pmr,'b');
    xlabel('3dB Bandwidth(rad/second)');
    ylabel('Power/Mass(watts/Kg)');
    title('3dB Bandwidth vs Power/Mass Ratio,PI controller W/
        Variable Proportional Gain');
    grid minor;
    tmc={'CL, kp= '};
    leg=strcat(tmc,num2str(pgain'));
    legend(leg{:},'Open Loop');

    figure(f6);
    plot(mc,BWOL(:),'b');
    ylabel('3dB Bandwidth(rad/second)');
    xlabel('Km');
    title('3dB Bandwidth vs Torque constant,PI controller W/
       Variable Proportional Gain');
    grid minor;
    tmc={'CL, kp= '};
    leg=strcat(tmc,num2str(pgain'));
    legend(leg{:},'Open Loop');


    for i=1:length(igain)

        C1=pid(0.1,igain(i));
        C2=C1;


        for j=1:length(mc)

            Km=mc(j);
```

```
            kb=0.0847;
            La=0.64*(10^-3);
            Ra=0.27;
            r=0.1;
            m=mass;
            L=0.5;
            I=m*(L^2)/6; %moment of inertia for a cube with
               width = length = L
            beta=0.021;


            %Transfer functions and their input output names
               in chanel 1

            h1=tf(Km,[La Ra]);
            h1.u='e1'; h1.y='tau1';

            h2=tf(1,[(r^2)*m 0]);
            h2.u='x1'; h2.y='vhat1';

            h3=tf(L^2,[2*(r^2)*I 0]);
            h3.u='x2'; h3.y='omegahat1';

            h7=tf(beta,1);
            h7.u= 'omega1'; h7.y='tauf1';

            h8=tf(kb,1);
            h8.u= 'omega1'; h8.y='vb1';

            %sumblocks in channel 1
            sum1= sumblk('e1=omegar1 - vb1');
            sum2= sumblk('c1=tau1-tauf1');
            sum3= sumblk('x1=c1+tau2');
            sum4= sumblk('x2=c1-tau2');
            sum5= sumblk('omega1 = vhat1 + omegahat1');


            %Transfer functions and their input output names
               in channel 2

            h4=tf(Km,[La Ra]);
            h4.u='e2'; h4.y='tau2';

            h6=tf(1,[(r^2)*m 0]);
            h6.u='x4'; h6.y='vhat2';

            h5=tf(L^2,[2*(r^2)*I 0]);
```

```
            h5.u='x3'; h5.y='omegahat2';

            h9=tf(beta,1);
            h9.u= 'omega2'; h9.y='tauf2';

            h10=tf(kb,1);
            h10.u= 'omega2'; h10.y='vb2';


            %sumblocks in channel 1
            sum6= sumblk('e2=omegar2 - vb2');
            sum7= sumblk('c2=tau2-tauf2');
            sum8= sumblk('x3=tau1 - c2');
            sum9= sumblk('x4=c2 + tau1');
            sum10= sumblk('omega2 = vhat2 - omegahat2');

            %connect models
            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'
                });
            ML.statename={'ia1','x2','x3','ia2','x5','x6'};

            MLmin=minreal(ML,[],0);
            MLmin.u={'omegar1n','omegar2n'};
            MLmin.y={'omega1n','omega2n'};

            Kp=append(C1,C2);
            FF=MLmin*Kp;
            robcl=feedback(FF,eye(2));

            %3dB bandwidth
            BWOL(j)=bandwidth(MLmin(1,1));
            BWCL(i,j)=bandwidth(robcl(1,1));

        end

        figure(f7);
        plot(mc,BWCL(i,:),lineorder{i});
        hold on;

        figure(f8);
        plot(Power2,BWCL(i,:),lineorder{i});
        hold on;
%
        figure(f9);
        plot(BWCL(i,:),pmr,lineorder{i});
```

```
        hold on;


    end

    figure(f7);
    plot(mc,BWOL(:),'b');
    ylabel('3dB Bandwidth(rad/second)');
    xlabel('Km');
    title('3dB Bandwidth vs Torque constant,PI controller W/
       Variable Integral Gain');
    grid minor;
    tmc={'CL, ki= '};
    leg=strcat(tmc,num2str(igain'));
    legend(leg{:},'Open Loop');

    figure(f8);
    plot(Power2,BWOL(:),'b');
    ylabel('3dB Bandwidth(rad/second)');
    xlabel('Power(watts)');
    title('3dB Bandwidth vs Power, PI controller W/ Variable
       Integral Gain');
    grid minor;
    tmc={'CL, ki= '};
    leg=strcat(tmc,num2str(pgain'));
    legend(leg{:},'Open Loop');

    figure(f9);
    plot(BWOL(:),pmr,'b');
    xlabel('3dB Bandwidth(rad/second)');
    ylabel('Power/Mass(watts/Kg)');
    title('3dB Bandwidth vs Power/Mass Ratio,PI controller W/
        Variable Integral Gain');
    grid minor;
    tmc={'CL, ki= '};
    leg=strcat(tmc,num2str(pgain'));
    legend(leg{:},'Open Loop');



