INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI
A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700 800/521-0600
Inverse Kinematics and Dynamic Control Methods for Robotic Systems

by

Arati S. Deo

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree

Doctor of Philosophy

Approved, Thesis Committee:

Dr. Ian D. Walker, Chairman
Assistant Professor
Electrical and Computer Engineering

Joseph R. Cavallaro
Associate Professor
Electrical and Computer Engineering

Dr. Dar C. Sorensen
Professor
Computational and Applied Mathematics

Houston, Texas
February, 1995
ABSTRACT

Inverse Kinematics and Dynamic Control Methods for Robotic Systems

by

Arati S. Deo

This dissertation presents new algorithms for inverse kinematic computations of robotic manipulators and for the control of multiple cooperating manipulator systems. The results presented in this thesis can be classified into three parts. The first part is an extension of our earlier work in computing inverse kinematic solutions using the damped least squares method. An adaptive algorithm is presented, which switches from the damped least-squares model to a second-order model, in situations where the former is unable to converge to the desired configuration. This algorithm is insensitive to the reachability of the desired end-effector position. The second part introduces minimum-effort inverse kinematics for redundant robotic manipulators. The Euclidean norm has been universally used in optimizing various criteria for computing the joint velocities of a redundant arm. Here, the use of the infinity norm for defining these criteria is investigated. It is shown that in various applications, better physical representation of the performance criteria is obtained by using this norm instead of the Euclidean norm. The third section of the thesis deals with the formulation of dynamic equations and control law for a multiple cooperating manipulator system handling a common object, when the surfaces of the end-effectors and the object maintain rolling contact with each other. A new unified dynamic formulation for such a robotic system is derived, by modeling the rolling contacts as unactuated joints of the manipulators. This enables the formulation of trajectory planning methods that can be used to perform an additional subtask such as collision avoidance. In addition, a computed-torque type control law is designed, which explicitly controls the object trajectory, object internal forces and the contact trajectories.
Acknowledgments

It is with a great deal of pleasure that I acknowledge the encouragement and assistance provided by various people in the course of my graduate studies.

I consider myself extremely lucky to have had an advisor like Dr. Ian Walker. His guidance and advice at each juncture have been crucial to the development of this thesis. By giving me full freedom to work on the problems that interested me and by helping with suggestions and ideas whenever I was stuck, he provided an ideal environment for doing research.

Dr. Joseph Cavallaro and Dr. Dan Sorensen, have been very supportive and encouraging through the development of this dissertation. Special thanks are due to them for serving on my thesis committee and for their patience and understanding during the course of my research.

To Deirdre, Kristin and Monica, I am grateful for the shared ideas, conversation and laughter. To Kotax, a big thank you, for always being there to help me with any problem or question. To the ‘weekend group’, thanks for so many pleasant and fun-filled diversions.

I would also like to thank Nora for helping out cheerfully with the various administrative details of this dissertation.

Finally, I am eternally indebted to my parents for making all this possible, to my brother for taking up all the slack at home and to Shiva, for always being there for me.
## Contents

Acknowledgments ........................................ iii
List of Illustrations ....................................... vii
List of Tables ........................................... viii

1 Introduction ........................................ 1
   1.1 What is a Robot? ...................................... 1
   1.2 Industrial Robots ..................................... 2
   1.3 The Robotic Manipulator ......................... 3
   1.4 Manipulator Models .................................. 6
   1.5 Summary of Dissertation Contributions ............. 10
   1.6 Organization of Thesis ............................. 12

2 Nonlinear Least Squares for Inverse Kinematics ...... 14
   2.1 Introduction ........................................ 14
   2.2 Singular Configurations ........................... 16
   2.3 Damped Least Squares ............................. 19
   2.4 Failure of the first-order model .................. 21
   2.5 Augmentation with Second-Order Model .......... 24
   2.6 Non-existence of local minimizers ............... 26
   2.7 Switching strategy ................................ 30
   2.8 Simulations ........................................ 30
   2.9 Conclusions ....................................... 31

3 Minimum Effort Inverse Kinematics .................. 34
   3.1 Introduction ........................................ 34
   3.1.1 Resolved Motion Rate Control ................. 34
   3.1.2 Redundant Manipulators ....................... 35
   3.1.3 The Pseudoinverse Solution ................... 36
   3.2 Motivation for using the $\infty$-norm .......... 37
3.3 Definitions and notation ........................................ 39
3.4 Dual Optimization Problems .................................... 40
3.5 Algorithm for computing the dual solution .................. 44
3.6 Alternate Method for Dual Solution ......................... 48
3.7 Steepest Descent Algorithm .................................... 49
3.8 Weighting the \( \infty \)-norm ..................................... 53
3.9 Subtask Performance With Infinity-Norm Reduction ....... 54
3.10 Subtask Performance with \( \infty \)-Norm Criterion .......... 55
3.11 Examples .................................................................. 57
  3.11.1 Simulation 1 ...................................................... 57
  3.11.2 Simulation 2 ...................................................... 60
  3.11.3 Simulation 3 ...................................................... 65
  3.11.4 Simulation 4 ...................................................... 67
3.12 Discussion and Conclusions ...................................... 70

4 Multiple Manipulator Systems ................................... 75
  4.1 Introduction .......................................................... 75
  4.2 Grasping and Fine Manipulation .............................. 78
    4.2.1 Types of Contact ............................................. 79
    4.2.2 Rigid/Non-rigid grasps ..................................... 81
  4.3 Kinematics of Rolling Contact .................................. 82
    4.3.1 Some Differential Geometry ............................... 83
    4.3.2 Contact Kinematics ......................................... 87
  4.4 System Kinematics and Dynamics ............................. 90
    4.4.1 Kinematics of the Virtual Manipulator .................. 93
    4.4.2 Dynamics of the Virtual Manipulator .................... 97
    4.4.3 Concatenating the equations .............................. 106
    4.4.4 Static force analysis ....................................... 109
  4.5 Exploiting redundancy in system equations ................. 112
  4.6 Control Algorithm .................................................. 115
    4.6.1 Dimensional analysis ....................................... 118
    4.6.2 Control Law .................................................... 118
  4.7 Simulations .......................................................... 120
    4.7.1 Kinematic Simulations ..................................... 120
4.7.2 Control algorithm simulations ....................................... 120
4.8 Conclusions ..................................................................... 129

5 Summary and Future Work ...................................................... 135

A Proofs of Lemmas 2.1, 2.2 and 3.1 ........................................ 139
   A.1 Lemmas 2.1 and 2.2 ....................................................... 139
   A.2 Alignment Lemma 3.1 ..................................................... 140

B Proofs of Differential Geometric Lemmas .............................. 142

C Control Algorithm for redundant systems ............................. 146

D Parameters used in Controller simulations ............................ 149

   Bibliography ..................................................................... 151
Illustrations

1.1 Serial-link and Parallel-link Manipulators .......................... 3
1.2 Two-link Planar Manipulator ........................................... 4
1.3 Components of a Single-Arm Robotic System ......................... 6
1.4 Computed-torque control scheme ....................................... 9

2.1 Singularities in a 3-joint planar arm ................................ 17
2.2 Singular configuration generates high joint velocity ................. 18
2.3 Example yielding oscillations ........................................... 22
2.4 Schematic representation ................................................ 27
2.5 Configurations with zero gradient ...................................... 29
2.6 Example 1 - Linear model vs Adaptive algorithm .................... 32
2.7 Example 2 - Starting close to an internal singularity ............... 33

3.1 Manipulator with two prismatic joints ................................ 38
3.2 Simulation 1 : Comparison of Joint Velocity Norms - Planar Arm 59
3.3 Simulation 1 : Comparison of Joint Velocity Norms - the PUMA 60
3.4 Simulation 2 : Weighting of the first joint .......................... 61
3.5 Simulation 2 : Weighting of the PUMA wrist joints ................. 63
3.6 Simulation 2 : Weighting of the PUMA wrist joints ................. 64
3.7 Simulation 3 : Pseudoinverse solution trajectory .................... 65
3.8 Simulation 3 : Trajectories with and without norm reduction .... 66
3.9 Simulation 3 : Joint Velocity Norms and Elbow Trajectories ... 68
3.10 Simulation 4 : Trajectories of Robot Arm .......................... 69
3.11 Simulation 4 : Maximum deviation of the joints ................. 70
3.12 Evolutions of Robot Arm ............................................... 72
3.13 Comparison of norms .................................................. 73

4.1 Variations of a Robotic Manipulator ................................. 75
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>Cooperating Manipulator Systems</td>
<td>76</td>
</tr>
<tr>
<td>4.3</td>
<td>Fingertip Geometries</td>
<td>79</td>
</tr>
<tr>
<td>4.4</td>
<td>Coordinate System Mapping</td>
<td>83</td>
</tr>
<tr>
<td>4.5</td>
<td>Tangent Plane of S</td>
<td>84</td>
</tr>
<tr>
<td>4.6</td>
<td>Shape Operator</td>
<td>85</td>
</tr>
<tr>
<td>4.7</td>
<td>Contact Parameters</td>
<td>88</td>
</tr>
<tr>
<td>4.8</td>
<td>Rolling contact between two surfaces</td>
<td>92</td>
</tr>
<tr>
<td>4.9</td>
<td>Cooperating arms - Rolling contact</td>
<td>93</td>
</tr>
<tr>
<td>4.10</td>
<td>Rolling contact between end-effector and object</td>
<td>94</td>
</tr>
<tr>
<td>4.11</td>
<td>Directed System Graph for 3-link arm</td>
<td>98</td>
</tr>
<tr>
<td>4.12</td>
<td>Kinematics of contiguous bodies</td>
<td>102</td>
</tr>
<tr>
<td>4.13</td>
<td>Internal Forces at Object Center of Mass</td>
<td>107</td>
</tr>
<tr>
<td>4.14</td>
<td>End-effector forces at contact points and object center of mass</td>
<td>111</td>
</tr>
<tr>
<td>4.15</td>
<td>Example 1: System evolution with no subtask specified</td>
<td>121</td>
</tr>
<tr>
<td>4.16</td>
<td>Example 1: System evolution with collision avoidance subtask</td>
<td>122</td>
</tr>
<tr>
<td>4.17</td>
<td>Example 2: System evolution with no subtask specified</td>
<td>123</td>
</tr>
<tr>
<td>4.18</td>
<td>Example 2: System evolution with collision avoidance subtask</td>
<td>124</td>
</tr>
<tr>
<td>4.19</td>
<td>Example 3: Evolution of system</td>
<td>125</td>
</tr>
<tr>
<td>4.20</td>
<td>Example 3: Object trajectories</td>
<td>126</td>
</tr>
<tr>
<td>4.21</td>
<td>Example 3: Contact trajectories</td>
<td>127</td>
</tr>
<tr>
<td>4.22</td>
<td>Example 3: Internal force components</td>
<td>128</td>
</tr>
<tr>
<td>4.23</td>
<td>Example 4: Evolution of system</td>
<td>130</td>
</tr>
<tr>
<td>4.24</td>
<td>Example 4: Object trajectories</td>
<td>131</td>
</tr>
<tr>
<td>4.25</td>
<td>Example 4: Contact trajectories</td>
<td>132</td>
</tr>
<tr>
<td>4.26</td>
<td>Example 4: Internal force components</td>
<td>133</td>
</tr>
<tr>
<td>A.1</td>
<td>Schematic representation</td>
<td>139</td>
</tr>
</tbody>
</table>
Tables

4.1 Basic Types of Contact ........................................... 80
D.1 Link lengths and masses ......................................... 149
Chapter 1

Introduction

Though our smoke may hide the Heavens from your eyes,
It will vanish and the stars will shine again,
Because, for all our power and weight and size,
We are nothing more than the children of your brain.

- Rudyard Kipling, ‘The Secret of the Machines’

1.1 What is a Robot?

A machine made in the image of man [2]- this is the picture that the word robot most commonly conjures up, in the mind of a layman. The prevailing popular image of the robot is that of a walking, talking mechanical man (or, in rare instances, a woman!). This concept has enjoyed a lively existence in the human imagination even before it gained the name “robot”. Throughout history - in myth, fiction and reality - there have been many devices and concepts which today we might dub robots. These range from the early clockwork figures designed to amuse the wealthy to the computer-controlled industrial robots of the modern age.

In the twentieth century, fictional robots have proliferated in books, films and magazines, creating tantalizing images of androids and cyborgs and greatly influencing the popular imagination. However, no existing robot looks any more like the robotical fantasies of science fiction than, say, the Mercury space capsules looked like the USS Enterprise in Star Trek. While fictional robots are created for their dramatic effect, real-life robots, strive, like all other machines, principally for utility. The science or technology of robotics* certainly aims to produce devices which emulate humans, but in the utilitarian sense rather than the visual sense. What present-day robotics is concerned with, then, are those human attributes which it seems both useful and feasible, not only to try and imitate but also enhance, in terms of reliability, power and accuracy, for instance. However, there does not seem to be much advantage in

*It is worth noting that the word robot was given to the world by a dramatist (Capek, with his play Rossum’s Universal Robots) and the word robotics by a novelist (Isaac Asimov)!
incorporating all of the attributes in a single machine. It seems more likely that, on
the principle of ‘every robot to his trade’, the robots of the future will be specialized
to perform certain types of functions with certain desired attributes. Thus, it may
be more accurate to qualify the original definition to read something like ‘a category
of machines which are, in one or more respects, made in the image of man’

1.2 Industrial Robots

Most active robots in the world are involved with industry. The development of indus-
trial robots has been stimulated largely by a variety of economic motives. According
to Joseph Engelberger, who founded Unimation Inc. and who is considered the ‘Father
of Industrial Robots’, it was the rising cost of labor in the post-war years and the
difficulty, in a booming labor market, of finding employees prepared to do unpleas-
ant, boring or repetitive jobs, which gave the early robots a toehold in the workplace.
Although the need for industrial robots must have been perceived earlier, it was only
when a whole group of technologies had advanced to a certain point, that it was pos-
sible to build machines which could provide a direct and economically advantageous
substitute for human labor.

Despite the aura of high technology that surrounds it, the role for which the
industrial robot was conceived was hardly glamorous, or in human terms, challenging.
Around 1960, when the first industrial robots appeared on the scene, the tasks that
sorely needed automation in industry were usually unskilled and mundane - presenting
workpieces to machines and removing them, transferring parts from one conveyor belt
to another, packing and stacking finished products, etc. Thus, a lot of them boiled
down to pick-and-place operations i.e. picking up objects in one place and putting
them down in another - actions which a human does using his/her arm. Hence,
the first robots were designed to reproduce the human arm. This basic equivalence
between the robot manipulator and the human arm has undergone little change over
the years, although the level of sophistication of the robotic arm has been steadily
improving.

The early industrial robots had little programmability; they operated on tradi-
tional mechanical lines with their movements dictated by systems of cranks, levers
or cogs and often limited to set routines and altering those meant redesigning the
machine. Today, however, with computer control, industrial robots are extremely
flexible production units, involved in a variety of tasks such as painting, welding, de-
burring, diecasting, etc. They can drill holes, sand-down workpieces, and assemble a wide range of units from tiny electronic components to large automobile parts. There is a growing diversity of robot models, ranging from delicate devices for fine work to mighty machines for handling heavy castings. Robots are being used in environments that could be hazardous or inconvenient to humans such as nuclear tanks or in partial vacuum. Robots are more flexible than humans in some respects, less flexible in others, but while the capabilities of humans are more or less well defined, those of the robot are evolving rapidly.

1.3 The Robotic Manipulator

Until about a decade ago, robotic research was concentrated almost entirely on controlling a single robotic manipulator. In recent years, the need for more complex manipulations and the evolution of devices such as multi-fingered hands and multi-legged systems, has spawned a large body of research in the study of robotic systems formed by several manipulators working cooperatively together, either for manipulating a common object or for the purposes of locomotion. However, the building block, so to speak, for these systems is still the robotic manipulator.

![Diagram of 3-Link Planar Arm and 6-legged Stewart Platform]

**Figure 1.1** Serial-link and Parallel-link Manipulators
A robotic manipulator consists of several links connected by joints to form an arm. Manipulators can be classified as being serial or parallel, depending on the way the links are connected. Most of the robots in use today have their links connected serially. In this work, when we refer to a manipulator, we always mean a serial manipulator.

Serial robot manipulators, in their basic form, are open kinematic chains, composed of rigid links connected by joints. They can be moved around in space by imparting motion at the joints through actuators (which may be of different kinds, such as electric, hydraulic or pneumatic). Typically, each joint possesses a single degree-of-freedom and is either of the revolute or prismatic type, depending on whether it allows relative rotational or translational motion between its two adjacent links.

![Two-link Planar Manipulator](image)

**Figure 1.2** Two-link Planar Manipulator

An end-effector or gripper (hand) is attached to the arm by means of a wrist. The positions of the joints determine the configuration of the arm. Each arm configuration corresponds to a unique position and orientation of the end-effector. The workspace of a manipulator is the space composed of all points which can be reached by the end of the arm or some point on its wrist (not necessarily the end-effector since the end-effector could be a replaceable fixture).
An important parameter of a manipulator is the number of degrees of freedom (i.e. the number of joints) that it possesses. To move and orient a body in three-dimensional space, a mechanism should have at least six degrees of freedom - three to locate the end-effector at a prescribed point in the manipulator workspace and three to properly orient it at this location. Thus, for an arm with six joints (and hence 6 degrees of freedom), the joint configuration corresponding to a given position and orientation of the end-effector will be unique up to finite variations. The human arm itself can be considered to have 6 main joints - two at the shoulder (one motion away and towards the body and the other forward and backward), two at the elbow (one motion used to fold/stretch the arm and the other to roll the front portion of the arm about an axis joining the elbow to the wrist) and two at the wrist (the pitch and the yaw motions). A detailed analysis of the motions of the various joints can be found in kinesiologic studies ([44], [81], [103]).

Robotic arms can be made kinematically redundant by imparting them with extra degrees of freedom in the joint space. Thus, a manipulator is said to be redundant when $n$, the number of its joints is greater than its task space dimension $m$. A redundant manipulator is characterized by the fact that it can have infinite number of joint configurations corresponding to most end-effector positions in its workspace. While the extra degrees of freedom significantly complicate programming and control strategies, they also considerably augment the usefulness of the robot and make it more human-like.

Mathematically, a robotic arm is described by its kinematic and dynamic equations. The kinematics of the arm involves the study of relations between the positions, velocities and accelerations of its various parts. Kinematic analysis is necessary for planning and executing desired motions of the manipulator as well as for dynamic computations. The dynamic equations of the arm describe its evolution in time in response to a set of external forces/torques acting on it.

However, the robotic system is more than just the arm itself. As shown in fig.(1.3), in addition to the arm, the system consists of an external power source, end-of-arm tooling, external and internal sensors, servo mechanism, computer interface and the controller [94]. The robot controller, in some sense, is the brain controlling the motions of the mechanical arm. It is the controller which determines, based on the kinematic and dynamic models of the arm and the sensor measurements, the set of inputs to the joint actuators necessary for producing a desired motion. Without the controller, the robotic manipulator would be like a human arm cut off from its body.
In this work, we investigate kinematic and dynamic control algorithms for single-arm and multiple-arm robot systems. Before discussing the contributions of this dissertation, it will be helpful to present some background and notation for the mathematical models of a single manipulator.

1.4 Manipulator Models

Consider the manipulator in fig. 1.2. Let the end-effector position and orientation vector in the base frame be represented by $x \in \mathbb{R}^m$ and the vector of joint positions by $\theta \in \mathbb{R}^n$. 
The forward kinematic function $f$ of a robotic manipulator maps $\theta$ to $x$, i.e.

$$x = f(\theta)$$  \hspace{1cm} (1.1)

$f$ is a nonlinear function of the joint positions $\theta$. Closed-form expressions for $f$ can be obtained by setting up coordinate frames for each link and using the transformation matrices between these frames to derive the forward kinematic equations. A systematic method to define the link coordinate frames and obtain the transformation matrices has been proposed by Denavit and Hartenberg [22] and is almost universally used in the analysis of robotic manipulators. For any robotic manipulator, the forward kinematic function $f$ always exists and can be computed in closed-form. Note that the forward kinematic solution is unique, i.e. for any given vector $\theta$, there exists a unique end-effector position $x$.

The reverse procedure i.e. the computation of a set of joint positions for a given end-effector position vector, comprises the Inverse Kinematic computation of a robotic manipulator. However, unlike the forward kinematic function $f$, the inverse kinematic function for most manipulators, cannot be obtained in the form of analytical, closed-form expressions. More importantly, the inverse kinematic solution, for a given end-effector position, is not unique and there can exist multiple or even infinitely many joint position vectors corresponding to a given $x$. However, solving the inverse kinematic problem is fundamental to the control of a robotic manipulator, since a manipulation task is inherently specified in the form of end-effector positions, velocities and accelerations while the control inputs need to be specified at the joint level.

Differentiation of eq.(1.1) yields the velocity relation$^1$

$$\dot{x} = J\dot{\theta} \quad ; \quad J(\theta) \triangleq \frac{\partial f}{\partial \theta} \in \mathbb{R}^{m \times n}$$  \hspace{1cm} (1.2)

where $J(\theta)$ is the Jacobian matrix of the direct kinematic function $f(\theta)$. Computation of the inverse kinematic solution at the velocity level involves the determination of a joint velocity vector $\dot{\theta}$ for a given end-effector velocity $\dot{x}$, at a joint configuration $\theta$. Although (1.2) gives an analytical definition of the Jacobian matrix $J$, computationally it is more efficient to construct this matrix using the vectors defined by the Denavit-Hartenberg notation [94].

$^1$The symbol $\triangleq$ is used whenever we introduce a new variable to denote a certain matrix or vector. For example, in eq.(1.2), the symbol $J$ is used to denote the Jacobian matrix of $f$ defined on the right-hand side
Similarly, the second-order kinematic equations relating the accelerations of the end-effector and the joints can be obtained by differentiating (1.2).

\[ \ddot{x} = J\ddot{\theta} + J\dot{\dot{\theta}} \]  
(1.3)

The dynamic equations of motion of the manipulator are commonly written in matrix-vector form as:

\[ [M(\theta)] \ddot{\theta} + N(\theta, \dot{\theta})\dot{\theta} + G(\theta) + [J^T] F = \tau \]  
(1.4)

where

\[ [M(\theta)] = \text{Inertia Matrix of the manipulator} \]
\[ N(\theta, \dot{\theta}) = \text{Vector of Coriolis and centrifugal terms} \]
\[ G(\theta) = \text{Gravity vector} \]
\[ F = \text{Forces/torques exerted by end-effector} \]
\[ \tau = \text{Vector of torque inputs to the joint actuators} \]

Derivation and further details about the kinematic and dynamic equations of a manipulator can be found in any standard robotic text such as [94].

The control problem for robot manipulators involves the determination of a set of joint torque inputs \( \tau \) which will result in the end-effector following a desired trajectory, typically specified either by a sequence of end-effector position and orientation vectors \( \mathbf{x} \) or by a continuous trajectory. Various types of control schemes have been studied for robotic manipulators. The simplest method and one which is still employed for many industrial manipulators, is independent joint control, where each manipulator joint is controlled as a single-input, single-output (SISO) system. As the name suggests, for this control scheme, the dynamics of each joint actuator are determined and controlled independently. Any coupling effects between the joints are either ignored, or treated as a disturbance. This control scheme gives acceptable results for simple motions but is not suitable for fast motions or motions with large variations.

The independent joint control method can be considered to be a simplified form of the Computed-Torque method, which is one of the most common control techniques for robot manipulators. Most manipulator control methods can be considered as special cases of this technique. The computed torque method itself, is a special application of the feedback linearization technique for nonlinear systems. Fig. 1.4 depicts this control method schematically [52].

\[ ^\dagger \text{Conventionally, the force/torque vector at the end-effector is represented as a force while the force/torque inputs to the joints are represented as torques} \]
The computed-torque control scheme can be considered to be a partitioned controller, one section being model-based while the other being the servo portion. Thus, consider a manipulator with its end-effector in free space, with the dynamic model

$$\tau = [M(\theta)] \ddot{\theta} + N(\theta, \dot{\theta}) \dot{\theta} + G(\theta)$$

The model-based portion of the control law is

$$\tau = [M(\theta)] (\ddot{\theta}_d - u) + N(\theta, \dot{\theta}_d) \dot{\theta} + G(\theta)$$  \hspace{1cm} (1.5)

where $\ddot{\theta}_d$ is the desired joint acceleration vector and $u$ is the control input signal which is determined by the servo control law. Usually, a PD control law is used for robot manipulators. Thus, if $\theta_d$ and $\dot{\theta}_d$ are the desired joint position and velocity vectors and $\theta$ and $\dot{\theta}$ are the actual (or measured) joint position and velocity vectors, the errors in joint space are computed as

$$e = \theta_d - \theta; \quad \dot{e} = \dot{\theta}_d - \dot{\theta}$$
Then, the control input $u$ can be chosen as

$$ u = -K_v \dot{e} - K_p e $$

Then, substituting from eq.(1.6) into eq.(1.5), the error equation for the closed loop system is obtained as

$$ \ddot{e} + K_v \dot{e} + K_p e = 0 $$

By choosing the gain matrices $K_v$ and $K_p$ to be diagonal matrices with positive values along the diagonal, this error system can be made asymptotically stable. It is important to note that although the selection of diagonal gain matrices results in decoupled control at the outer loop, this does not imply a decoupled joint-control strategy. This is because the multiplication by the inertia matrix and the addition of the nonlinear terms in the model-based law "scrambles" the control input $u$ among all the joints. As a result, the joint positions and velocities at all joints are required to compute the torque input for any one joint.

Different choices of the servo control law result in variations of the basic control technique described above. (This method reduces to independent joint control when $[M] = I$ and $\mathbf{N} + \mathbf{G} = -\dot{\theta}_d$). The essential idea is to cancel out the nonlinearities in the model (using the model-based law) and then treat the system as a linear system.

The biggest disadvantage of this control scheme is that the parameters and structure of the system must be known to compute eq.(1.5). Nevertheless, when the errors in the parameters are not too large, this control scheme gives satisfactory performance. Also, the computed-torque controller provides a starting point, based on which adaptive and robust controllers can be designed. A detailed stability analysis of these controllers is contained in [52].

### 1.5 Summary of Dissertation Contributions

The focus of this dissertation is on developing new algorithms for inverse kinematics of manipulators and control of multiple cooperating manipulator systems. The results presented in this thesis can be classified under three different headings and are summarized below. The ideas outlined below are described in detail through the thesis.

- **Inverse Kinematics for Unreachable End-Effector Positions**

  In earlier work[25], we have studied the damped least-squares method for com-
puting the inverse kinematics of robot manipulators with bounded joint velocities. This technique can be used to prevent high joint velocities of the manipulator near singular configurations. However, convergence problems are encountered when the desired end-effector position does not lie within the reachable workspace of the manipulator.

In this dissertation, we propose the use of an adaptive nonlinear least-squares algorithm to ensure convergence of the manipulator at the workspace boundary. This algorithm switches from the first-order model used in the damped least-squares technique to a second-order model to ensure convergence, in cases where oscillations would result otherwise.

- **Minimum-effort Inverse Kinematics**
  As described above, redundant manipulators admit an infinite choice of inverse kinematic solutions for a specified end-effector trajectory. Euclidean norm criteria have been widely used in the robotics literature to optimize different performance indices.

  Here, we introduce infinity-norm inverse kinematics for redundant manipulators. Various algorithms for computing a minimum-effort inverse kinematic solution, corresponding to minimizing the infinity-norm of the joint velocity vector, are investigated. In addition, we show how subtask performance criteria such as obstacle avoidance or joint limit avoidance can be incorporated in the infinity-norm framework.

- **Control of Multiple Manipulator System with Rolling Contacts**
  For tasks which, for various reasons, may be beyond the capability of a single manipulator, multiple manipulator systems must be employed. The need for increasingly sophisticated manipulations and the advent of artificial multi-fingered hands has led to the study of systems where the object being manipulated is not held rigidly by the cooperating manipulators, but instead rolls or slides between the end-effector surfaces.

  The dynamics and control of multiple manipulator systems in which rolling contact is maintained between the end-effectors and the object, is investigated in this work. A new dynamic formulation and resulting computed-torque type control algorithm are derived by modeling the rolling contacts as unactuated joints of the manipulators. The main contribution of this new algorithm is
that it enables simultaneous control of the object trajectory, the object internal forces and the contact trajectories of the contacts on the object/end-effector surfaces.

1.6 Organization of Thesis

The chapters of the thesis are organized corresponding to each of the research issues described above. In each chapter, we present the background material relevant to each subject before discussing the results of our research. At the end of each chapter, simulations demonstrating these results are presented.

Chapter 2 deals with the problem of computing inverse kinematics when the desired end-effector position is outside the reachable workspace of the manipulator. Such a situation can occur, for example, when the manipulator is being controlled using a joystick. This may cause the manipulator to oscillate near the boundary of its workspace. We discuss the augmentation of the first-order damped least-squares algorithm with a second-order model in such situations. The resulting algorithm ensures convergence to a configuration that places the end-effector at the closest possible position from the desired position.

In chapter 3, the use of the infinity-norm in computing inverse kinematic solutions for redundant manipulators is investigated. The Euclidean norm has been the predominant measure used to compute performance criteria for inverse kinematic computations. We show how using the infinity-norm instead of the Euclidean norm can lead to a better measure of performance in certain applications. In particular, methods for minimizing the norm of the joint velocity vector and for optimizing sub-task criteria are considered. Computationally efficient algorithms for determining such solutions are discussed.

Multiple manipulator systems (such as cooperating arms or multifingered hands) handling a common object with rolling contact are the subject of discussion in chapter 4. Rolling contact kinematics is studied and new insights into the differential geometric concepts are presented. The main contribution of this chapter is a new formulation for the dynamic equations of the system which is based on the idea of modeling the rolling contacts between the end-effector and object surfaces as additional unactuated joints of the manipulator. This formulation enables the design of a control law that allows explicit control of the object’s trajectory, the internal forces in the object and the trajectory of the contact points on the object/end-effector surfaces.
Finally, conclusions are presented in chapter 5, along with a discussion of future avenues of research.
Chapter 2

Nonlinear Least Squares for Inverse Kinematics

Anything that is manufactured is manipulated. Every part is manipulated while it is made. Every part is manipulated while it is assembled. A part is manipulated when it is delivered from a plant. Everything is manipulation.

-George C. Devol, Jr.

2.1 Introduction

As described in the previous chapter, solving the inverse kinematic problem i.e. determining the joint position, velocity and acceleration vectors for a given end-effector trajectory, is fundamental to the control of robot manipulators. This problem has been researched extensively by the robotics community in the past two decades. The proposed solution techniques can be broadly classified as being either geometric or iterative. Geometric methods are used to determine a closed-form inverse kinematic solution, if it exists. However, only certain classes of robots (e.g. spherical wrist manipulators, such as the PUMA) admit closed-form inverse kinematic solutions. On the other hand, numerical techniques can be applied to general manipulators and hence constitute the preferred approach when closed-form solutions do not exist or are extremely tedious to compute.

Many different numerical techniques for computing the joint kinematic parameters corresponding to specified end-effector kinematic parameters have been investigated in the literature (e.g. [5], [35], [43], [102]). One approach is to apply the Newton-Raphson method to the governing kinematic equations \( x = f(\theta) \) (eq. 1.1) ([40], [41], [96], [104]. This is equivalent to a more popular approach, which is to solve the relation \( \ddot{x} = J\dot{\theta} \) in eq. (1.2), for the joint velocities \( \dot{\theta} \) and then use a first-order integration scheme such as,

\[
\theta_{t+1} = \theta_t + \dot{\theta}(\delta t)
\]

(2.1)

to compute the corresponding joint positions. (Here \( \delta t \) is the time interval between two successive iterations). The pseudoinverse of the Jacobian is commonly used to
solve for the joint velocity vector as

$$\dot{\theta} = J^+ \dot{x} = J^T (JJ^T)^{-1} \dot{x}$$

with the expression on the right hand side being valid when $J$ has full row rank. This method has been widely used, especially for redundant manipulators, for which the pseudoinverse yields a minimum Euclidean norm solution (for a review, see [79], [92]). To improve the performance of these approaches in terms of stability and efficiency, modifications to the Newton-Raphson and predictor-corrector integration methods have also been studied ([3], [16], [36], [40], [41]). These modified algorithms always converge, if a physically realizable robot configuration exists. They also yield better performance near singular configurations (see section 2.2 below), though they still do not converge to the exact singularity. Heuristic direct search methods have been proposed to enable convergence to singularities ([43], [63], [102]). However, the local convergence rates of these methods are expected to be slower than those of the gradient-based methods [57].

In this chapter, we consider the problem of computing the inverse kinematic solution in situations where the desired end-effector position may or may not be within the reachable workspace of the manipulator. Situations where the specified end-effector position is unreachable can arise when, for example, the motion of the manipulator is being controlled through a joystick or when the reachable workspace of the robot is reduced due to the failure of a joint. In such cases, the robot manipulator will be required to converge to an external singularity which will place its end-effector to the closest possible position from the desired position. Since the algorithms using first-order models of the forward kinematic function (i.e. approximations of the function which utilize first derivatives only) can be ill-behaved near singularities and do not converge to exact singular configurations, they will be unable to bring the manipulator to rest at the outstretched configuration and may also cause the end-effector to oscillate about the desired position. In [85], this problem has been tackled by formulating the kinematics in the complex domain and computing complex joint angles.

Here, the use of an adaptive algorithm is proposed, to obtain convergence of the arm to a configuration that places the end-effector at the closest possible distance from the desired position, irrespective of whether the desired end-effector position lies within or outside the reachable workspace. Convergence takes place even if the configuration is a singular one. The algorithm uses the damped least-squares method
(see section 2.3) at the position level and an alternate second-order model in the circumstances mentioned above. The second-order model is formulated by twice differentiating the (two-) norm of the error vector from the actual to the desired end-effector location. (This model is different from the one used in quadratic rate control [84] where a 3rd-order tensor is used as the Hessian). Such an adaptive algorithm was proposed in [23] to obtain more reliable results than the Gauss-Newton or Levenberg-Marquardt methods, for large residual problems in overdetermined systems. Here, we use the same idea but with somewhat different convergence criteria and switching strategy, since our problem is primarily zero-residual, but could have a non-zero residual in certain cases.