%% CL, Sensitivity (P)
% Properties of the plant

clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
   BWrovcl BWrovol powerrov ppmrov MLminrover rob;
close all;

```

```
mc=%motor torque constant;
mass=%system Mass;
reqbw=; %Required BW in rad/sec

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.',
    'r-.','g--'};
lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.',
    'r-.','g--'};


f2=figure;
f3=figure;
f4=figure;
f5=figure;



    for g=1:length(gain)

        C1=tf(gain(g),1);
        C2=tf(gain(g),1);

            Km=mc;
            kb=0.0847;
            La=0.64*(10^-3);
            Ra=0.27;
            r=0.1;
            m=mass;
            L=0.5;
            I=m*(L^2)/6; %moment of inertia for a cube with
                width = length = L
            beta=0.021;


            %Transfer functions and their input output names
                in chanel 1

            h1=tf(Km,[La Ra]);
            h1.u='e1'; h1.y='tau1';

            h2=tf(1,[(r^2)*m 0]);
            h2.u='x1'; h2.y='vhat1';

            h3=tf(L^2,[2*(r^2)*I 0]);
            h3.u='x2'; h3.y='omegahat1';

            h7=tf(beta,1);
```

```
h7.u= 'omega1'; h7.y='tauf1';

h8=tf(kb,1);
h8.u= 'omega1'; h8.y='vb1';

%sumblocks in channel 1
sum1= sumblk('e1=omegar1 - vb1');
sum2= sumblk('c1=tau1-tauf1');
sum3= sumblk('x1=c1+tau2');
sum4= sumblk('x2=c1-tau2');
sum5= sumblk('omega1 = vhat1 + omegahat1');


%Transfer functions and their input output names
   in channel 2

h4=tf(Km,[La Ra]);
h4.u='e2'; h4.y='tau2';

h6=tf(1,[(r^2)*m 0]);
h6.u='x4'; h6.y='vhat2';

h5=tf(L^2,[2*(r^2)*I 0]);
h5.u='x3'; h5.y='omegahat2';

h9=tf(beta,1);
h9.u= 'omega2'; h9.y='tauf2';

h10=tf(kb,1);
h10.u= 'omega2'; h10.y='vb2';


%sumblocks in channel 1
sum6= sumblk('e2=omegar2 - vb2');
sum7= sumblk('c2=tau2-tauf2');
sum8= sumblk('x3=tau1 - c2');
sum9= sumblk('x4=c2 + tau1');
sum10= sumblk('omega2 = vhat2 - omegahat2');

%connect models
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'
    });
ML.statename={'ia1','x2','x3','ia2','x5','x6'};

MLmin=minreal(ML,[],0);
```

```
            MLmin.u={'omegar1n','omegar2n'};
            MLmin.y={'omega1n','omega2n'};

            Kp=append(C1,C2);
            FF=MLmin*Kp;
            robcl=feedback(FF,eye(2));

            Loopspid=loopsens(FF,eye(2));

            figure(f2);
            bodemag(Loopspid.Si,lineorder{g});
            title('Sensitivity Bode Magnitude with P controller, variable K1, fixed K2');
            grid minor;
            hold all;

            figure(f3);
            bodemag(Loopspid.Ti,lineorder{g});
            title('Complement Sensitivity Bode Magnitude with P controller, variable K1, fixed K2');
            grid minor;
            hold all;

            figure(f5)
            bodemag(robcl,lineorder{g});
            title('Bode magnitude of the close loops system');
            grid minor;
            hold all;

            %3dB bandwidth
            BWCL(g)=bandwidth(robcl(1,1));

    end

tmc={'Kp= '};
leg=strcat(tmc,num2str(gain'));

figure(f2);
legend(leg{:});

figure(f3);
legend(leg{:})

figure(f5);
legend(leg{:})

```

```
figure(f4);
plot(gain,BWCL);
title('3dB Bandwidth Vs. Controller gain');
xlabel('Controller Gain');
ylabel('3dB Bandwidth');

%% CL, Sensitivity (P VS PI)
% Properties of the plant

clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
    BWrovcl BWrovol powerrov ppmrov MLminrover rob;
close all;

mc=;
mass=;
gain=;
Km=mc;
kb=;
La=;
Ra=;
r=;
m=mass;
L=;
I=m*(L^2)/6; %moment of inertia for a cube with width =
    length = L
beta=;
%Transfer functions and their input output names in chanel 1

h1=tf(Km,[La Ra]);
h1.u='e1'; h1.y='tau1';

h2=tf(1,[(r^2)*m 0]);
h2.u='x1'; h2.y='vhat1';

h3=tf(L^2,[2*(r^2)*I 0]);
h3.u='x2'; h3.y='omegahat1';

h7=tf(beta,1);
h7.u= 'omega1'; h7.y='tauf1';

h8=tf(kb,1);
h8.u= 'omega1'; h8.y='vb1';

%sumblocks in channel 1
sum1= sumblk('e1=omegar1 - vb1');
sum2= sumblk('c1=tau1-tauf1');
sum3= sumblk('x1=c1+tau2');
```

```
sum4= sumblk('x2=c1-tau2');
sum5= sumblk('omega1 = vhat1 + omegahat1');


%Transfer functions and their input output names in channel 2

h4=tf(Km,[La Ra]);
h4.u='e2'; h4.y='tau2';

h6=tf(1,[(r^2)*m 0]);
h6.u='x4'; h6.y='vhat2';

h5=tf(L^2,[2*(r^2)*I 0]);
h5.u='x3'; h5.y='omegahat2';

h9=tf(beta,1);
h9.u= 'omega2'; h9.y='tauf2';

h10=tf(kb,1);
h10.u= 'omega2'; h10.y='vb2';


%sumblocks in channel 1
sum6= sumblk('e2=omegar2 - vb2');
sum7= sumblk('c2=tau2-tauf2');
sum8= sumblk('x3=tau1 - c2');
sum9= sumblk('x4=c2 + tau1');
sum10= sumblk('omega2 = vhat2 - omegahat2');

%connect models
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'});
ML.statename={'ia1','x2','x3','ia2','x5','x6'};

%Minimum realization Plant

MLmin=minreal(ML,[],0);
MLmin.u={'omegar1n','omegar2n'};
MLmin.y={'omega1n','omega2n'};


lineorder={'b','g','r','c','m','k'};
lineorder2={'b--','g--','r--','c--','m--','k--'};


f2=figure;
```

```
f3=figure;
f4=figure;
f5=figure;



    for g=1:length(gain)

        C1=tf(gain(g),1);
        C2=C1;
        CPI1=tf(gain(g),[1 0]);
        CPI2=CPI1;

            Kp=append(C1,C2);
            FF=MLmin*Kp;
            robcl=feedback(FF,eye(2));

            Loopspid=loopsens(FF,eye(2));

            Kpi=append(CPI1,CPI2);
            FFpi=MLmin*Kpi;
            robclpi=feedback(FFpi,eye(2));

            Loopspi=loopsens(FFpi,eye(2));

            figure(f2);
            bodemag(Loopspid.Si,lineorder{g});
            hold all;
            bodemag(Loopspi.Si,lineorder2{g});
            title('Sensitivity Bode Magnitude with P
               controller, variable K1, fixed K2');
            grid minor;
            hold all;

            figure(f3);
            bodemag(Loopspid.Ti,lineorder{g});
            hold all;
            bodemag(Loopspi.Ti,lineorder2{g});
            title('Complement Sensitivity Bode Magnitude with
                P controller, variable K1, fixed K2');
            grid minor;
            hold all;

            figure(f5)
            bodemag(robcl,lineorder{g});
            hold all;
            bodemag(robclpi,lineorder2{g});
```

```
            title('Bode magnitude of the close loops system');
            grid minor;
            hold all;

            %3dB bandwidth
            BWCL(g)=bandwidth(robcl(1,1));
            BWCLPi(g)=bandwidth(robclpi(1,1));

    end

tmc={'Kp= '};
leg=strcat(tmc,num2str(gain'));

figure(f2);
legend(leg{:});

figure(f3);
legend(leg{:})

figure(f5);
legend(leg{:})

figure(f4);
plot(gain,BWCL);
hold on;
plot(gain,BWCLPi,'r');
title('3dB Bandwidth Vs. Controller gain');
xlabel('Controller Gain');
ylabel('3dB Bandwidth');
%% OPEN LOOP POWER + MASS PLOTS
% Properties of the plant
clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol BWrovcl BWrovol powerrov ppmrov MLminrover rob;
close all;

mc=%motor torque constant;
mass=%system Mass;
reqbw=; %Required BW in rad/sec

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.','r-.','g--'};
reqts=reqbw/5;
reqrat=10; %Required diagonal/offdiagonal ratio

%gray area calculation

```

```
tenpoffbw=0.9*reqbw;
tenpoffrat=0.9*reqrat;
tenpoffts=1.1*reqts;


for i=1:length(mc)