We also prove, in this work, that the nonlinear least squares problem in inverse kinematics has no local minimizers within the workspace of the manipulator. This aids in the detection of regions where oscillations will occur and enables switching over to the second-order model to ensure convergence.

The chapter is organized as follows. The next section describes what singular configurations of a manipulator are. Section 2.3 establishes the framework for using the damped least-squares method at the position level. Section 2.4 describes the problems that may be encountered when using this method alone to compute the inverse kinematic solution. In section 2.5, we introduce the second-order model and show how it can be used to ensure convergence in situations where the damped least-squares method fails. Section 2.6 contains the statement and proof of a theorem about local minimizers which guarantees proper switching to the second-order model. In section 2.7 we present a strategy of using the two models to ensure convergence to the desired configuration. Simulations are presented in section 2.8 and conclusions in section 2.9.

2.2 Singular Configurations

Singular configurations are manipulator configurations where the number of degrees of freedom of the manipulator in the task space reduces due to certain kinematic alignments of the manipulator links. As a result, the ability of the manipulator to move the end-effector decreases drastically in certain directions. Hence, very high joint velocities are required to make the end-effector move in these directions, which are known as singular directions. Mathematically, singular configurations correspond
to a rank-deficient Jacobian. (For redundant manipulators, this implies that the Jacobian does not have full row rank).

Numerous ways of characterizing a manipulator’s singular configurations have been studied (e.g. [8], [91]). For our purposes, two types of singularities can be identified:

1. External: When a singular configuration places the end-effector on the boundary of the manipulator workspace, the configuration corresponds to an external singularity.
2. Internal: If the manipulator loses its freedom of movement in some direction when the end-effector is in the interior of the workspace, the singularity is said to be an internal singularity.

Fig. (2.1) depicts the singularities in a three-DOF planar arm. In this case, only the configuration A is an external singularity while configurations B, C, D are internal singularities. Physically, singular configurations correspond to the robot arm being folded or outstretched in some manner. Hence, moving the end-effector in certain directions, from such a configuration, requires that one or more joints of the arm undergo a large change in position and hence a high velocity has to be generated at such joints. Fig. (2.2) shows the large changes in the joint angles required to move the end-effector of the manipulator by a relatively small distance in the singular direction, from the near-singular joint configuration A to the joint configuration B.

![Diagram](image)

**Figure 2.2** Singular configuration generates high joint velocity

Although the pseudoinverse yields a minimum-norm solution, it is still unable to prevent high joint velocities near a singular configuration. This is because, close to a singularity, all joint velocity vectors which can achieve the desired end-effector velocity can have very high norm, if the specified end-effector velocity has some component in the singular direction. Hence, even the minimum-norm solution may be infeasible.

In order to achieve feasible solutions in such situations, the Singularity Robust inverse arising from the damped least-squares method has been proposed ([75], [101]). In this case, whenever the pseudoinverse solution is not feasible, the joint velocity
vector is damped using a damping factor $\lambda$ which is equivalent to bounding the Euclidean norm of the vector $\hat{\theta}$ by some upper bound $\Delta$ ([27], [31], [60]). ($\lambda = 0$ if the pseudoinverse solution is feasible). Thus, at each configuration, the damped least-squares method defines a “trust region” ([71], [93]) in joint space from which the joint velocity solution is picked.

### 2.3 Damped Least Squares

In this section, we describe the damped least-squares method which has been used to prevent high joint velocities of a manipulator. This method uses the Levenberg-Marquardt technique ([24], [70]) to damp joint velocities, if necessary, in the vicinity of a singular configuration.

Let the desired end-effector location be represented in terms of a homogeneous matrix as

$$T_d = \begin{bmatrix} \mathbf{u}_1^d & \mathbf{u}_2^d & \mathbf{u}_3^d & \mathbf{p}^d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $\mathbf{u}_1^d, \mathbf{u}_2^d, \mathbf{u}_3^d \in \mathbb{R}^3$ are the desired unit orientation vectors while $\mathbf{p}^d \in \mathbb{R}^3$ is the vector of desired Cartesian coordinates. If $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \mathbf{p}$ are the corresponding vectors at any given end-effector location, then the distance from the desired location can be determined as\(^5\)[102]

- Position error $\|\mathbf{p}^d - \mathbf{p}\|$
- Orientation error $\left[ \sum_{i=1}^{3} (\mathbf{u}_i^d \cdot \mathbf{u}_i - 1)^2 \right]^{1/2}$

Define the vector $\mathbf{z} \in \mathbb{R}^6$ (for the spatial case) as

$$\mathbf{z} = [\mathbf{p}^T, \mathbf{u}_1^d \cdot \mathbf{u}_1, \mathbf{u}_2^d \cdot \mathbf{u}_2, \mathbf{u}_3^d \cdot \mathbf{u}_3]^T$$

and the corresponding $\mathbf{z}^d$ as

$$\mathbf{z}^d = [(\mathbf{p}^d)^T, 1, 1, 1]^T$$

From now on, $\mathbf{z}$ will be referred to as the actual or current end-effector position, although it incorporates both position and orientation information. Similarly, $\mathbf{z}^d$ is called the desired end-effector position. (Of course, if the orientation of the end-effector is not important, then $\mathbf{z}^d = \mathbf{p}^d$ and $\mathbf{z} = \mathbf{p}$).

\(^5\)Throughout this chapter, $\|\cdot\|$ indicates the 2-norm
For a given joint configuration \( \theta \in \mathbb{R}^n \), the residual function \( r : \mathbb{R}^n \rightarrow \mathbb{R}^n \) is defined as the difference between the desired end-effector position \( x^d \) and the current end-effector position \( x \).

\[
r(\theta) = x^d - x = x^d - f(\theta)
\]

Here \( f(\theta) \) is the forward kinematic function (see eq.(1.1)) which is a continuous function, mapping the joint positions to the end-effector location. An affine or first-order model of \( r(\theta) \) around the current joint configuration \( \theta_t \) is [24]

\[
A_t(\theta) = r_t - J_t[\theta - \theta_t]
\]

where \( r_t = r(\theta_t) \) and \( J_t = J(\theta_t) = \frac{\partial f}{\partial \theta}(\theta_t) \)

Thus, we can pose the problem of moving toward the desired end-effector position with bounded steps as [36]

\[
\min_{\delta} \|A_t\| \quad \text{subject to} \quad \|\theta - \theta_t\| \leq \Delta
\]

The solution to this problem can be obtained using the Lagrange multiplier method as

\[
\theta = \theta_t + (J_t^T J_t + \lambda I)^{-1}J_t^T r_t
\]

Here \( \lambda \geq 0 \) is known as the damping factor. This iteration is also known as the Levenberg-Marquardt iteration ([35], [24]). The equivalence of the two expressions (2.4) and (2.5) can be shown using the Singular Value Decomposition (SVD) of \( J_t \) [61]. This solution yields the joint configuration for the next time step. Also, the inequality constraint ensures that the norm of the difference between successive joint configurations is bounded and is always less than or equal to a pre-specified parameter \( \Delta \). Notice that dividing (2.3) by the time interval \( \delta t \) between two successive iterations, converts the problem to

\[
\min_{\delta} \|\dot{x} - J\dot{\theta}\|
\]

subject to \( \|\dot{\theta}\| \leq \Delta/\delta t \)

where we have used the first-order approximations

\[
\dot{x} = (x^d - x_t)/\delta t \quad \text{and} \quad \dot{\theta} = (\theta - \theta_t)/\delta t
\]
The problem (2.6) has been studied as the damped least-squares problem at the velocity level ([26], [28], [31], [60], [75], [90], [101]). This problem is equivalent to ([27], [31], [61])

\[ \min \| \dot{\hat{x}} - J\dot{\theta}\|_2^2 + \lambda \| \dot{\theta}\|_2^2 \]  \hspace{1cm} (2.7)

The solution to both (2.6) and (2.7) is

\[ \dot{\theta} = (J^T J + \lambda I)^{-1} J^T \dot{\hat{x}} \]  \hspace{1cm} (2.8)

\[ = J^T (J J^T + \lambda I)^{-1} \dot{\hat{x}} \]  \hspace{1cm} (2.9)

which is equivalent to dividing (2.4) and (2.5) by $\delta t$ when a first-order approximation is assumed.

In this work, we have opted to formulate the same problem at the position level as in (2.2). This enables the computation of the distance of the current end-effector position from the final goal position and this information is useful in making the decision to switch over to a second-order model whenever the first-order model fails to be a good approximation to the direct kinematic function $f(\theta)$.

In the following section, we demonstrate the need for switching to a second-order approximation in certain situations.

### 2.4 Failure of the first-order model

As mentioned in the previous section, problems (2.6) and (2.7) are equivalent. This is because the objective function of (2.7) is the Lagrangian for (2.6). Thus, the only difference between (2.6) and (2.7) is the choice of parameters i.e. in (2.6) we need to choose a suitable upper bound $\Delta$ for the joint velocity norm whereas in (2.7), the damping factor $\lambda$ has to be chosen explicitly. This implies that using a non-zero positive damping factor $\lambda$ in (2.7) is equivalent to bounding the joint velocity norm by $\Delta/\delta t$ for some $\Delta$.

From the numerical point of view, by placing a bound on the step taken in joint space in (2.3) (or, on the joint velocity norm in (2.6) or (2.7)), we make the assumption that the linear model (2.2) adequately approximates the nonlinear function $r(\theta)$ within a joint space ball of radius $\Delta$. Hence, in the regions where the actual function is not well approximated by the affine model within the "trust region" defined by the chosen $\Delta$ (chosen directly or indirectly), the inverse kinematic solution obtained by using eqs. (2.5) or (2.9) will not be a valid one.
One example of such a situation is when a 3-link planar arm (with link lengths 1, 0.5 and 0.5) is started from the configuration $[5^\circ \quad -5^\circ \quad 0^\circ]^T$ and the desired end-effector velocity is $\dot{x} = [1 \quad 0]^T$ which would mean that the desired end-effector position is out of the manipulator workspace (fig. 2.3). Suppose we determine the damping factor in (2.7) using the manipulability measure $w$ as a monitor [75] i.e.

$$\lambda = k \left( 1 - \frac{w}{w_t} \right) \quad \text{if} \quad w < w_t$$

$$\lambda = 0 \quad \text{if} \quad w \geq w_t$$

(2.10)

where $k$ is a constant, $w = \sqrt{det(J^TJ)}$ and $w_t$ is a threshold value. If $w_t = 0.5$ and $k = 0.001$, then the manipulator oscillates about the external singularity. The sequence of configurations is obtained as

$$\begin{bmatrix}
5.0 \\
-5.0 \\
0.0
\end{bmatrix}
\begin{bmatrix}
0.4463 \\
2.2674 \\
3.6337
\end{bmatrix}
\begin{bmatrix}
4.7007 \\
-3.2120 \\
-2.4895
\end{bmatrix}
\begin{bmatrix}
0.2876 \\
2.9036 \\
2.8924
\end{bmatrix} \ldots$$
Using (2.6) with $\Delta = 5$ also results in similar oscillations after 4 iterations. In this case, the sequence of configurations from the 4th iteration looks like

\[
\begin{bmatrix}
3.2677 \\
-1.1134 \\
-0.9817
\end{bmatrix}
\begin{bmatrix}
1.9029 \\
0.7416 \\
0.7221
\end{bmatrix}
\begin{bmatrix}
3.2441 \\
-1.0965 \\
-1.0183
\end{bmatrix}
\begin{bmatrix}
1.882 \\
0.7449 \\
7.0228
\end{bmatrix}
\ldots
\]

Note that the oscillation problem can be avoided by increasing the damping factor $\lambda$ (in the case of (2.7)) or reducing the trust region $\Delta$ (in the case of (2.3) or (2.6)) in these situations. For instance, in the above examples, in the first case if $k = 0.01$, then the oscillations do not take place but the manipulator moves very, very slowly towards the singularity. Similarly in the second case if $\Delta$ is taken to be smaller or is successively reduced (similar to (2.10)), say

\[
\Delta = \Delta_m \left( \frac{w}{w_l} \right) \quad \text{if } w < w_l
\]

\[
= \Delta_m \quad \text{if } w \geq w_l
\]

then once again the oscillations do not occur as the arm moves infinitesimally slowly towards the singular configuration.

In these cases, the manipulator will only move slower and slower without convergence to the minimizing configuration. In these situations, we would actually like the arm to settle into the outstretched position which places the end-effector at the closest possible distance from the desired position. Note that the corresponding joint configuration, in this case, would be an external singularity.

If the end-effector is not near the workspace boundary but is in the interior of the workspace, the linear model will always be able to generate joint velocities that will take the end-effector closer to the goal point, assuming that it is not at the goal position itself (see theorem 2.1). Hence, the problem of oscillations is likely to occur only near the workspace boundary, when the goal position is physically not reachable by the arm.

When the inverse kinematic problem is formulated at the position level as in (2.3), an indication of possible oscillations in the solution can be obtained by monitoring the reduction in the distance of the end-effector from the desired position i.e. $red(t) = \|z(L_t)\| - \|z(L_{t+\delta t})\|$. If at any iteration, this quantity is not positive, it indicates that the end-effector has not moved closer to the goal and hence the linear model is not approximating the function well.

What we need in these situations is a better approximation of the nonlinear function that we are trying to minimize, in order to make the manipulator converge to
the solution in a few iterations. Let us see how we can obtain a model which will give an adequate approximation in cases where the linear model is not satisfactory.

2.5 Augmentation with Second-Order Model

In this section, we describe a second-order model [24] for the direct kinematic function and use it to augment the damped least-squares technique in regions where the linear model fails to hold true. This second-order model is obtained by posing the inverse kinematic problem as a nonlinear least-squares problem. Nonlinear least-squares methods are widely used in data-fitting applications and hence are applied to overdetermined problems where an exact solution usually does not exist. However, in our case, the inverse kinematic problem is normally a zero-residual problem i.e. an exact solution does exist, except when the desired end-effector position happens to be outside the reachable workspace of the manipulator. Using the second-order model, an algorithm can be devised, which chooses adaptively between the two models ([32], [23]), so as to ensure convergence of the arm to the desired configuration.

Define $\phi(\theta)$ as

$$\phi(\theta) = \frac{1}{2} || x(\theta) ||^2$$

(2.11)

which is the square of the Euclidean distance between $x^d$ and $x$ at the configuration $\theta$. Then the gradient of $\phi(\theta)$ is

$$\nabla \phi(\theta) = -J(\theta)^T r(\theta)$$

(2.12)

and the Hessian of $\phi(\theta)$ is

$$\nabla^2 \phi(\theta) = J(\theta)^T J(\theta) + \sum_{i=1}^{n} r_i(\theta) \nabla^2 r_i(\theta)$$

$$= J(\theta)^T J(\theta) + S(\theta)$$

(2.13)

where $S(\theta)$ is defined as $\sum_{i=1}^{n} r_i(\theta) \nabla^2 r_i(\theta)$ [24].

Hence the quadratic model of the function $\phi(\theta)$ around the current joint configuration $\theta_i$ can be written as

$$Q_i(\theta) = \frac{1}{2} r(\theta_i)^T r(\theta_i) - (\theta - \theta_i)^T J(\theta_i)^T r(\theta_i)$$

$$+ \frac{1}{2} (\theta - \theta_i)^T \left[ J(\theta_i)^T J(\theta_i) + S(\theta_i) \right] (\theta - \theta_i)$$

(2.14)
Therefore, minimizing $Q_t(\theta)$ using Newton’s method results in the step [24]

$$\theta - \theta_t = (J_t^T J_t + S_t)^{-1} J_t^T \hat{r}_t$$  \hspace{1cm} (2.15)

where $J_t = J(\theta_t)$ and $S_t = S(\theta_t)$. Or, once again dividing both sides of (2.15) by $\Delta t$ and dropping the suffix $t$ gives

$$\dot{\theta} = (J^T J + S)^{-1} J^T \hat{r}$$  \hspace{1cm} (2.16)

Note that although eq.(2.16) is written in terms of velocities, we still need to know the final goal position $\bar{x}^d$ in order to compute the matrix $S(\theta)$.

If eq.(2.15) is employed in the example illustrated in section 2.4 with the goal position specified as $[2.1 \ 0.0]$, the manipulator settles into the configuration $[0^\circ \ 0^\circ \ 0^\circ]^T$ as desired. The sequence of configurations is

$$\begin{bmatrix}
5.0 \\
-5.0 \\
0.0
\end{bmatrix}
\begin{bmatrix}
-0.0127 \\
0.0127 \\
0.0
\end{bmatrix}
\begin{bmatrix}
0.0 \\
0.0 \\
0.0
\end{bmatrix}$$

The quadratic model is thus able to zero in on the external singularity because of the second-order information that it contains.

The above discussion and examples suggest that non-convergence and oscillations caused by the linear model can be avoided by employing the quadratic model (2.14) instead. Hence a strategy has to be devised to switch to the quadratic model whenever the Gauss-Newton model is unable to move the end-effector closer to $\bar{x}^d$.

Notice that the use of the Gauss-Newton model and the Levenberg-Marquardt iteration [(2.4) or (2.8)] means that the Hessian $J^T J + S$ of the function $\phi(\theta)$ is being approximated by the matrix $J^T J$ (or $J^T J + \lambda I$ in the case of non-zero damping). If $\bar{x}^d$ lies within the reachable workspace of the manipulator, then at the solution $r(\theta)$ becomes zero and hence the matrix $S$ becomes very small near the solution. Therefore, the Gauss-Newton approximation of the Hessian will be valid.