Km=mc(i);
kb=0.0847;
La=0.64*(10^-3);
Ra=0.27;
beta=0.021;
J=0.00057892;


h1= tf(Km,[La,Ra]);
h1.u='e'; h1.y='tau';

h2= tf(1,[J,beta]);
h2.u='tau'; h2.y='omega';

h3= tf(kb,1);
h3.u='omega'; h3.y='vb';

sum1=sumblk('e=v-vb');

dcm=connect(ss(h1),h2,h3,sum1,'v',{'tau','omega'});

Power2(i)=(24^2)*beta*((Km/(beta*Ra+Km*kb))^2) ; %Power in watts
Power(i)=Power2(i)*(1.341*10^-3); %Power in hp

end

% f3=figure;
% f4=figure;
f5=figure;
f6=figure;
% f7=figure;
% f8=figure;
% f9=figure;


k=1;


for i=1:length(mass)
```

```
    for j=1:length(mc)

        Km=mc(j);
        kb=0.0847;
        La=0.64*(10^-3);
        Ra=0.27;
        r=0.1;
        m=mass(i);
        L=0.5;
        I=m*(L^2)/6; %moment of inertia for a cube with width = length = L
        beta=0.021;

        pmr(k)=Power(j)/mass(i); %Computing Power to Mass ratio


        %Transfer functions and their input output names in chanel 1

        h1=tf(Km,[La Ra]);
        h1.u='e1'; h1.y='tau1';

        h2=tf(1,[(r^2)*m 0]);
        h2.u='x1'; h2.y='vhat1';

        h3=tf(L^2,[2*(r^2)*I 0]);
        h3.u='x2'; h3.y='omegahat1';

        h7=tf(beta,1);
        h7.u= 'omega1'; h7.y='tauf1';

        h8=tf(kb,1);
        h8.u= 'omega1'; h8.y='vb1';

        %sumblocks in channel 1
        sum1= sumblk('e1=omegar1 - vb1');
        sum2= sumblk('c1=tau1-tauf1');
        sum3= sumblk('x1=c1+tau2');
        sum4= sumblk('x2=c1-tau2');
        sum5= sumblk('omega1 = vhat1 + omegahat1');


        %Transfer functions and their input output names in channel 2

```

```
        h4=tf(Km,[La Ra]);
        h4.u='e2'; h4.y='tau2';

        h6=tf(1,[(r^2)*m 0]);
        h6.u='x4'; h6.y='vhat2';

        h5=tf(L^2,[2*(r^2)*I 0]);
        h5.u='x3'; h5.y='omegahat2';

        h9=tf(beta,1);
        h9.u= 'omega2'; h9.y='tauf2';

        h10=tf(kb,1);
        h10.u= 'omega2'; h10.y='vb2';


        %sumblocks in channel 1
        sum6= sumblk('e2=omegar2 - vb2');
        sum7= sumblk('c2=tau2-tauf2');
        sum8= sumblk('x3=tau1 - c2');
        sum9= sumblk('x4=c2 + tau1');
        sum10= sumblk('omega2 = vhat2 - omegahat2');

        %connect models
        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'});
        ML.statename={'ia1','x2','x3','ia2','x5','x6'};

        %Minimum realization Plant

        MLmin=minreal(ML,[],0);
        MLmin.u={'omegar1n','omegar2n'};
        MLmin.y={'omega1n','omega2n'};

        BW(i,j)=bandwidth(MLmin(1,1)); % Motor Bandwidth

        S=stepinfo(MLmin(1,1));
        ts(i,j)=S.SettlingTime;

        k=k+1;

    end

    figure(f5);
    plot(Power,ts(i,:),lineorder{i});
    hold on;
```

```
    figure(f6);
    plot(Power,BW(i,:),lineorder{i});
    hold on;

end

tmc={'Mass(Kg)= '};
tcstr=num2str(mass');
leg=strcat(tmc,tcstr);
lege=strtrim(cellstr(leg));

figure(f5);
ylabel('Settling Time (seconds)');
xlabel('Power (hp)');
title('Settling time Vs. Power for Open Loop systems with different Masses');
grid on;

line([0 max(Power)],[reqts reqts],'color','r','LineStyle','--') % Required Bandwidth Line
line([0 max(Power)],[tenpoffts tenpoffts],'color',[0.5 0.5 0.5],'LineStyle','--') %10% off Bandwidth Line
plot(powerrov,tsrovol,'rO','MarkerFaceColor','r') % Rover Specification


legend(lege{:},'Minimum Design Goal','10% Off Design Goal','Rover');


figure(f6);
ylabel('Bandwidth (radian/seconds)');
xlabel('Power (hp)');
title('Sysem Bandwidth Vs. Power for Open Loop systems with different Masses');
grid on;

line([0 max(Power)],[reqbw reqbw],'color','r','LineStyle','--') % Required Bandwidth Line
line([0 max(Power)],[tenpoffbw tenpoffbw],'color',[0.5 0.5 0.5],'LineStyle','--') %10% off Bandwidth Line
plot(powerrov,BWrovol,'rO','MarkerFaceColor','r') % Rover Specification
```