However if $\bar{x}^d$ is unreachaible, then $J^T J + S$ will not in general be well approximated by $J^T J$ near the solution. It is in these situations that the algorithm needs to switch to the Hessian method in order to make sure that the arm converges to the external singularity which minimizes $\phi(\theta)$.

When the goal point is outside the workspace, an external singularity minimizes the function $\phi(\theta)$ since no other configuration can place the end-effector closer to the
goal point. (If the workspace is convex, as is usually the case for revolute-joint robotic manipulators, this will be a unique global minimizer). Hence the gradient (2.12) will be zero at this configuration and the Hessian (2.13) will be positive definite in some neighborhood around the singularity. Therefore, monitoring the gradient and the Hessian of $\phi(\theta)$, will help in detecting that the arm is near the minimizing configuration, in which case, the quadratic model can be employed. However in order to ensure that the algorithm does not switch erroneously to the Hessian method, we need to make sure that the same does not happen at any other point within the workspace at which the gradient becomes zero. Note that when $r \neq 0$, the gradient $J^T r$ being zero implies that the arm is in a singular configuration.

The following theorem shows that even if the gradient $J^T r$ is zero at any point in the interior of the manipulator workspace, if $r \neq 0$, then that point is not a minimizer (i.e. the Hessian is not positive definite around that point). Or, in other words, a point within the manipulator workspace is a minimizer if and only if it corresponds to the desired end-effector position.

2.6 Non-existence of local minimizers

Let $W \subset \mathbb{R}^m$ be the interior of the manipulator workspace. It is assumed that every point within the workspace can be reached by the manipulator end-effector i.e. a joint configuration that places the end-effector at the point, exists.

Theorem 2.1

1. If $x^d \in W$, then the only point at which $J^T r = 0$ and $(J^T J + S)$ is positive definite is the point at which $r = 0$ i.e. $x^d = x$.

2. If $x^d \notin W$, then there is no point in the interior of $W$ at which $J^T r = 0$ and $(J^T J + S)$ is positive definite.

Proof (By contradiction)

Suppose there exists a joint configuration $\vartheta^*$ (with $x^* = f(\vartheta^*)$ and $x^* \in W$) at which $J^T r^* = 0$ and $(J^T J + S)$ is positive definite with $r^* = x^d - x^* \neq 0$. Hence $\vartheta^*$ is a local minimizer of the function $\phi(\vartheta)$ defined in eq.(2.11) i.e. there exists an open convex set $\mathcal{D}$ with $\vartheta^* \in \mathcal{D}$ such that for all $\vartheta \in \mathcal{D}$,

$$\phi(\vartheta^*) < \phi(\vartheta) \Rightarrow \|x^d - x^*\| < \|x^d - x\|$$
where $\mathbf{x} = f(\theta)$ and $\mathbf{x}^* = f(\theta^*)$.

Define

$$\hat{\mathbf{x}} = \mathbf{x}^* + \alpha(\mathbf{x}^d - \mathbf{x}^*)$$

(2.17)

where $\alpha$ is a positive scalar with $|\alpha| < 1$ chosen such that $\hat{\mathbf{x}}$ lies inside the workspace (fig. 2.4). Such an $\alpha$ exists because $\mathbf{x}^*$ lies in the interior of the manipulator workspace, by assumption.

Consider the hypersphere $S \subset \mathbb{R}^m$ defined by

$$\|\hat{\mathbf{x}} - \mathbf{x}^*\| = \alpha\|\mathbf{x}^d - \mathbf{x}^*\|$$

(2.18)

Then, $\mathbf{x}^* \in S$ since $\|\hat{\mathbf{x}} - \mathbf{x}^*\| = \|\alpha(\mathbf{x}^d - \mathbf{x}^*)\| = \alpha\|\mathbf{x}^d - \mathbf{x}^*\|$. 

**Lemma 2.1** For any point $\mathbf{x} \neq \mathbf{x}^*$ on $S$ i.e. satisfying (2.18),

$$\|\mathbf{x}^d - \mathbf{x}\| < \|\mathbf{x}^d - \mathbf{x}^*\| = \|\mathbf{R}^*\|$$

Thus, for some end-effector trajectory $\mathbf{z}(s)$ on $S$, passing through $\mathbf{z}^* = x(s^*)$, if there exists a continuous trajectory $\theta(s)$ corresponding to $\mathbf{z}(s)$ around $s^*$, i.e. $\mathbf{z}(s) = f(\theta(s))$ for all $s$ in an interval containing $s^*$, then this implies that there exists a joint configuration arbitrarily close to $\theta^*$ which takes the end-effector closer to $\mathbf{z}^d$ than does $\theta^*$.

At $\theta^*$, by assumption we have

$$J^T \mathbf{R}^* = 0 \Rightarrow \mathbf{R}^* \in \text{null}(J^T) \perp \text{range}(J)$$
which means that at this joint configuration, there is no joint velocity vector \( \dot{\theta} \) which satisfies \( \ddot{x} = J \ddot{\theta} \) for \( \ddot{x} = \beta R^* \) where \( \beta \) is a non-zero scalar. However, assuming \( J \neq 0 \), there does exist a \( \dot{\theta} \) which will realize an end-effector velocity \( \dot{x} \) which is in range(\( J \)). Then such an \( \dot{x} \) has to be orthogonal to \( R^* \) as range(\( J \)) \( \perp \) null(\( J^T \)).

Therefore, in order to be able to find a trajectory \( \theta(s) \) corresponding to any trajectory \( \bar{x}(s) \) on \( S \) and passing through \( \bar{x}^* = \bar{x}(s^*) \), we must have \( \dot{x}(s^*) \) orthogonal to \( R^* \). The following lemma proves that this is indeed the case.

**Lemma 2.2**

\[
\dot{x}(s^*)^T R^* = 0
\]

Thus, we can choose a trajectory \( x(s) \) lying on \( S \) and passing through \( x(s^*) \) such that \( \dot{x}(s^*) \in \text{range}(J) \). Then there exists a joint velocity \( \dot{\theta}(s^*) \) such that \( \dot{x}(s^*) = J \dot{\theta}(s^*) \) which implies that we can choose a trajectory \( \theta(s) \) such that \( x(s) = J(\theta(s)) \) and \( \theta(s) \) is continuous at \( s = s^* \) i.e. for a given convex set \( \mathcal{D} \) containing \( \theta^* \), there exists a \( \delta > 0 \) such that for \( |s - s^*| < \delta \), \( \theta(s) \in \mathcal{D} \). But by (A.4), for all such \( s \),

\[
||x^d - x(s)|| < ||x^d - x(s^*)|| \quad \text{i.e.} \quad \phi(\theta(s)) < \phi(\theta^*)
\]

Therefore, there exists a joint configuration \( \theta \in \mathcal{D} \) for which \( \phi(\theta) < \phi(\theta^*) \). Thus, \( \theta^* \) cannot be a minimizer of \( \phi(\theta) \).

The above analysis has shown that it is not possible for \( J^T R \) to be zero and \( J^T J + S \) to be positive definite if \( r \neq 0 \). This means that if the desired end-effector position is within the workspace, then the only minimizer(s) that \( \phi(\theta) \) can have are those for which \( r = 0 \) i.e. those which place the end-effector at the desired position. If the desired end-effector position is not within the manipulator workspace, then \( \phi(\theta) \) will have no minimizers inside the workspace (since \( r \neq 0 \) inside the workspace) and hence all minimizers will correspond to the end-effector being on the boundary of the workspace.

To illustrate the statement of the theorem, consider the three-link planar arm with unit link lengths. Suppose the goal position is specified as \([3.1 \ 0.0]\), which is outside the reachable workspace of the manipulator. Then, the joint configurations shown in fig. 2.5 are all those that correspond to the vector \( J^T \) being zero. The stars marked goal indicate the fact that the end-effector position lies to the right of each configuration. Table 2.6 shows the eigenvalues of the Hessian matrix for each of these configurations.

Thus, the Hessian is positive definite only for configuration E, which is the minimizing configuration. It is interesting to note that the negative eigenvalues for each
configuration correspond to that joint, whose motion will not take the end-effector further away from the goal position.

One issue that we have not addressed explicitly is that if the goal position is exactly on the boundary of the workspace, the Hessian corresponding to the external singularity that places the end-effector at the goal position will be singular since the residuals are zero and the Jacobian does not possess full row rank. However, in all the simulations we have run till now, this has not posed any problems, as the tolerance limit for the error is always reached before the matrix becomes numerically singular. Even with a task space error limit of $10^{-11}$, the algorithm with the switching strategy described below, converges with the desired accuracy!
2.7 Switching strategy

Using the theorem proved above, we can devise a strategy to choose between the two models [(2.5) and (2.15)], which will ensure convergence to the minimizer irrespective of whether the goal point is within the workspace or not.

Initially, the pseudoinverse/damped least-squares approach is employed. At each iteration, the value of $||\mathbf{L}_t||, ||J_t^T\mathbf{L}_t||$ as well as that of $red(t) = ||T(\theta_t)|| - ||T(\theta_{t+\Delta_t})||$ is monitored. If $red(t)$ is greater than a certain threshold value $\epsilon_1$ which can be a small positive number or even zero, this means that the Gauss-Newton model is taking the end-effector closer to the desired position and hence that step is accepted (note that this will be the case as long as $||J_t^T\mathbf{L}_t||$ is not nearly zero). If $||\mathbf{L}_t||$ drops below a certain positive threshold $\epsilon_2$, this implies that the goal point can be reached and the Hessian method does not have to be employed. If $red(t)$ drops below $\epsilon_1$ (which means that $||J_t^T\mathbf{L}_t|| \approx 0$), and $||\mathbf{L}_t|| > \epsilon_2$, then this indicates that the goal point may not be reachable.

Hence, in this situation the Hessian is computed and checked for positive definite-ness. If it is positive definite, the quadratic model is employed since by the theorem in the previous section, the manipulator must be in the neighborhood of the solution (as the gradient is close to zero and the Hessian is positive definite). If the Hessian is not positive definite this indicates that the manipulator may not have come close enough to the minimizing configuration. Hence, in this case, the trust region radius is reduced (in order to prevent oscillations) and a new step is computed. This is repeated until the Hessian becomes positive definite, after which the quadratic model is used.

2.8 Simulations

In this section, we present simulations of a 3-link planar arm with unit link lengths whose inverse kinematics is carried out using the strategy outlined above.

The manipulator is started from the configuration $[5^\circ \ -3^\circ \ -1^\circ]$. The goal position is given as $[3.5 \ 0.0]$ which is outside the manipulator workspace.

Fig.(2.6) shows plots of the norm of the difference between the goal position and the end-effector position at each successive iteration for the cases when the inverse kinematics is carried out using the linear model by itself and when the adaptive algorithm is used. As seen from the plot, when only the first order model is used, the manipulator does not converge to any configuration but instead keeps oscillating.
about the minimizing configuration. The algorithm was stopped after 150 iterations. The bottom half of the figure shows the task space error when the quadratic augmentation is used. In this case, the switch from the linear model to the quadratic model takes place after 6 iterations. The manipulator converges to the position [0° 0° 0°] in 7 iterations. The values of $\epsilon_1$ and $\epsilon_2$ used were 0 and 0.1 respectively while convergence occurred when the task space error was below $10^{-4}$.

In the second simulation, the manipulator is started close to an internal singularity with the configuration [5° -175° 175°]. The goal point specified is [2.5 0.0] which is within the workspace. In this case, the damped least squares method and the adaptive algorithm give the same performance as the algorithm uses the first order model throughout the trajectory. Only in the last iteration, it switches to the Hessian method. The goal position is reached in 54 iterations. Fig. 2.7 shows the evolution of the manipulator and the corresponding task space error. It is interesting to note that if the Hessian method is employed close to the initial configuration, the manipulator converges to the internal singularity. The sequence of joint configurations is

\[
\begin{bmatrix}
5.0 \\
-175.0 \\
175.0
\end{bmatrix}
\begin{bmatrix}
-0.0651 \\
-180.1436 \\
180.1436
\end{bmatrix}
\begin{bmatrix}
0.0 \\
-180.0 \\
180.0
\end{bmatrix}
\]

Thus, it is important that the adaptive algorithm not switch to the Hessian method under such circumstances.

2.9 Conclusions

This chapter has described a numerical method of computing the inverse kinematics of a general robotic manipulator - nonredundant or redundant. The method adaptively chooses between a first-order and second-order model of the direct kinematic function of the manipulator. The algorithm is insensitive to the reachability of the desired position i.e. it converges to a joint configuration that places the end-effector at the closest possible distance from the desired goal, irrespective of whether the goal is within the manipulator workspace or not. Thus, convergence is ensured even if the goal position corresponds to a singular configuration of the manipulator. Based on the theorem proven in section 2.6, a strategy of switching between the two models to ensure convergence to a global minimizer can be devised. This algorithm will be especially useful in situations where there is a likelihood of the commanded end-effector position lying outside the manipulator's reachable workspace.
Figure 2.6 Example 1 - Linear model vs Adaptive algorithm
Figure 2.7  Example 2 - Starting close to an internal singularity
Chapter 3

Minimum Effort Inverse Kinematics

...you're going to get closer and closer to what Asimov was all about when he talked about robots. A robot becomes more and more robotic, the more and more it emulates the human being.

- Joseph Engelberger

3.1 Introduction

In Chapter 2, the problem of computing a vector of joint positions for a robot manipulator, corresponding to a (sequence of) specified end-effector position and orientation vectors was addressed. In this chapter, we focus our attention on redundant manipulators and explore new methods of computing the inverse kinematics for such manipulators at the velocity level i.e. computing a joint velocity vector $\dot{\theta}$ for a given end-effector velocity vector $\dot{x}$.

3.1.1 Resolved Motion Rate Control

Differential kinematics, introduced by Whitney [105], utilizes differential relationships to solve for joint motions, given a desired end-effector trajectory. This technique, which is termed resolved motion rate control, uses a local linearization of the nonlinear function $f$ to compute the inverse kinematic solution. An important advantage of this method is that the linear relationship used, allows the utilization of general techniques for determining the solution, which do not depend on the specific kinematic design of the robotic manipulator.

As seen previously, the end-effector velocity vector $\dot{x} \in \mathbb{R}^m$ and the joint velocity vector $\dot{\theta} \in \mathbb{R}^n$ of a robotic manipulator are related through the Jacobian matrix $J \in \mathbb{R}^{m \times n}$ of the direct kinematic function $f(\theta)$ as

$$\dot{x} = J \dot{\theta} \quad ; \quad J(\theta) \triangleq \frac{\partial f}{\partial \theta} \in \mathbb{R}^{m \times n} \quad (3.1)$$
Computation of the inverse kinematic solution at the velocity level involves the
determination of a joint velocity vector $\dot{\theta}$ for a given end-effector velocity $\dot{x}$ at a joint
configuration $\theta$. Although (3.1) gives an analytical definition of the Jacobian matrix
$J$, computationally it is more efficient to construct this matrix using the vectors
defined by the Denavit-Hartenberg notation [94].

3.1.2 Redundant Manipulators

As seen in chapter 1, a robotic manipulator is said to be redundant if the dimension
$n$ of its joint space (equal to the number of joints of the manipulator) is greater than
the dimension $m$ of its task space (usually, equal to 3 in the planar case and 6 in the
spatial case). The advantages of redundancy in a manipulator can be categorized in
two primary ways [9] -

- **Design** - Redundancy can be exploited in the mechanical design of the ma-
nipulator to overcome kinematic, mechanical and other design limitations and
optimize dexterity and workspace measures.

- **Self Motion** - The extra degrees of freedom in the joint space lend the manipu-
lator the property of self motion i.e. the ability to move the joints while keeping
the end-effector stationary. This can be utilized in simultaneously satisfying
additional constraints such as obstacle avoidance while tracking the specified
end-effector trajectory.