```
legend(lege{:},'Minimum Design Goal','10% Off Design Goal','
    Rover');

%% Closed Loop , POWER/MASS Plots

% Properties of the plant
clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
    BWrovcl BWrovol powerrov ppmrov MLminrover rob;
close all;

mc=%motor torque constant;
mass=%system Mass;
reqbw=; %Required BW in rad/sec

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.',
    'r-.','g--'};

% Power Calculation

for i=1:length(mc)

Km=mc(i);
kb=0.0487;
La=0.64*(10^-3);
Ra=0.27;
beta=0.021;
J=0.00057892;


h1= tf(Km,[La,Ra]);
h1.u='e'; h1.y='tau';

h2= tf(1,[J,beta]);
h2.u='tau'; h2.y='omega';

h3= tf(kb,1);
h3.u='omega'; h3.y='vb';

sum1=sumblk('e=v-vb');

dcm=connect(ss(h1),h2,h3,sum1,'v',{'tau','omega'});


t=0:1:24;
u=t;
[y,t]= lsim(dcm,u,t);

```

```
Power(i)=y(24,1)*y(24,2)/746;

end
f5=figure;

%rover bode
bodemag(rob,'k-');
h=findobj(gcf,'type','line');
set(h,'linewidth',1.2);
hold on;


loops=loopsens(MLminrover,eye(2));

f3=figure;

bodemag(loops.Si,'k-');
h=findobj(gcf,'type','line');
set(h,'linewidth',1.2);
hold on;

f4=figure;

bodemag(loops.Ti,'k-');
h=findobj(gcf,'type','line');
set(h,'linewidth',1.2);
hold on;

% f6=figure;
% f7=figure;
% f8=figure;
% f9=figure;


k=1;


for i=1:length(mc)

    for j=1:length(mass)

        Km=mc(i);
        kb=0.0487;
        La=0.64*(10^-3);
        Ra=0.27;
        r=0.1;
        m=mass(j);
```

```
        L=0.5;
        I=m*(L^2)/6; %moment of inertia for a cube with width
            = length = L
        beta=0.021;

        pmr(k)=Power(i)/mass(j); %Computing Power to Mass
           ratio


        %Transfer functions and their input output names in
           chanel 1

        h1=tf(Km,[La Ra]);
        h1.u='e1'; h1.y='tau1';

        h2=tf(1,[(r^2)*m 0]);
        h2.u='x1'; h2.y='vhat1';

        h3=tf(L^2,[2*(r^2)*I 0]);
        h3.u='x2'; h3.y='omegahat1';

        h7=tf(beta,1);
        h7.u= 'omega1'; h7.y='tauf1';

        h8=tf(kb,1);
        h8.u= 'omega1'; h8.y='vb1';

        %sumblocks in channel 1
        sum1= sumblk('e1=omegar1 - vb1');
        sum2= sumblk('c1=tau1-tauf1');
        sum3= sumblk('x1=c1+tau2');
        sum4= sumblk('x2=c1-tau2');
        sum5= sumblk('omega1 = vhat1 + omegahat1');


        %Transfer functions and their input output names in
           channel 2

        h4=tf(Km,[La Ra]);
        h4.u='e2'; h4.y='tau2';

        h6=tf(1,[(r^2)*m 0]);
        h6.u='x4'; h6.y='vhat2';

        h5=tf(L^2,[2*(r^2)*I 0]);
        h5.u='x3'; h5.y='omegahat2';

```

```
        h9=tf(beta,1);
        h9.u= 'omega2'; h9.y='tauf2';

        h10=tf(kb,1);
        h10.u= 'omega2'; h10.y='vb2';


        %sumblocks in channel 1
        sum6= sumblk('e2=omegar2 - vb2');
        sum7= sumblk('c2=tau2-tauf2');
        sum8= sumblk('x3=tau1 - c2');
        sum9= sumblk('x4=c2 + tau1');
        sum10= sumblk('omega2 = vhat2 - omegahat2');

        %connect models
        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'});
        ML.statename={'ia1','x2','x3','ia2','x5','x6'};

        %Minimum realization Plant

        MLmin=minreal(ML,[],0);
        MLmin.u={'omegar1n','omegar2n'};
        MLmin.y={'omega1n','omega2n'};