Although redundant manipulators enjoy the aforementioned advantages over non-
redundant ones, the presence of extra joints makes the analysis and control of the
former more complex than the latter. In particular, inverse kinematics of redundant
manipulators poses a challenging (and interesting !) problem because, for any given
end-effector velocity, there usually exist infinite number of solutions (except in special
cases) ([79], [92]). This is because the equations in (3.1) are underdetermined (i.e.
they contain more unknowns than equations) as $m < n$. In fact, this is the reason
that the manipulator can perform self-motion. Hence, to obtain a joint velocity vector
for a given end-effector velocity, it is necessary to determine some optimality criterion
which will choose from among the infinite possible feasible vectors.
3.1.3 The Pseudoinverse Solution

The Moore-Penrose pseudoinverse $J^+$ of the Jacobian matrix is commonly employed to compute the joint velocities of a redundant manipulator as

$$\hat{\theta}^+ = J^+ \ddot{x}$$

This solution uses the Euclidean norm (or two-norm) as the optimality criterion i.e. it solves the problem

$$\min \|\hat{\theta}\|_2$$
subject to $$\min \|\ddot{x} - J\hat{\theta}\|_2$$

where $\|z\|_2 = (z^T z)^{1/2}$ is the 2-norm of the vector $z$ (see section 3.3). When the Jacobian has full row rank (= $m$), the pseudoinverse can be written analytically as

$$J^+ = J^T (J J^T)^{-1}$$

(3.2)

A common variation of this solution is the weighted pseudoinverse which minimizes a weighted two-norm. Thus, if $D$ is a positive definite matrix, for any vector $z$, the product $z^T [D] z$ is a norm as it satisfies all the required properties. The solution to the problem

$$\min \hat{\theta}^T [D] \hat{\theta}$$
subject to $$\min \|\ddot{x} - J\hat{\theta}\|_2$$

is given by

$$\hat{\theta}^+_D = J_D^+ \ddot{x} = D^{-1} J^T (J D^{-1} J^T)^{-1} \ddot{x}$$

(3.3)

Note that if $D$ is taken to be the inertia matrix of the manipulator, then the product $(\hat{\theta}^T [D]\hat{\theta})$ represents the instantaneous kinetic energy of the manipulator and so, the solution $\hat{\theta}^+_D$ represents the joint velocity that minimizes the instantaneous kinetic energy. Thus, the pseudoinverse can be considered to approximately minimize instantaneous power [47]. However, the more typical way of interpreting the pseudoinverse solution is the fact that it is a minimum-norm solution (the norm in this case being the Euclidean norm or the 2-norm) and hence is expected to yield "low" joint velocities.
3.2 Motivation for using the $\infty$-norm

One of the main reasons for the popularity of the 2-norm as an optimality criterion, is the fact that the related optimization problems yield closed-form analytical expressions such as (3.2) and (3.3), which can be conveniently computed using well-known techniques (such as the Singular Value Decomposition) ([73], [108]). Thus, in many problems, the two-norm is utilized, not because it best represents the physical constraint but because of its analytical tractability. However, in certain instances, criteria defined using the Euclidean norm may not represent the real problem that we are trying to solve.

One such case is when we want to obtain the lowest possible magnitudes of joint velocities that will perform the given task. The pseudoinverse solution is not the ideal choice for satisfying this constraint even though it minimizes the Euclidean norm of the joint velocities. This is because, minimizing the two norm minimizes the sum of squares of the joint velocities and hence will not necessarily minimize the magnitudes of the individual joint velocities i.e. there could be unequal distribution of the ‘energy’ resulting in a relatively high velocity for a particular joint. This will be undesirable in situations where the individual joint velocities (and not the value of the 2-norm itself) are of primary interest.

The following simple example serves to illustrate this point. Consider a two-link manipulator with prismatic joints connected as shown in fig.(3.1). The first link can move back and forth at an angle of 60° to the x-axis while the second link has a translational motion parallel to the x-axis.

The task space of this manipulator is taken to be the horizontal line i.e. in a direction parallel to the x-axis. The one-dimensional task space gives the manipulator a single degree of redundancy. The Jacobian matrix corresponding to this task space is

$$J = \begin{bmatrix} 0.5 & 1 \end{bmatrix}$$ \hspace{1cm} (3.4)

If the desired end-effector velocity $\dot{x}$ is 1 unit/sec, then the pseudoinverse solution is

$$\dot{\theta}^+ = \begin{bmatrix} 0.400 & 0.800 \end{bmatrix}^T$$ \hspace{1cm} (3.5)

However, the following joint velocity vector also gives the same end-effector velocity

$$\dot{\theta}^* = \begin{bmatrix} 0.667 & 0.667 \end{bmatrix}^T$$ \hspace{1cm} (3.6)

and calls for a maximum joint velocity which is 20% less than that of $\dot{\theta}^+$. On the other hand, the Euclidean norm of $\dot{\theta}^*$ is only 5.5% greater than that of $\dot{\theta}^+$. Note
that the equal velocities in the $\hat{\theta}^*$ solution mean that the task is distributed in the same proportion as the relative ability of the joints to perform the task at the given configuration (as defined by the columns of the Jacobian). For instance, in this case, the ability of the first joint to accomplish the task (with equal effort or velocity) is half that of the second (from (3.4), $J_{11}/J_{12} = 0.5$). The $\hat{\theta}^*$ solution distributes the task in this same ratio since from (3.6), the contribution of the first joint towards the end-effector task is 0.333 units/sec (= $j_1\hat{\theta}_1^*$) while that of the second joint is 0.667 units/sec (= $j_2\hat{\theta}_2^*$). On the other hand, the joint velocities in the pseudoinverse solution $\hat{\theta}^+$ in (3.5) are in the same proportion as their abilities to perform the task, since $\hat{\theta}_1^+ / \hat{\theta}_2^+ = 0.5$. This results in a more disproportionate distribution of the task itself, as $j_1\hat{\theta}_1^+ = 0.2$ and $j_2\hat{\theta}_2^+ = 0.8$, which means that the contribution of the first joint is one-fourth that of the second. This simple example illustrates the fact that minimization of the two-norm will not produce the best results when low magnitudes of joint velocities and equity in task distribution are desired. This calls for the use of a different optimality criterion or a different norm.

In the robotics literature, there has been very little attention devoted to optimizing any norms other than the two-norm. (One notable exception is [47], where the definition of subtasks using p-norms is discussed - see section 3.10). As mentioned above, this is largely due to the fact that the solutions obtained using two-norm criteria yield elegant analytical results that can be computed using well-established methods. The objective of this chapter is to explore the optimization of the $\infty$-norm
as it can deal with individual joint velocities better than the Euclidean norm. The \( \infty \)-norm of a vector is defined as the maximum absolute value of its components (see eq. (3.9) below). Thus, computing an inverse kinematic solution by minimizing this norm will yield a joint velocity vector that will have minimum absolute value of its largest component. Such a solution can thus be thought of as a *minimum-effort* solution as opposed to the *minimum-energy* solution yielded by the pseudoinverse. (The solution \( \hat{\theta}^* \) in the above example is the minimum \( \infty \)-norm solution for the specified problem).

The main issues that we address in this chapter are -

- Algorithms for computing a minimum \( \infty \)-norm solution (sections 3.5, 3.6, 3.7).
- Reducing the \( \infty \)-norm of solutions optimizing subtask constraints such as obstacle avoidance, when there is more than one extra joint (section 3.9).
- Using the \( \infty \)-norm to define subtask criteria that better represent the physical constraints on the manipulator (section 3.10)

### 3.3 Definitions and notation

This section sets up the basic definitions and notation that will be used in the following sections.

- For any vector \( \mathbf{z} = [z(1), z(2), \ldots, z(n)]^T \in \mathbb{R}^n \), the family of norms known as \( p \)-norms for \( p \geq 1 \) is defined as [37]

\[
\|\mathbf{z}\|_p = \left( \sum_{i=1}^{n} |z(i)|^p \right)^{1/p}
\]

In particular, we will be concerned with the 1-norm, the 2-norm and the \( \infty \)-norm.

- \( p = 1 : \|\mathbf{z}\|_1 = \sum_{i=1}^{n} |z(i)| \) (3.7)
- \( p = 2 : \|\mathbf{z}\|_2 = \sqrt{\sum_{i=1}^{n} |z(i)|^2} \) (3.8)
- \( p = \infty : \|\mathbf{z}\|_{\infty} = \max_{i=1,\ldots,n} |z(i)| \) (3.9)
• The space of all vectors in $\mathbb{R}^n$ with the associated (finite) $l_\infty$-norm is denoted as the $l^n_\infty$-space and similarly the space of all real $n$-tuples with the associated (finite) $l_1$-norm is denoted as the $l^n_1$-space.

• In general for any $\mathbf{w} \in l^n_1$ and $\mathbf{z} \in l^n_\infty$, the inequality $\mathbf{w}^T \mathbf{z} \leq ||\mathbf{w}||_1 ||\mathbf{z}||_\infty$ holds. However if the inner product of $\mathbf{w}$ and $\mathbf{z}$ is equal to the product of their norms then the two vectors are said to be aligned i.e.

$$\mathbf{w}^T \mathbf{z} \triangleq \sum_{i=1}^{n} w(i)z(i) = ||\mathbf{w}||_1 ||\mathbf{z}||_\infty$$

Notice that alignment is a relation between two vectors in two distinct vector spaces - a normed space and its normed dual (in this case $l^n_\infty$ and $l^n_1$ respectively). The concept of alignment [56] is motivated by the result in $l_2$ space (or Hilbert spaces, in general) that equality in the Cauchy-Schwarz inequality holds if and only if one vector is a scalar multiple of the other.

**Lemma 3.1** Given a nonzero vector $\mathbf{w} \in l^n_1$, it can be easily shown that (see appendix A) only vectors of the following form in $l^n_\infty$ are aligned with $\mathbf{w}$,

$$z(i) = \alpha \cdot \text{sgn}[w(i)] \quad \text{if} \quad w(i) \neq 0$$

$$\quad = \alpha_i \quad \text{if} \quad w(i) = 0$$

(3.10)

where $\alpha \geq 0$ and $|\alpha_i| \leq \alpha$.

• For any vector $\mathbf{z} \in \mathbb{R}^n$, the corresponding sign vector consists of the signs of its components i.e. if $\mathbf{z} = \text{sgn}[z]$, then

$$\sigma_i = \begin{cases} 1 & \text{if } z_i > 0 \\ -1 & \text{if } z_i < 0 \\ 0 & \text{if } z_i = 0 \end{cases}$$

• In the following analysis, we will assume that the equations $\dot{\mathbf{x}} = J\dot{\mathbf{a}}$ are consistent i.e. the Jacobian $J$ has full row rank $m$ (see end of section 3.5).

### 3.4 Dual Optimization Problems

In order to find a minimum infinity-norm inverse kinematic solution, we wish to solve the problem [30]

$$\min ||\dot{\mathbf{a}}||_\infty \quad \text{subject to} \quad \dot{\mathbf{x}} = J\dot{\mathbf{a}}$$

(3.11)
i.e. among all the vectors in $l_\infty^n$, satisfying eq.(3.1), we need to determine one which has minimum $\infty$-norm. We make use of the following theorem from functional analysis [56] to formulate an equivalent optimization problem in the dual space i.e. $l_1^n$. The solution to (3.11) is then determined from the solution to the dual problem. (A proof of a general version of this theorem can be found in [56]).

**Theorem 3.1** Given the system (3.1) of $m$ consistent equations in $n$ unknowns, the following equality exists

$$\min_{\dot{\theta} = \hat{\theta}} \|\dot{\theta}\|_\infty = \max_{\|J^T E\|_1 \leq 1} \dot{\theta}^T E$$  \hspace{1cm} (3.12)

*(Primal Problem) \hspace{1cm} (Dual Problem)*

Furthermore, the optimal $\hat{\theta}^*$ is aligned with the optimal $J^T E^*$.

Here $\hat{\theta}^* \in \mathbb{R}^n$ and $E^* \in \mathbb{R}^m$ are the solution vectors for the primal and dual problems respectively. The theorem states that the $\infty$-norm of the minimum infinity norm inverse kinematic solution $\hat{\theta}^*$, is equal to the maximum value of the inner product $\dot{\theta}^T E$ over all $E \in \mathbb{R}^m$ for which $\|J^T E\|_1 \leq 1$. Note that the physical dimensions of the objective functions of the primal and dual problems are the same, with both having units of work as shown below (eq.(3.13)).

Physically, this theorem can be interpreted in the following manner. The inner product $\dot{\theta}^T E$ represents the work done by the end-effector when moving with velocity $\dot{\theta}$ and exerting a force $E$ on its environment. Also, statically, the joint torque vector $\tau \in \mathbb{R}^n$ (representing the torque inputs at the manipulator joints) required to generate a force $E$ at the end-effector is given by $\tau = J^T E$ [94]. Thus, the dual problem represents the maximization of the work done by the end-effector, for the given $\dot{\theta}$, over all force vectors $E$ in $\mathbb{R}^m$ which are generated by torque vectors lying within the unit ball in $l_1^n$ space (i.e. $\|\tau\|_1 = \|J^T E\|_1 \leq 1$). It can be shown in fact that for the $E^*$ which solves the dual problem, $\|\tau^*\|_1 = \|J^T E^*\|_1 = 1$ (see theorem 3.2 below).

By the principle of virtual work we know that

$$\dot{\theta}^T E^* = (\tau^*)^T \hat{\theta}^*$$

and hence using the alignment condition in theorem 3.1,

$$(\tau^*)^T \hat{\theta}^* = \|\tau^*\|_1 \|\hat{\theta}^*\|_\infty = (1)(\|\hat{\theta}^*\|_\infty) = \|\hat{\theta}^*\|_\infty$$  \hspace{1cm} (3.13)

Hence the equality between the primal and dual problems represents the equality in the optimal work done by the joints and the end-effector at any instant.
Note that, although the above interpretation has introduced static force and torque into the picture, it has been done only to facilitate the understanding of the dual relation. Our problem is of course, only kinematic in nature.

Thus the solution to problem (3.11) can be obtained by determining \( \mathcal{F}^\ast \) and then using the alignment condition in theorem (3.1) to compute \( \hat{\vartheta}^\ast \). The following theorem, proven in [12], facilitates the determination of \( \mathcal{F}^\ast \).

**Theorem 3.2** Given the \( m \times n \) matrix \( J \) of rank \( r \leq m \) and the vector \( \hat{x} \in \mathbb{R}^n \), there exists a vector \( \mathcal{F}^\ast \in \mathbb{R}^m \) such that

\[
\hat{x}^T \mathcal{F}^\ast = \max_{\|J^T \mathcal{F}\|_1 \leq 1} \hat{x}^T \mathcal{F} = \max_{\|J^T \mathcal{F}\|_1 = 1} \hat{x}^T \mathcal{F} \tag{3.14}
\]

and at least \( r - 1 \) components of \( J^T \mathcal{F}^\ast \) are equal to zero with the corresponding columns of \( J \) linearly independent.

Thus, by the above theorem, if \( j_k \) denotes the \( k \)-th column of \( J \),

\[
j_i^T \mathcal{F}^\ast = 0 \quad \text{for} \quad i \in \{i_1, i_2, \ldots, i_{r-1}\}
\]

with \( 1 \leq i_k \leq n \) and \( i_k \neq i_j \) for \( j \neq k \). Moreover, the set of vectors in \( \mathbb{R}^m \), \( \{j_{i_1}, j_{i_2}, \ldots, j_{i_{r-1}}\} \) are linearly independent.

Hence assuming that the Jacobian matrix has rank \( m \), each feasible solution \( E_i \) to the dual problem is characterized by

1. It is orthogonal to a specific set of \( m - 1 \) linearly independent columns of \( J \).

2. It has to satisfy the condition \( \|J^T E\|_1 = 1 \)

(In all the following discussion we assume without loss of generality that the rank of \( J \) is \( m \). If the rank of \( J \) is equal to \( r < m \), then replace \( m \) by \( r \).)

For a specific set of \( m - 1 \) linearly independent columns of \( J \), the above two conditions yield exactly two vectors, \( \pm E_i \). Hence the set of all feasible solution vectors is

\[
\mathcal{F} = \{\pm E_1, \pm E_2, \ldots, \pm E_N\}
\]

where \( N \leq {n \choose m-1} \).

Thus, a direct method of computing \( \mathcal{F}^\ast \) is
1. For each set of $m - 1$ linearly independent columns of $J$, compute a vector $\hat{f}_i$ which is orthogonal to all of them.

2. The feasible solution vector $F_i$ is then generated from $\hat{f}_i$ as

$$
F_i = \left( \frac{\text{sgn}(\hat{x}^T \hat{f}_i)}{\| J^T \hat{f}_i \|_1} \right) \hat{f}_i
$$

The multiplication by $\text{sgn}(\hat{x}^T \hat{f}_i)$ ensures that $\hat{x}^T F_i$ is positive (and hence greater than $-\hat{x}^T F_i$) and the division by $\| J^T \hat{f}_i \|_1$ makes $\| J^T F_i \|_1 = 1$.

3. Then the solution to the dual problem is given by that $F_i = F^*$ for which

$$
\hat{x}^T F^* = \max \{ \hat{x}^T F_1, \hat{x}^T F_2, \ldots, \hat{x}^T F_N \}
$$

4. The minimum-norm joint velocity vector can now be determined using the alignment condition in theorem 3.1 and the constraint $\hat{x} = J\hat{\theta}$ [12]. By Lemma 3.1 (page 40), the alignment condition determines the components of $\hat{\theta}$ corresponding to the non-zero components of $J^T F^*$ as

$$
\hat{\theta}^*(i) = (\hat{x}^T F^*)(\text{sgn}[J^T F^*])_i \quad \text{if} \quad [J^T F^*]_i \neq 0
$$

This specifies $(n - m + 1)$ elements of the joint velocity vector $\hat{\theta}^*$ and in effect, resolves the redundancy by providing the extra constraints needed in (3.1). The scale factor $\alpha$ in eq.(3.10) is equal to $\hat{x}^T F^*$ in eq.(3.15) as it is equal to the infinity norm (and hence the largest component) of $\hat{\theta}^*$ (by theorem 3.1). Thus, if the number of zero components of $J^T F^*$ is $r$ and $r = m-1$, the equation $\hat{x} = J\hat{\theta}$ reduces to a set of $m$ equations in $m-1$ unknowns which can be solved uniquely. If $r > m-1$, then the original $m$ constraint problem reduces to an $m-1$ constraint problem [12].

However the direct method can become time consuming when $N$ is large. The following algorithm [13] uses a strategy to choose sets of linearly independent columns in such a way that the quantity $\hat{x}^T F_i$, which we are trying to maximize is always increased for each new set of $m - 1$ columns.
3.5 Algorithm for computing the dual solution

In this section, an algorithm first proposed in [13] for minimum-amplitude control of linear discrete systems, is described. Adaptation of this algorithm for solving the inverse kinematic problem, as described here, produces an efficient means of computing the joint velocity vector with minimum \( \infty \)-norm. This is especially true when inverse kinematics for a continuous end-effector trajectory is being computed, since the algorithm can be made to converge in one iteration for almost the entire trajectory in this case.

The algorithm uses the following partitioning of the Jacobian matrix

\[
J = [J_1 \ J_2]
\]  
(3.16)

where \( J_1 \) is an \( m \times (m - 1) \) matrix while \( J_2 \) has dimensions \( m \times (n - m + 1) \). Then the equation \( \dot{x} = J\dot{\theta} \) can be written as the following sum

\[
\dot{x} = J_1\dot{\theta}_1 + J_2\dot{\theta}_2 = J\dot{\theta}
\]  
(3.17)

where \( \dot{\theta}_1 \in \mathbb{R}^{m-1} \) and \( \dot{\theta}_2 \in \mathbb{R}^{n-m+1} \) and \( \dot{\theta} = [\dot{\theta}_1^T \ \dot{\theta}_2^T]^T \).

The main strategy of the algorithm is then to interchange one column from \( J_1 \) with one column from \( J_2 \) such that the inner product \( \dot{x}^T F \) is increased. The following are the steps involved.

1. Partition the Jacobian matrix as in (3.16) such that the columns of \( J_1 \) are linearly independent.

2. Determine a nonzero vector \( \ell \) such that \( J_1^T \ell = 0 \).

3. Then a feasible force vector \( F \) (for the dual problem) is generated as

\[
F = \left( \frac{\text{sgn}(\dot{x}^T \ell)}{\| J^T \ell \|_1} \right) \ell
\]

It can be easily shown that with this choice of \( F, \| J^T F \|_1 = 1 \) and \( \dot{x}^T F > 0 > \dot{x}^T (-F) \).

4. Compute \( w = \dot{x}^T F = |\dot{x}^T \ell|/\| J^T \ell \|_1 \)

5. Compute \( \dot{\theta}_2 = w \ast \text{sgn}[J_2^T \ell] \). Then, from eq.(3.17), solve for \( \dot{\theta}_1 \) the equation

\[
J_1 \dot{\theta}_1 = \dot{x} - w J_2 (\text{sgn}[J_2^T \ell])
\]
6. Check for alignment between \( \tau = J^T \hat{E} \) and \( \dot{\hat{\theta}} \) as follows

(a) If \( \|\dot{\hat{\theta}}_1\|_\infty \leq w \), then \( \tau \) and \( \dot{\hat{\theta}} \) are aligned and \( \dot{\hat{\theta}} = [\dot{\hat{\theta}}_1^T \dot{\hat{\theta}}_2^T]^T \) is the desired solution. (Although this does not directly follow from theorem 3.1, it can be shown [13] that if \( E \) is a feasible solution vector and if \( J^T E \) and \( \dot{\hat{\theta}} \) are aligned, then \( \dot{\hat{\theta}} \) is an optimal solution of (3.11))

(b) If \( \|\dot{\hat{\theta}}_1\|_\infty > w \), then \( \tau \) and \( \dot{\hat{\theta}} \) are not aligned. Proceed to step 7.

7. Let \( \dot{\hat{\theta}}_1(p) \) be the largest component of \( \dot{\hat{\theta}}_1 \). Then the \( p \)-th column of \( J_1 \) is chosen to be interchanged with a column in \( J_2 \).

8. In this step, we need to determine a column from \( J_2 \) which must be interchanged with the \( p \)-th column of \( J_1 \) such that the value of \( w \) corresponding to the new \( J_1 \) and \( J_2 \) is greater than that corresponding to the previous one. Correspondingly, a perturbation vector \( \gamma \) for computing an updated force vector \( \bar{E} \) also needs to be determined. (For an explanation of the following steps (a) and (b), see page 46)

(a) Determine a vector \( \gamma \) by solving the \( m \) equations

\[
J_i^T \gamma = \text{sgn}(\dot{\hat{\theta}}_1(p))e_p \quad \text{and} \quad E_i^T \gamma = 0
\]

Here, \( e_p \) is a vector which has all its components zero, except the \( p \)-th one, which is equal to 1.

(b) Compute the ratios

\[
\sigma_i = -\frac{[J_i^T \gamma]}{[J_i^T E]}; \quad i = 1, 2, \ldots, n - m + 1
\]

Let \( \sigma_q \) be the largest of these ratios and let \( \epsilon_q = 1/\sigma_q \). Then the \( q \)-th column of \( J_2 \) is chosen for interchanging with the \( p \)-th column of \( J_1 \).

(see explanation below)

9. Interchange the \( p \)-th column of \( J_1 \) with the \( q \)-th column of \( J_2 \) to form the new partition of \( J = [J_1 \ J_2] \).

10. The feasible force vector \( \bar{E} \) is then computed as

\[
\bar{E} = \frac{\text{sgn}(\epsilon_q)}{\|J_2^T (E + \epsilon_q \gamma)\|_1} [E + \epsilon_q \gamma]
\]

(3.18)
The division by the term $||J_2^T(F + \epsilon \gamma)||_1$ normalizes $||J_2^T\overline{F}||_1 = 1$ while the multiplication by $sgn(\epsilon_q)$ ensures that $\tilde{\dot{\theta}}^T\overline{F} > \tilde{\dot{\theta}}^TF$ [13].

11. $J_1 = J_1; J_2 = J_2; F = \overline{F}$

Go to step 4.

Since the number of sets of $m-1$ linearly independent columns is bounded above by $m!C_{m-1}$ and as the quantity $w$ increases at each iteration, it follows that the above algorithm must converge in a finite number of iterations. (Note that $m!C_{m-1}$ is only an upper bound for the number of iterations. The algorithm converges in much fewer iterations as discussed on page 46).

Step 8 in the above algorithm requires some explanation. While determining the column in $J_2$ to be interchanged, we also need to find a new feasible solution vector $\overline{F}$. Hence the current feasible vector $F$ should be perturbed such that the resultant vector is orthogonal to all but the $p$-th column of $J_1$ and also is orthogonal to the column that is going to be chosen from $J_2$. Therefore the perturbation vector $\gamma$ is chosen orthogonal to all but the $p$-th column of $J_1$. In order to make the perturbed vector orthogonal to one of the columns of $J_2$ as well, consider

$$\overline{F} = F + \epsilon \gamma$$

Then, to determine $\epsilon$, we need to consider the various values of $\epsilon$ that force different components of $J_2^T\overline{F}$ to zero i.e. for $i = 1, \ldots, (n - m + 1)$,

$$[J_2^T\overline{F}]_i = [J_2^TF]_i + \epsilon \gamma_i[J_2^T\gamma]_i = 0$$

$$\Rightarrow \epsilon_i = -\frac{[J_2^TF]_i}{[J_2^T\gamma]_i}; \quad i = 1, \ldots, (n - m + 1)$$

It can be shown [13] that by choosing the value of $\epsilon$ as equal to that $\epsilon_q$ for which

$$\frac{1}{\epsilon_q} = max_{i = 1, \ldots, (n - m + 1)} \left( \frac{1}{\epsilon_i} \right)$$

we obtain $\overline{F}$ for which $\tilde{\dot{\theta}}^T\overline{F} > \tilde{\dot{\theta}}^TF$.

One of the main advantages of using this algorithm for computing the inverse kinematic solution is the fact that, when following a continuous end-effector trajectory (in discrete time steps), the solution can be determined in one iteration for most of the joint configurations. This is achieved by choosing the partitioning at the beginning of the algorithm (at each time step $k$, corresponding to the computation of $\tilde{\dot{\theta}}^*_k$) to be
the same as the one that corresponded to the solution \( \hat{\theta}^*_{k-1} \) at the previous time step. Since the perturbation in the Jacobian matrix will be small from one time step to the next and assuming that the rank of \( J(\theta) \) is preserved, the partitioning corresponding to the solution will change rarely. This will make the algorithm converge in one iteration for most of the trajectory (see examples in section 3.11). Another advantage is that computationally, too, several steps in the algorithm can be carried out efficiently with minimum cost. For example, steps 5 and 8 require the following two systems of equations to be solved

\[
\begin{bmatrix}
J_1 & E
\end{bmatrix}
\begin{bmatrix}
\hat{\theta}_1 \\
0
\end{bmatrix}
= \hat{\dot{\theta}} - w J_2 (\text{sgn}(J_2^T E))
\]

\[
\begin{bmatrix}
J_1^T & E^T
\end{bmatrix}
\gamma
= \begin{bmatrix}
e_p \\
0
\end{bmatrix}
\text{sgn}(\hat{\dot{\theta}}(p))
\]

Both systems can be solved simultaneously by the computation of a single inverse. Also, at most two columns of \( J_1 \) change at a time so that the inverse can be updated expeditiously too.

As shown in [10], this algorithm requires fewer than \([(3m + 1)(n - m + 1) + 2m^2 + 3m]\) multiplications per iteration. The number of additions required is less than \([3m(n - m + 1) + 2m^2 + 3m]\). Note that for a single degree of redundancy, this reduces to \((4m^2 + 18m + 2)\) floating point operations (flops) per iteration. Also, for a general linear problem, the algorithm typically converges in \([n + m]\) to \([2(n + m)]\) iterations. However, as discussed in the previous paragraph and as shown in the simulations (section 3.11), when tracking a continuous end-effector trajectory, the number of iterations is reduced to one, for most of the trajectory due to the continuity from one joint configuration to the next. Thus, the computational complexity of this algorithm compares favorably to that of computing the 2-norm solution. For instance, with the QR-factorization, the minimal 2-norm solution can be computed with \((2m^2 n - 2m^3/3)\) flops [37]. Hence, for \( m = 6 \) and \( n = 7 \), the computation of the minimum \( \infty \)-norm solution with the above algorithm requires 254 flops while the pseudoinverse solution with QR factorization requires 360 flops.

One implicit assumption made in the above algorithm is that every set of \( m - 1 \) columns of \( J \) is linearly independent. This assumption is not strictly necessary, in the sense that once the values of \( p \) and \( q \) are computed, i.e. the columns of \( J_1 \) and \( J_2 \) which have to be interchanged are decided, \( J_1 \) can be checked for full rank. If it does not have full rank, then some other component of \( \hat{\dot{\theta}} \) which has value greater than \( w \)
can be chosen (in step 7, we choose the component with the maximum value). Note that this situation will occur only for certain "special" configurations. Future work entails the characterization of such configurations.

Also, in the above analysis, the rank of \( J \) is assumed to be \( m \). Although theorem 3.2 holds good even if \( \text{rank}(J) \) is less than \( m \) [13], theorem 3.1 assumes consistency of the equations in (3.1). Hence, the above algorithm can be used at a singular configuration only if the desired end-effector velocity does not have any component in the singular direction.

For more details on the working of the algorithm, the interested reader is referred to [10], [13].

### 3.6 Alternate Method for Dual Solution

An alternate method of solving the dual problem is to transform it to that of finding a best \( l_1 \) approximation to a system of overdetermined linear equations [11]. Then the solution can be obtained using any one of several well-known techniques for solving such a problem. The dual problem is transformed as follows.

The Jacobian matrix can be uniquely decomposed as the sum \( J = B + C \), where the columns of \( C \) are orthogonal to \( \hat{\alpha} \) and the matrix \( B \) is of the form \( B = \hat{\alpha} \zeta^T \) with \( \zeta = (J^T \hat{\alpha})/(\hat{\alpha}^T \hat{\alpha}) \) [11]. The vector \( E \in \mathbb{R}^m \) can be correspondingly decomposed as \( E = \alpha \hat{\alpha} + p \) where \( \alpha \in \mathbb{R} \) and \( \hat{\alpha}^T p = 0 \). Then,

\[
\hat{\alpha}^T E = \alpha \hat{\alpha}^T \hat{\alpha}, \quad J^T E = \alpha B^T \hat{\alpha} + C^T E \tag{3.19}
\]

Substituting the expressions from (3.19) in the dual problem in (3.12), we get

\[
\max_{\|J^T E\|_1 \leq 1} \hat{\alpha}^T E = \left( \hat{\alpha}^T \hat{\alpha} \right) \max_{\alpha B^T \hat{\alpha} + C^T E, \|E\|_1 \leq 1} \alpha
\]

The constraint on the right hand side may be written as

\[
\|\alpha B^T \hat{\alpha} + C^T E\|_1 = |\alpha| \|B^T \hat{\alpha} + C^T E/\alpha\|_1 \leq 1 \quad \Rightarrow \quad |\alpha| \leq \frac{1}{\|B^T \hat{\alpha} + C^T E/\alpha\|_1} \tag{3.20}
\]

Thus, the dual problem reduces to

\[
\max_{\left[ |\alpha| \leq \|B^T \hat{\alpha} + C^T E/\alpha\|_1 \right]} \alpha
\]
From theorem 3.2, we know that the constraint (3.20) is satisfied with equality for the optimal solution. Therefore, the optimal solution corresponds to the maximum \( \alpha \) such that

\[
\alpha = \frac{1}{\|B^T \hat{z} + C^T E / \alpha\|_1}
\]  

(3.21)

The maximization of \( \alpha \) over all \( \alpha \) and \( E \) that satisfy constraint (3.21) is thus equivalent to minimizing the quantity in the denominator on the right-hand side of (3.21). Note that the term \( B^T \hat{z} \) and the matrix \( C \) are specified, so the only variable is the vector \( E / \alpha \). Hence, the solution to the dual problem is obtained by solving

\[
\min_{E \in \mathbb{R}^n} \|B^T \hat{z} + C^T E\|_1
\]  

(3.22)

Problem (3.22) can be solved by any one of several proposed algorithms for solving an overdetermined \( l_1 \) problem (e.g. [4], [11]). Once the solution \( E^* \) to the problem (3.22) is obtained, the solution \( \hat{u}^* \) of the primal problem can be computed using the alignment condition between \( (B^T \hat{z} + C^T E^*) \) and \( \hat{u}^* \) (see item 4 on page 43).

One advantage of this algorithm compared to the one described in the previous section is that it does not require the testing of linear independence of sets of \( m - 1 \) columns of \( J \). However, a disadvantage is that it does not lend itself to convergence in a single iteration, as does the other one, when the manipulator is following a continuous end-effector trajectory.

Both of these algorithms in sections 3.5 and 3.6 use the dual problem solution to obtain the minimum infinity-norm joint velocity vector. The algorithm described in the next section solves problem (3.11) directly by modifying any valid joint velocity vector (one that satisfies eq.(3.1)) using self-motion components to obtain the minimum infinity-norm solution.

### 3.7 Steepest Descent Algorithm

In this section, an algorithm is described that solves the primal problem (see eq.(3.12)) directly using an iterative procedure based on the steepest descent method for constrained optimization [50]. This method uses the null space gradient projection technique to compute the minimum infinity-norm solution. The gradient projection technique has been used by robotics researchers to optimize different performance indices in the joint space, for accomplishing various subtasks ([79], [92]). Here, the performance index is, in effect, defined to be the minimum infinity-norm of the joint
velocity vector. The algorithm described in this section iteratively chooses a scalar $\alpha$ multiplying the null space term such that the optimal solution is obtained.

The main strategy of the algorithm is as follows. Suppose $\hat{\theta}^1$ is an initial point satisfying eq.(3.1). Then, compute the gradient $g^1$ of the function

$$h(\hat{\theta}) = \|\hat{\theta}\|_\infty$$

(3.23)

Let $p^1$ be the orthogonal projection of the gradient $g^1$ on the nullspace of the Jacobian $J$. The joint velocity vector is then updated as $\hat{\theta}^2 = \hat{\theta}^1 + \alpha^1 p^1$, where the scalar $\alpha^1$ is selected so as to minimize $h(\hat{\theta}^2) = h(\hat{\theta}^1 + \alpha^1 p^1)$. Note that $\hat{\theta}^2$ automatically satisfies the constraint (3.1). $\hat{\theta}^2$ can now be taken as the initial point for the next iteration.

The following algorithm from [50], which describes a way of computing the scalar $\alpha^k$ at the $k$-th iteration, ensures convergence in $n-m$ iterations. In the description of the algorithm, the pseudoinverse solution has been used as the initial choice for $\hat{\theta}$. However, a different initial choice can be used for purposes of computational efficiency (e.g. a computationally cheaper solution such as a {1}-inverse [55] could be used) or for satisfaction of additional criteria (see section 3.9).

1. Let $k = 0$ and $J^k = J$. Compute

$$\hat{\theta}^0 = (J^k)^T (J^k (J^k)^T)^{-1} \hat{x}$$

(3.24)

2. Store the index $i_k$ corresponding to the maximum component of $\hat{\theta}^0$.

$$|\hat{\theta}_{i_k}^0| = \max_i |\hat{\theta}_i^0|$$

3. Let $s_k = \text{sign} \hat{\theta}_{i_k}^k$. Form the gradient vector $g^k \in \mathbb{R}^n$ with all components zero except the $i_k$-th one which is equal to $-s_k$.