        %Minimum Realization Plots

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

        %Singular Value plot
%           figure(f4);
%           sigmaplot(MLmin,sopt,lineorder{k});
        % setoptions(f4,'FreqUnits','Hz');
%           grid;
%           title('Singular Values of the 2 Motor channels,
    Robot''s dynamics included');
%           hold all;

        robcl=feedback(MLmin,eye(2));

        figure(f5);
```

```
        bodemag(robcl,lineorder{k});
        grid on;
        title('Frequency Response of the Closed Loop System');
        hold all;

        BW(k)=bandwidth(robcl(1,1)); % Motor Bandwidth

        %evaluating the response at 0 rad/sec
        mag0=bode(robcl,0);
        magrat0(k)=mag0(1,1)/mag0(1,2);

        %evaluating the response at OmegaBW

        magbw=bode(robcl,reqbw);
        magratbw(k)=magbw(1,1)/magbw(1,2);

        S=stepinfo(robcl(1,1));
        ts(k)=S.SettlingTime;

        %Sensetivity plots
        loops=loopsens(MLmin,eye(2));

        figure(f3);
        bodemag(loops.Si,lineorder{k});
        title('Sensetivity Magnitude Closed loop System with K=I, Variable Power/Mass');
        grid on;
        hold all;

        figure(f4);
        bodemag(loops.Ti,lineorder{k});
        title('Complement Magnitude Closed loop System with no K=I, Variable Power/Mass');
        grid on;
        hold all;

        k=k+1;


    end
end

tmc={'Power/Mass(hp/Kg) = '}; %adding Mass= to begining of each torque constant legend
tcstr=num2str(pmr');

```

```
leg=strcat(tmc,tcstr);

tmc2={', BW= '};
tcstr2=num2str(BW');
leg2=strcat(tmc2,tcstr2);

% tmc3={', Ts= '};
% tcstr3=num2str(ts');
% leg3=strcat(tmc3,tcstr3);

legen=strcat(leg,leg2);


lege=strtrim(cellstr(legen));


figure(f3);
legend('Rover',lege{:});

figure(f4);
legend('Rover',lege{:});

figure(f5);
legend('Rover',lege{:});

figure(f6);
plot(pmr,ts);
ylabel('Settling Time (Seconds)');
xlabel('Power per Kg (hp/Kg)');
title('Settling time vs Power to Mass ratio plot');

figure(f7);
plot(pmr,BW);
ylabel('System Bandwidth (rad/sec)');
xlabel('Power per Kg (hp/Kg)');
title('Badnwidth vs Power to Mass ratio plot');

figure(f8);
plot(pmr,magrat0);
ylabel('diagonal DC gain / off diagonal DC gain');
xlabel('Power per Kg (hp/Kg)');
title('diagonal to off diagonal dc gain ratio vs Power to
   Mass ratio plot');

figure(f9);
plot(pmr,magratbw);
ylabel('diagonal amplitude / off diagonal amplitude');
```

```
xlabel('Power per Kg (hp/Kg)');
title('diagonal to off diagonal amplitude ratio @ bandwidth
   frequency vs Power to Mass ratio plot');

%% OL VS CL
% Properties of the plant
clearvars -EXCEPT tsrovcl tsrovol magrat0rovcl magrat0rovol
   BWrovcl BWrovol powerrov ppmrov;
close all;

mc=%motor torque constant;
mass=%system Mass;
reqbw=; %Required BW in rad/sec

lineorder={'b','g','r','c','m','k-.','b--','r--','k--','b-.',
   'r-.','g--'};
reqts=reqbw/5;

reqrat=10; %Required diagonal/offdiagonal ratio

%gray area calculation

tenpoffbw=0.9*reqbw;
tenpoffrat=0.9*reqrat;
tenpoffts=1.1*reqts;


lineorder={'b','g','r','c','m','y','k','b--','r--','k--','b-.
   ','r-.','g--'};

Kminit=0.0487;

% Power Calculation @ 24 V

for i=1:length(mc)

Km=mc(i);
kb=0.0847;
La=0.64*(10^-3);
Ra=0.27;
beta=0.021;
J=0.00057892;

dcm=tf(Km,[J*La J*Ra+beta*La beta*Ra+Km*kb]);

vin=24; %input voltage
Ts=(Km/Ra)*vin; %Stall Torque
```

```
omega0=vin/kb; %No load speed
Power2(i)=(Ts*omega0)/4;
Power(i)=Power2(i)*(1.341*10^-3); %Power in hp

end


f6=figure;
f7=figure;
f8=figure;
f9=figure;


k=1;


for i=1:length(mc)

    for j=1:length(mass)