4. Compute the orthogonal projection $p^k = [(I - J^k J^k)^T g^k]$ of the gradient $g^k$ on the nullspace of $J^k$.

$$p^k = g^k - s_k (J^k)^T [J^k (J^k)^T]^{-1} j_{i_k}^k$$

(3.25)

where $j_{i_k}^k$ is the $i_k$-th column of the matrix $J^k$.

5. For $i \neq i_k$, compute the coefficients $\alpha_i^k$ as follows (see explanation after algorithm)
(a) If sign $\hat{u}_t^k = \text{sign} \ p_t^k$, \quad $\alpha_t^k = (|\hat{u}_t^k| - |\hat{v}_t^k|)/(|\hat{u}_t^k| + |\hat{v}_t^k|)$

(b) If sign $\hat{u}_t^k \neq \text{sign} \ p_t^k$ and $|\hat{u}_t^k| > |\hat{v}_t^k|$, \quad $\alpha_t^k = (|\hat{u}_t^k| - |\hat{v}_t^k|)/(|\hat{u}_t^k| - |\hat{v}_t^k|)$

(c) If sign $\hat{u}_t^k \neq \text{sign} \ p_t^k$ and $|\hat{u}_t^k| \leq |\hat{v}_t^k|$, \quad $\alpha_t^k = (|\hat{u}_t^k| + |\hat{v}_t^k|)/(|\hat{u}_t^k| + |\hat{v}_t^k|)$

6. Choose the smallest coefficient $\alpha_{j_k} = \min \alpha_t^k$ and store its index $j_k$.

7. The joint velocity vector is then updated as $\hat{\phi}^{k+1} = \hat{\phi}_t^k + \alpha_{j_k} \hat{p}_t^k$

8. A reduced matrix $J^{k+1}$ for the next iteration is formed by removing the $i_k$-th column and modifying the $j_k$-th column, as follows -

(a) If $i \neq i_k, j_k$, then $\hat{j}_t^{k+1} = \hat{j}_t^k$.

(b) If sign $\hat{u}_{i_k}^{k+1} = \text{sign} \ \hat{u}_{j_k}^{k+1}$, then $\hat{j}_k^{k+1} = \hat{j}_k^k + \hat{j}_k^k$

(c) If sign $\hat{u}_{i_k}^{k+1} \neq \text{sign} \ \hat{u}_{j_k}^{k+1}$, then $\hat{j}_k^{k+1} = \hat{j}_k^k - \hat{j}_k^k$

A corresponding reduced vector $\hat{\phi}^{k+1}$ is formed with the $i_k$-th component removed. This step constrains the velocity of the $i_k$-th and $j_k$-th joints to be equal in magnitude in the optimal solution. (Recall, from the discussion in section 3.4, that at least $(n - m + 1)$ components of $\hat{\phi}^*$ will have equal absolute values when $J$ has rank $m$).

9. Assume $i_{k+1} = j_k$ and go to step 3. Repeat steps 3 to 9 until $k = n - m - 1$.

Thus, after the last (i.e. $(n-m)$-th) iteration, the minimum infinity-norm solution $\hat{\phi}^*$, with at least $n - m + 1$ equal magnitude components is reconstructed as follows

$$\hat{\phi}_t^* = \begin{cases} \hat{\phi}_t^{n-m}, & i \neq i_k \\ \hat{\phi}_{i_k-m-1}^{n-m}, & i = i_k \end{cases}, \quad k = 0, 1, \ldots, n - m - 1$$

Note that for a single degree of redundancy, the above algorithm converges in one iteration.

The essence of the algorithm lies in step 5 which computes a scalar $\alpha$ that gives best possible reduction of the maximum component of the current estimate $\hat{\phi}^k$. The chosen $\alpha$ must be such that the absolute value of the current maximum ($i_k$-th) component is reduced and that the absolute values of all other components remain less than or equal to that value. The eqs. in lines (a), (b), (c) of step 5 arise from the following analysis.
From eq.(3.25), the projection vector $p^k$ can be written as

$$p^k = -s_k [I - J^+ J]_{i_k}$$

where $[I - J^+ J]_{i_k}$ is the $i_k$-th column of $(I - J^+ J)$. Since $(I - J^+ J)$ is a projection matrix, its diagonal elements are always positive. This implies that the sign of the $i_k$-th element of $p^k$ is always opposite to that of $s_k$ which is the sign of the maximum component of $\hat{\theta}^k$. Hence,

$$|\hat{\theta}_{i_k}^k + \alpha p_{i_k}^k| = |\hat{\theta}_{i_k}^k| s_k + \alpha |p_{i_k}^k| (-s_k)|$$

$$= |s_k (|\hat{\theta}_{i_k}^k| - \alpha |p_{i_k}^k|)|$$

$$= \left|\hat{\theta}_{i_k}^k - \alpha |p_{i_k}^k|\right|$$

Thus, as long as $\alpha > 0$, the absolute value of the $i_k$-th component of the updated vector $\hat{\theta}^{k+1}$ will always be less than that of $\hat{\theta}_{i_k}^k$. Now, in order to ensure that the values of all other components are also less than that of $\hat{\theta}_{i_k}^{k+1}$, we need for each $j = 1, \ldots, n$ (except $j = i_k$),

$$|\hat{\theta}_j^k + \alpha p_j^k| \leq |\hat{\theta}_j^k + \alpha p_{i_k}^k| = \left|\hat{\theta}_{i_k}^k - \alpha |p_{i_k}^k|\right| \quad \text{for } j = 1, \ldots, n; \quad j \neq i_k \quad (3.26)$$

The computations (a), (b), (c) in step 5 correspond to computing $\alpha$ with equality in eq.(3.26) for the various components (taking into account the different signs of the components). By choosing the smallest positive $\alpha$, all the components will have absolute value less than or equal to that of $\hat{\theta}_{i_k}^{k+1}$.

Although this algorithm is computationally more expensive than the ones using the duality principle (since it involves the computation of the null space projection matrix of $J$), it has more flexibility in that it can be used to compute a suboptimal solution with additional subtask performance, as described in section 3.9. This is useful in biasing the joint velocity vector to satisfy an additional subtask criterion, while simultaneously reducing the $\infty$-norm and hence the individual joint velocities required to perform the tasks. Also, since the pseudoinverse solution can be used as the initial choice, this algorithm will be useful in situations where the pseudoinverse solution can be computed cheaply (for instance, when there exists special-purpose hardware to do so [97]) or, if there is a need for computing both types of solutions.

---

*Notice though, that only $(n - m)$ columns of $(I - J^+ J)$ (corresponding to the values of $i_k$) need to be computed*
3.8 Weighting the $\infty$-norm

In the above analysis, while computing the minimum $\infty$-norm solution, we have implicitly assumed that all the joints are weighted equally i.e. we would like to obtain equally low velocities for all the joints. However, there will be instances when the motors on the different joints have different characteristics and hence, the maximum velocity threshold for each joint may not be the same. In that case, before computing the optimal solution, it will be necessary to the weight the joint velocities appropriately.

Suppose the physical specifications of a manipulator require that the actuator for the $i$-th joint have velocity less than $\Delta_i$ for $i = 1, 2, \ldots, n$. Let

$$D = \begin{bmatrix} \frac{1}{\Delta_1} & \cdots & \frac{1}{\Delta_n} \end{bmatrix}$$  \hspace{1cm} (3.27)

Then, in order to reduce the magnitudes of the joint velocities in proportion to the limits on their actuators, the joint velocity vector is scaled, using the matrix $D$. Hence, the weighted optimization problem becomes

$$\min \|D\hat{\theta}\|_\infty$$  \hspace{1cm} (3.28)

subject to  \hspace{0.5cm} \hat{\theta} = J\hat{\theta}$$

Using the transformations

$$\tilde{\theta} = D\hat{\theta} \quad \text{and} \quad \tilde{J} = JD^{-1}$$  \hspace{1cm} (3.29)

the problem reduces to

$$\min \|\tilde{\theta}\|_\infty$$  \hspace{1cm} (3.30)

subject to  \hspace{0.5cm} \tilde{\theta} = \tilde{J}\tilde{\theta}$$

This is the minimum infinity-norm optimization problem with $\hat{\theta}$ and $J$ replaced by $\tilde{\theta}$ and $\tilde{J}$ respectively. By using one of the algorithms described in the previous sections, the solution $\tilde{\theta}^*$ to problem (3.30) can be obtained. Then the joint velocity vector $\hat{\theta}^*$ which solves (3.28) is computed as

$$\hat{\theta}^* = D^{-1}\tilde{\theta}^*$$

Note that $\|\tilde{\theta}^*\|_\infty < 1$ will mean that $\tilde{\theta}^*_i < \Delta_i$ for all $i = 1, \ldots, n$. For examples using this technique and comparing it with the weighted Euclidean norm solution (eq. 3.3), see section 3.11.2.
3.9 Subtask Performance With Infinity-Norm Reduction

In various applications, redundant manipulators can be employed to perform a secondary task in addition to the main task of following the specified end-effector trajectory. This subtask will usually have a lower priority than that of the main task and the accuracy with which it can be performed will depend on the degree of redundancy of the manipulator. Task priority strategies have been utilized in [59] and [76], for example, to exploit the extra degrees of freedom of redundant manipulators for performing subtasks such as obstacle avoidance and singularity avoidance. These subtasks are incorporated in the inverse kinematic solution by projecting them onto the null space corresponding to the first task, so that they do not interfere with the main task. This is done as follows.

Suppose \( \dot{x}_1 \) is the main task and \( \dot{x}_2 \) is the secondary subtask to be performed by the manipulator. Let \( J_1 \) and \( J_2 \) be the Jacobian matrices corresponding to the two tasks. Then the joint velocity vector can be computed as ([59], [76])

\[
\dot{\theta} = J_1^+ \dot{x}_1 + \beta_1 \left[ J_2 (I - J_1^+ J_1) \right]^+ (\beta_2 \dot{x}_2 - J_2 J_1^+ \dot{x}_1) \tag{3.31}
\]

The scalars \( \beta_1 \) and \( \beta_2 \) are introduced in order to incorporate the priorities of the tasks and to minimize the rank change effects (due to the matrix \( J_2 (I - J^+ J) \)) [59].

If a manipulator has a degree of redundancy \( r = (n - m) > 1 \), then it is possible to combine subtask performance with reduction of infinity norm of the inverse kinematic solution, using the algorithm described in the previous section. The only changes in the algorithm would be the computation of the initial joint velocity vector \( \dot{\theta}^0 \) and the number of iterations that the algorithm is run for.

Instead of using eq. (3.24), we use (3.31) to compute the initial joint velocity vector \( \dot{\theta}^0 \) in step 1. The steepest descent algorithm in section 3.7 can then be run for a number of iterations strictly less than the degree of redundancy \( r \) of the manipulator. This will result in a solution that partly performs the subtask and also has infinity norm lower than that of the solution given by (3.31). The degree to which the subtask is performed is determined by the relative values of the scalars \( \beta_i \) in (3.31) and those of the scalars \( \alpha_j \) computed in the infinity-norm algorithm. The number of iterations that the infinity-norm reduction algorithm is run for, is determined by the priorities associated with the respective tasks and the value of \( r \). Note that, if the algorithm is run for a number of iterations equal to \( r \), it will converge to the optimal \( \infty \)-norm solution, which will make it equivalent to starting from the pseudoinverse solution. This strategy of computing \( \dot{\theta} \) will be useful in applications where the magnitude of
each joint velocity is to be kept small while the manipulator is commanded to perform a main task accurately and a subtask of lower priority. In section 3.11.3, this method has been used to perform obstacle avoidance while maintaining low joint velocities.

3.10 Subtask Performance with $\infty$-Norm Criterion

Until now, we have considered minimizing (or reducing, as in section 3.9), the $\infty$-norm of $\dot{\boldsymbol{\theta}}$ to obtain inverse kinematic solutions that will keep the joint velocities as low as possible. This has been our implicit subtask, in addition to tracking the desired end-effector trajectory. In this section, we look at using the $\infty$-norm to define functions in the joint space which are useful in performing other tasks such as avoiding obstacles in its workspace, increasing the manipulability of configurations, tracking a desired “elbow” trajectory, avoiding joint limits, etc. These subtasks can be specified as functions of the joint positions. Once again, the norm commonly used to formulate the criteria for performing these tasks is the two-norm. For example, one way to define a performance criterion for such subtasks is [108]

$$h(\boldsymbol{\theta}) = k(\boldsymbol{\theta} - \boldsymbol{\theta}^r)^T [H] (\boldsymbol{\theta} - \boldsymbol{\theta}^r)$$

Here, $k$ is a scalar constant and $H$ is a weighting matrix. $\boldsymbol{\theta}^r$ is a reference joint configuration which can be specified as a desirable or an undesirable configuration. Accordingly, $h(\boldsymbol{\theta})$ is either minimized or maximized since it represents the weighted Euclidean distance of the joint configuration $\boldsymbol{\theta}$ from the configuration $\boldsymbol{\theta}^r$. For example, a joint limit avoidance criterion can be specified as the Euclidean norm of the difference between the joint position vector and the vector of the center positions of the respective joints [54] i.e. $\theta_{ri}$ is the center of travel of the $i$-th joint.

However, what we really want is for the absolute distance of each joint position to be as small (or as large) as possible from the corresponding component of $\theta^r$. Thus, once again, the physical constraint will be better represented by the $\infty$-norm of the subtask criterion as compared to the Euclidean norm. In [47], the optimization of joint limit avoidance criteria defined using the family of $p$-norms (for $p \geq 1$) has been discussed. For these norms the subtask performance functions in [47] are defined as

$$H_p(\boldsymbol{\theta}) = \left( \sum_{i=1}^{n} \left[ \frac{\theta_i - \theta_{ri}}{\Delta \theta_i} \right]^p \right)^{1/p}$$
where the division by $\Delta \theta_i$ is used to normalize the distance by the maximum range of travel. The joint velocities are then computed as

$$\dot{\theta} = J^+ \dot{x} + \alpha (J^+ J - I_n) \nabla H_p(\theta)$$  \hspace{1cm} (3.32)

It was noted in [47] that behavior of the system depends on the size of the weighting constant $\alpha$ and for large $p$, if the gain $\alpha$ is too high, the system may become unstable.

In this section, a new method of choosing the optimal $\alpha$ for the case $p = \infty$ is introduced. This method is obtained by reducing the problem of subtask optimization to the minimum $\infty$-norm optimization problem as defined in (3.11).

Suppose $\theta^r$ is the joint configuration at the current time step and $\dot{\theta}$ is the joint velocity vector (to be determined). Define the joint configuration vector $\theta^n$ as

$$\theta^n = \theta^r + \dot{\theta}$$ \hspace{1cm} (3.33)

The subtask is considered to be that of making each new joint position $\theta^n_i$ as close as possible to a given position $\theta^r_i$. Let $\Delta \theta_i$ be the maximum range of travel for the $i$-th joint. Then, the components of the subtask criterion vector $z$ (whose norm should be minimized) can be defined as

$$z_i = \frac{\theta^n_i - \theta^r_i}{\Delta \theta_i}$$

or, in matrix-vector form,

$$z = D(\theta^n - \theta^r)$$ \hspace{1cm} (3.34)

where $D$ is an $n \times n$ diagonal matrix with its $i$-th diagonal element equal to the reciprocal of $\Delta \theta_i$. Substituting from (3.33) in (3.34),

$$\Rightarrow z = D \left( \dot{\theta} + \theta^r - \theta^r \right) = D \left( \dot{\theta} + \varepsilon \right) ; \varepsilon = \theta^r - \theta^r$$ \hspace{1cm} (3.35)

Hence, the optimization problem becomes

$$\min \|z\|_{\infty}$$ \hspace{1cm} (3.36)

subject to

$$\dot{\theta} = J \dot{\theta}$$

$$z = D(\dot{\theta} + \varepsilon)$$

This problem can be easily reduced to the standard form (3.11) as follows. From (3.35),

$$\dot{\theta} = D^{-1} z - \varepsilon$$
Thus,
\[
\dot{x} = J\dot{\theta} \Rightarrow \dot{x} = J \left( D^{-1}z - \xi \right) \Rightarrow \dot{x} + J\xi = (JD^{-1})z
\]

Let
\[
b = \dot{x} + J\xi; \quad \tilde{J} = JD^{-1}
\]

Hence, the problem (3.36) is equivalent to

\[
\min \|z\|_{\infty} \quad \text{subject to} \quad b = \tilde{J}z \quad (3.37)
\]

Problem (3.37) is equivalent to problem (3.11) and once again, any of the algorithms described in sections 3.5, 3.6 or 3.7 can be used to compute the solution \( z^* \) to the problem (3.37). The corresponding joint velocity vector is then computed as

\[
\dot{\theta}^* = D^{-1}z^* - \xi
\]

Note that, by solving (3.37), the scale factor \( \alpha \) in eq.(3.32) is automatically determined and does not have to be decided empirically. An example of joint limit avoidance using this method is shown in section 3.11.4.

3.11 Examples

Here, we demonstrate the use of the algorithms described in the previous sections, for computing inverse kinematic solutions for the planar four-link manipulator with unit link lengths and revolute joints and for the PUMA geometry which has been made redundant by adding one or more translational degrees of freedom to its base.