        Km=mc(i);
        kb=0.0847;
        La=0.64*(10^-3);
        Ra=0.27;
        r=0.1;
        m=mass(j);
        L=0.5;
        I=m*(L^2)/6; %moment of inertia for a cube with width = length = L
        beta=0.021;

        pmr(k)=Power(i)/mass(j); %Computing Power to Mass ratio


        %Transfer functions and their input output names in chanel 1

        h1=tf(Km,[La Ra]);
        h1.u='e1'; h1.y='tau1';

        h2=tf(1,[(r^2)*m 0]);
        h2.u='x1'; h2.y='vhat1';

        h3=tf(L^2,[2*(r^2)*I 0]);
        h3.u='x2'; h3.y='omegahat1';

```

```
        h7=tf(beta,1);
        h7.u= 'omega1'; h7.y='tauf1';

        h8=tf(kb,1);
        h8.u= 'omega1'; h8.y='vb1';

        %sumblocks in channel 1
        sum1= sumblk('e1=omegar1 - vb1');
        sum2= sumblk('c1=tau1-tauf1');
        sum3= sumblk('x1=c1+tau2');
        sum4= sumblk('x2=c1-tau2');
        sum5= sumblk('omega1 = vhat1 + omegahat1');


        %Transfer functions and their input output names in
           channel 2

        h4=tf(Km,[La Ra]);
        h4.u='e2'; h4.y='tau2';

        h6=tf(1,[(r^2)*m 0]);
        h6.u='x4'; h6.y='vhat2';

        h5=tf(L^2,[2*(r^2)*I 0]);
        h5.u='x3'; h5.y='omegahat2';

        h9=tf(beta,1);
        h9.u= 'omega2'; h9.y='tauf2';

        h10=tf(kb,1);
        h10.u= 'omega2'; h10.y='vb2';


        %sumblocks in channel 1
        sum6= sumblk('e2=omegar2 - vb2');
        sum7= sumblk('c2=tau2-tauf2');
        sum8= sumblk('x3=tau1 - c2');
        sum9= sumblk('x4=c2 + tau1');
        sum10= sumblk('omega2 = vhat2 - omegahat2');

        %connect models
        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'});
        ML.statename={'ia1','x2','x3','ia2','x5','x6'};

        %Minimum realization Plant
```

```
            MLmin=minreal(ML,[],0);
            MLmin.u={'omegar1n','omegar2n'};
            MLmin.y={'omega1n','omega2n'};

            robcl=feedback(MLmin,eye(2));

            BWol(k)=bandwidth(MLmin(1,1)); % System Bandwidth

            BWcl(k)=bandwidth(robcl(1,1)); % System Bandwidth

            %evaluating the response at 0 rad/sec
            mag0ol=bode(MLmin,0);
            magrat0ol(k)=mag0ol(1,1)/mag0ol(1,2);

            mag0cl=bode(robcl,0);
            magrat0cl(k)=mag0cl(1,1)/mag0cl(1,2);

            %evaluating the response at OmegaBW
            magbwol=bode(MLmin,reqbw);
            magratbwol(k)=magbwol(1,1)/magbwol(1,2);

            magbwcl=bode(robcl,reqbw);
            magratbwcl(k)=magbwcl(1,1)/magbwcl(1,2);

            Sol=stepinfo(MLmin(1,1));
            tsol(k)=Sol.SettlingTime;

            Scl=stepinfo(robcl(1,1));
            tscl(k)=Scl.SettlingTime;


            k=k+1;


        end
end



figure(f6);
plot(pmr,tsol);
ylabel('Settling Time (Seconds)');
xlabel('Power per Kg (hp/Kg)');
title('Settling time vs Power to Mass ratio plot');
hold on;
plot(pmr,tscl,'g');
```

```
grid on;

%Settling Time
pmtscl=interp1(tscl,pmr,reqts);
pmtsol=interp1(tsol,pmr,reqts);

pmtscl2=interp1(tscl,pmr,tenpoffts);
pmtsol2=interp1(tsol,pmr,tenpoffts);

line([0 max(pmr)],[reqts reqts],'color','r','LineStyle','--') % Required Bandwidth Line

line([0 max(pmr)],[tenpoffts tenpoffts],'color',[0.5 0.5 0.5],'LineStyle','--') %10% off Bandwidth Line

plot(ppmrov,tsrovcl,'rO','MarkerFaceColor','r') % Rover Specification

plot(ppmrov,tsrovol,'rO','MarkerFaceColor','r') % Rover Specification


if ~isnan(pmtsol)
line([pmtsol pmtsol],[0 reqts],'color','r','LineStyle','--');
end

if ~isnan(pmtscl)
line([pmtscl pmtscl],[0 reqts],'color','r','LineStyle','--');
end

if ~isnan(pmtsol2)
line([pmtsol2 pmtsol2],[0 tenpoffts],'color',[0.5 0.5 0.5],'LineStyle','--');
end

if ~isnan(pmtscl2)
line([pmtscl2 pmtscl2],[0 tenpoffts],'color',[0.5 0.5 0.5],'LineStyle','--');
end

legend('Open Loop System','Closed Loop System','Minimum Design Goal','10% Off Design Goal','Rover');


figure(f7);
plot(pmr,BWol);
```

```
ylabel('System Bandwidth (rad/sec)');
xlabel('Power per Kg (hp/Kg)');
title('Badnwidth vs Power to Mass ratio plot');
hold on;
plot(pmr,BWcl,'g');
grid on;

%Bandwidth
pmcl=interp1(BWcl,pmr,reqbw);
pmol=interp1(BWol,pmr,reqbw);

pmcl2=interp1(BWcl,pmr,tenpoffbw);
pmol2=interp1(BWol,pmr,tenpoffbw);

line([0 max(pmr)],[reqbw reqbw],'color','r','LineStyle','--') % Required Bandwidth Line
line([0 max(pmr)],[tenpoffbw tenpoffbw],'color',[0.5 0.5 0.5],'LineStyle','--') %10% off Bandwidth Line


plot(ppmrov,BWrovcl,'rO','MarkerFaceColor','r') % Rover Specification
plot(ppmrov,BWrovol,'rO','MarkerFaceColor','r') % Rover Specification

if ~isnan(pmol)
line([pmol pmol],[0 reqbw],'color','r','LineStyle','--');
end

if ~isnan(pmcl)
line([pmcl pmcl],[0 reqbw],'color','r','LineStyle','--');
end

if ~isnan(pmol2)
line([pmol2 pmol2],[0 tenpoffbw],'color',[0.5 0.5 0.5],'LineStyle','--');
end

if ~isnan(pmcl2)
line([pmcl2 pmcl2],[0 tenpoffbw],'color',[0.5 0.5 0.5],'LineStyle','--');
end

legend('Open Loop System','Closed Loop System','Minimum Design Goal','10% Off Design Goal','Rover');

%Diagonal/Off Diagonal
```

```
figure(f8);
plot(pmr,magrat0ol);
ylabel('diagonal DC gain / off diagonal DC gain');
xlabel('Power per Kg (hp/Kg)');
title('diagonal to off diagonal dc gain ratio vs Power to
    Mass ratio plot');
hold on;
plot(pmr,magrat0cl,'g');
grid on;

%interpolate data
pmratcl=interp1(magrat0cl,pmr,reqrat);
pmratol=interp1(magrat0ol,pmr,reqrat);

pmratcl2=interp1(magrat0cl,pmr,tenpoffrat);
pmratol2=interp1(magrat0ol,pmr,tenpoffrat);

line([0 max(pmr)],[reqrat reqrat],'color','r','LineStyle','--
    ') % Required Bandwidth Line
line([0 max(pmr)],[tenpoffrat tenpoffrat],'color',[0.5 0.5
    0.5],'LineStyle','--') % 10% off Bandwidth Line

plot(ppmrov,magrat0rovol,'rO','MarkerFaceColor','r') % Rover
    Specification

plot(ppmrov,magrat0rovcl,'rO','MarkerFaceColor','r') % Rover
    Specification

if ~isnan(pmratol)
line([pmratol pmratol],[0 reqrat],'color','r','LineStyle','--
    ');
end

if ~isnan(pmratcl)
line([pmratcl pmratcl],[0 reqrat],'color','r','LineStyle','--
    ');
end

if ~isnan(pmratol2)
line([pmratol2 pmratol2],[0 tenpoffrat],'color',[0.5 0.5
    0.5],'LineStyle','--');
end

if ~isnan(pmratcl2)
line([pmratcl2 pmratcl2],[0 tenpoffrat],'color',[0.5 0.5
    0.5],'LineStyle','--');
```

```
end
legend('Open Loop System','Closed Loop System','Minimum Design Goal','10% Off Design Goal','Rover');


figure(f9);
plot(pmr,magratbwol);
ylabel('diagonal amplitude / off diagonal amplitude');
xlabel('Power per Kg (hp/Kg)');
title('diagonal to off diagonal amplitude ratio @ bandwidth frequency vs Power to Mass ratio plot');
hold on;
plot(pmr,magratbwcl,'g');
legend('Open Loop System','Closed Loop System');

%interpolate data
pmratbwcl=interp1(magratbwcl,pmr,reqrat,'pchip');
pmratbwol=interp1(magratbwol,pmr,reqrat,'pchip');

line([0 max(pmr)],[reqrat reqrat],'color','r','LineStyle','--') % Required Bandwidth Line
line([pmratbwol pmratbwol],[0 reqrat],'color','r','LineStyle','--');
line([pmratbwcl pmratbwcl],[0 reqrat],'color','r','LineStyle','--');
```