3.11.1 Simulation 1

The first simulation compares the two-norms and infinity-norms of the inverse kinematic solutions obtained by using the pseudoinverse solution and by using the minimum infinity-norm algorithms (any of the \( \infty \)-norm algorithms in sections 3.5, 3.6 or 3.7 can be used). Since comparison of the norms of the solutions is valid only if it is done for the same joint configuration, the following procedure for comparing the norms has been adopted. For a given end-effector trajectory, at each time step, the joint velocity vector, \( \dot{\theta}^* \), is first computed, using the pseudoinverse. Its two-norm and infinity-norm are determined. The corresponding joint configurations for each time
step are stored. Then, for each of these joint configurations, the minimum infinity-norm joint velocity vector, $\hat{\theta}^*$, is determined using the described algorithms along with its respective norms.

**Planar Manipulator**

The planar manipulator used in these simulations has four links, with unit link lengths and all revolute joints. For this example, the joint angles for the initial configuration are $-60^\circ$, $40^\circ$, $25^\circ$ and $85^\circ$ respectively. The end-effector is commanded to move vertically upward at 3.0 units/sec. The trajectory followed by the robot arm is shown in fig. 3.2(a). The four norms obtained via the procedure described above are plotted in fig. 3.2(b). Plots 1, 2, 3 and 4 correspond to $\|\hat{\theta}^*\|_\infty$, $\|\hat{\theta}^*\|_\infty$, $\|\hat{\theta}^*\|_2$ and $\|\hat{\theta}^*\|_2$ respectively.

It is seen that while the average difference between the 2-norms of the two solutions is less than 12%, the average difference between the $\infty$-norms is 34%. Thus, with an increase of less than 12% in the Euclidean norm, the $\infty$-norm solution reduces the maximum joint velocity by 34% compared to that of the pseudoinverse solution. The simulation was run for 500 time steps with $\Delta t = 0.001$. When the column exchange algorithm in section 3.5 was used, the algorithm took two iterations to converge in the first and fifth time step and one iteration at all other configurations.

**The PUMA manipulator**

The PUMA manipulator has been made redundant by adding a translational degree of freedom in the z-direction (along the axis of its first revolute joint) to its base. The initial positions of the revolute joints were $28.75^\circ$, $18.90^\circ$, $75.90^\circ$, $11.10^\circ$, $14.25^\circ$ and $10.15^\circ$ respectively and the end-effector velocity specified was $[0.025, 0.0, 0.0, -0.1, 0.0, 0.0]^T$.

In this case while the mean percentage difference between the 2-norms of the two solutions was about 9%, that between the $\infty$-norms was about 23%. A general observation from various simulations (on both manipulators) is that typically, the average percentage increase in Euclidean norm of the $\infty$-norm solution (from the pseudoinverse solution) is less than the average percentage reduction obtained in the value of the $\infty$-norm between the two solutions. The simulation was run for 750 time steps. Except for 4 instances, the column exchange algorithm took one iteration to converge for all the time steps. For the first time step, it took 3 iterations while at
Figure 3.2 Simulation 1: Comparison of Joint Velocity Norms - Planar Arm
3 other configurations (corresponding to the corners in the plot), it took 2 iterations to converge.

3.11.2 Simulation 2

This simulation shows how the scaling of the joint velocity vector can be more effective when the infinity-norm is minimized compared to the minimization of the weighted Euclidean norm. Once again, we use a common evolution of the joint configuration (fig. 3.4 (a)) to compare the values of the joint velocity at each configuration.

**Planar Arm**

Fig. 3.4 (b) shows the velocity of joint 1, when the inverse kinematic solution is computed with three different methods, for the 4-link planar arm. The dotted line 1 represents the pseudoinverse solution. The line 2 represents the value obtained when
Figure 3.4 Simulation 2: Weighting of the first joint
a weighted pseudoinverse solution (eq. (3.3)) is used with

\[ D = \text{diag}\{2.0, 1.0, 1.0, 1.0\} \]

Note that the choice of the matrix \( D \) has been made in order to reduce the velocity of joint 1 approximately by half its value corresponding to the pseudoinverse solution. However, the average difference between the pseudoinverse and weighted pseudoinverse solutions (lines 1 and 2 in fig. 3.4(b)) is 18%. When the technique described in section 3.8 is used to compute a weighted \( \infty \)-norm solution with the same matrix \( D \), the velocity of the first joint is on an average 48% lower than the corresponding pseudoinverse solution (line 3 in fig. 3.4 (b)). This is close to the desired 50% reduction that the choice of the matrix \( D \) indicates.

**The PUMA Manipulator**

To depict the effects of weighting the two norms for a spatial arm, we have used the PUMA manipulator, this time with translational degrees of freedom along the \( x \) and \( y \) axes of the base. Thus, the arm has 8 degrees of freedom in the joint space, with the first two being the prismatic joints, the next three being the base, shoulder and elbow joints and the last three the wrist joints. The initial configuration for the revolute joints is \([60^\circ, -30^\circ, 45^\circ, 45^\circ, -20^\circ, 30^\circ]\). The commanded end-effector velocity is \([0.0, 0.0, 0.05, 0.05, 0.0, 0.0]\). We require the first two wrist joints (i.e. the joints corresponding to the pitch and yaw motions) of the arm to have low joint velocities. Hence, the weighting matrix \( D \) for this case is chosen to be

\[ D = \text{diag}\{1.0, 1.0, 1.0, 1.0, 15.0, 15.0, 1.0\} \]

The joint velocities resulting from using the three solutions - the pseudoinverse (line 1), the weighted pseudoinverse (line 2) and the weighted infinity-norm (line 3) - are shown in figs. 3.5 and 3.6. It is seen that the weighted infinity-norm has the lowest joint velocities for the pitch and yaw joints, with average differences of 47 % and 72 % from the corresponding pseudoinverse solutions. The weighted pseudoinverse solution actually has greater joint velocity for the pitch joint, while the yaw joint velocity is on an average 40 % lower than the yaw joint velocity of the pseudoinverse. Thus, by minimizing the (weighted) maximum joint velocity, the weighted infinity-norm, in this case, has performed the desired task more effectively than the weighted pseudoinverse solution. Note that in order to reduce the load on the wrist joints, the
Figure 3.5  Simulation 2: Weighting of the PUMA wrist joints
Figure 3.6 Simulation 2: Weighting of the PUMA wrist joints
weighted infinity-norm has to load the prismatic joints and the shoulder and elbow joints more heavily than the other solutions.

3.11.3 Simulation 3

![Pseudoinverse solution](image)

**Figure 3.7** Simulation 3: Pseudoinverse solution trajectory

This simulation demonstrates the combination of secondary task performance with infinity-norm reduction. The planar four-link arm is used for ease of presentation. To motivate this simulation, consider fig.(3.7) which shows the evolution of the robot's configuration when the joint velocity vector is computed using the pseudoinverse of the Jacobian. The initial configuration is $[60^\circ, -120^\circ, 120^\circ, -120^\circ]$ and the desired end-effector velocity is 2.0 units/sec in the positive x-direction. It is seen that the 'elbow' or the tip of the second link moves in the negative y-direction when the pseudoinverse solution is used (fig. 3.7). Now, suppose there is an obstacle in that direction and the elbow has to be moved in the upward direction for the same end-effector trajectory. Hence, the secondary task is specified in the form of a desired
Figure 3.8 Simulation 3: Trajectories with and without norm reduction
velocity of 0.5 units/sec in the positive y-direction, for the elbow (i.e. \( \dot{x}_2 \) in eq.(3.31) is equal to \([0.0, \ 0.5]\)).

Fig. 3.8(a) shows the evolution of the robot arm when eq.(3.31) is used to compute the joint velocities. In this case, although the elbow trajectory is followed exactly, the joint velocities become very high as evidenced by the plot of the infinity-norm of the joint velocity vector in fig. 3.9(a). However, when one iteration of the steepest descent algorithm is applied to the solution obtained from eq.(3.31), as described in section 3.9, the joint velocities do not shoot up as seen in fig. 3.9(a). Of course, now the elbow trajectory is not tracked exactly, but it is tracked accurately enough that the elbow still moves upward, as desired (fig. 3.8(b) and fig. 3.9(b)). Although, this same behavior could have been obtained without using the steepest descent algorithm (say, by empirically tuning the gains in eq.(3.31)), the use of the algorithm gives an automatic regulation of the gain, by reducing the maximum component of the joint velocity at each configuration. Another example of this technique can be found in [29].

3.11.4 Simulation 4

Here, the optimization of an \( \infty \)-norm subtask performance criterion by solving the problem (3.37) is depicted. Specifically, we look at the case of joint limit avoidance, although any subtask which can be defined in the form of eq.(3.34) can be attained. In the example shown, the center of travel of each joint is assumed to be \( 0^\circ \) so that

\[
\theta^* = [0^\circ \ 0^\circ \ 0^\circ \ 0^\circ]^T
\]

The subtask performance vector \( z \) is defined as in eq.(3.34) with \( \Delta \theta = 120^\circ \) defining the maximum range of travel for each joint.

Fig. 3.10 shows the evolution of the robot's configuration when the pseudoinverse solution is used and when the algorithm described in section 3.10 is used. The pseudoinverse solution causes the arm to get into an awkward configuration, with the third and the fourth joints very far away from their centers of travel. However, by using the algorithm, the manipulator is able to track the desired end-effector trajectory without any of the joints deviating a great deal from their centers of travel. The difference between the maximum deviations in the two evolutions is shown in fig. 3.11.
Figure 3.9  Simulation 3: Joint Velocity Norms and Elbow Trajectories
Figure 3.10 Simulation 4: Trajectories of Robot Arm
3.12 Discussion and Conclusions

Owing to its analytical tractability, the Euclidean norm has been extensively used in velocity inverse kinematics to compute minimum-norm solutions and to optimize subtask performance measures. In this chapter, computation of inverse kinematics of redundant manipulators using the infinity norm as an optimality measure is investigated. The main motivation for this work is the fact that the infinity norm proves to be a better measure of performance than the Euclidean norm when a solution with low individual components is desired. While optimization of the Euclidean norm yields "minimum-energy" kinds of solutions, minimization of the ∞-norm corresponds to the computation of a "minimum-effort" solution.

Several algorithms for computing the minimum infinity-norm inverse kinematic solution for redundant arms are presented. These algorithms demonstrate that the computation of the ∞-norm solution compares favorably to that of the pseudoinverse solution and can also be cheaper than the latter [33] for arms with a single degree of redundancy. Optimization of subtask criteria defined using the infinity norm, by
means of these algorithms has been addressed. Using this optimization technique, the need for determining an empirical scale factor for the null space term is eliminated. In addition, for multiple degrees of freedom, a technique for reducing joint velocities while performing a subtask of lower priority is introduced. The examples in section 3.11 illustrate the fact that optimization of the \(\infty\)-norm can be used to produce solutions that satisfy the physical constraints on the manipulator (such as joint velocity limits, joint position limits, distance from obstacle) better than those obtained by optimizing the Euclidean norm.

A point to be noted is that since only local optimization methods are considered, the only valid comparison for the norms is one done at the same joint configuration. However, the joint trajectories resulting from optimizing different norms are different and hence will result in different evolutions of the robot.

A need for determining some kind of measure to predict the behavior obtained by minimizing different norms is illustrated by the following example. The example also shows that there exist trajectories where minimizing the \(\infty\)-norm gives feasible solutions even when the pseudoinverse solution becomes infeasible.

Consider once again the planar 4-link arm with revolute joints. The initial configuration is taken to be \([20^\circ, -20^\circ, -20^\circ, 110^\circ]\) with the end-effector trajectory specified as \(-2.0\) units/sec in the x-direction. The respective evolutions of the robot arm with the pseudoinverse and minimum infinity-norm solutions are shown in fig. 3.12 while comparisons of their Euclidean and infinity norms are shown in fig. 3.13. (It should be noted that the norms for the pseudoinverse and the minimum \(\infty\)-norm solutions correspond to different joint trajectories in these plots). Note the high peak in the norms corresponding to the pseudoinverse which arises because the pseudoinverse solution takes the arm close to a singularity. On the other hand, the minimum infinity-norm solution is able to avoid the singular configuration. Thus, there exist instances when choosing one norm over the other will represent the difference between a (physically) feasible or infeasible trajectory. Future work involves the determination of measures which will indicate when the optimization of a particular norm will give considerably better performance than another.

Two important issues that remain to be discussed are the uniqueness and the continuity of the solutions obtained by minimizing the infinity-norm. It is well-known that these are properties possessed by the minimum two-norm solution (except at a singular configuration, where the pseudoinverse solution becomes discontinuous).
Figure 3.12  Evolutions of Robot Arm
Figure 3.13  Comparison of norms
Since, the infinity norm is not a strictly convex function, in theory, the inverse kinematic solution is not unique. However, in our simulations run with the three algorithms, we have obtained identical solutions. It would be interesting to investigate the existence and significance of multiple optimal solutions in certain or all regions of the manipulator joint space.

The other important property is the continuity of joint velocities from one iteration to the next, when the end-effector is made to follow a continuous trajectory. This property is most easily discussed using the fact that the solution to the dual problem $E^*$ is orthogonal to at least $m - 1$ columns of the Jacobian matrix. It is clear that the inverse kinematic solution will be continuous when the set of columns of $J$ that it corresponds to remains unchanged. According to our simulations, a switch in the columns occurs very rarely when the end-effector is following a continuous trajectory. However, a more interesting observation is that even when the columns are switched, there is no discontinuous jump in the joint velocities i.e. the joint velocities corresponding to the two sets of columns intersect at the point where the solution switches from one set to the other. One avenue of future work will be to prove or disprove the fact that this is always the case.

In conclusion, the minimum infinity-norm inverse kinematic solution is an attractive alternative to the pseudoinverse solution when individual joint velocity magnitudes are of concern. The algorithms described herein enable efficient computation of these solutions and hence can be used to obtain joint velocities with lower individual magnitudes, even when an additional subtask is being performed. The algorithms can also be used to optimize subtask performance criteria defined using the infinity norm, which give a better physical measure of the performance than those defined using the Euclidean norm.
Chapter 4

Multiple Manipulator Systems

...it would certainly be much simpler to consider building a single, all-powerful Super-Fungus-Eater instead of plural, less powerful Fungus-Eaters. Even such a super-robot, however, may occasionally commit a fatal mistake, or the environmental conditions may undergo an unexpected twist that undermines the ability of the super-robot, and then all is lost. On the other hand, by having a number of less powerful Fungus-Eaters with considerable individual differences, by design or as an outcome of different experiences, the whole colony of Fungus-Eaters will be made much less vulnerable both to individual errors and to changes in the environmental conditions, by virtue of variation in opinions, propensities and abilities. At the same time, they can be collectively very powerful when the need for cooperation arises.

- from 'Fungus-Eater Robots' by Masanao Toda

4.1 Introduction

![Diagram of a robotic manipulator](image)

Open-chain, Serial-link Arm

Movable Base

Constrained End-Effector

Figure 4.1 Variations of a Robotic Manipulator

In the previous chapters, we have investigated the kinematic control of a single open-chain serial robotic manipulator i.e. a mechanism consisting of links connected in
series via joints, with one end (the base) fixed and the other end (the end-effector) free to move in space. This structure, in some sense, the fundamental building block for more sophisticated robotic systems. Some immediate variations of this mechanism are the cases when, for instance, the base itself is capable of motion (e.g. mobile, robots, space robots) or when the motion of the end-effector is constrained in certain directions (e.g. applications where the end-effector is required to be in contact with a surface). Notice that the former case can be reduced to one in which the base is considered fixed, by adding fictitious joints corresponding to the degrees of freedom of the base. On the other hand, the latter case warrants the incorporation of force control at the end-effector, in addition to position control.

![Cooperating Multiple Arm System](image1)

*Figure 4.2 Cooperating Manipulator Systems*

Various combinations of the above kinds of mechanisms lead to more complex and often times, more anthropomorphic robotic systems such as Multifingered Hands, Legged Locomotion Vehicles and Multi-arm systems. Such systems better resemble the cooperative multi-limb manipulation carried out by humans in their everyday lives. Cooperative manipulation may be necessitated for a variety of reasons, as for example, when handling objects too large or too heavy for a single arm or by the need for fine manipulation in certain tasks or by the need for locomotion. The main difference between these systems and the single manipulator is the presence of one or more closed kinematic chains, formed due to the coordinated handling of a common load or object with several manipulators.
A multi-body system is said to contain a closed kinematic chain, when there exist two different paths connecting some or all pair(s) of bodies in the system and the coupling between each pair imposes kinematic constraints. The dynamics of the resulting system is subject to various constraints imposed by the interaction between the manipulators and the object, causing the number of degrees of freedom of the system to be less than the sum of the individual degrees of freedom of the cooperating mechanisms. The study of these kinds of systems is commonly grouped under the heading of Coordinated Multiple Manipulator systems. The focus of the work in this chapter is on systems comprising multiple arms or fingers manipulating a common object.

The recognition of the wide range of potential applications of multiple manipulator systems has spawned a large amount of interest in their study and development over the past decade. The primary research issues involved in the analysis of these kinds of systems may be broadly classified as:

- Grasp Analysis and Grasp Planning
- Load Distribution among manipulators
- Kinematic and Dynamic Modeling, Motion and Control Strategies
- Contact and Impact Analyses

The analysis and planning of grasp strategies involves the determination of suitable contact points on the object, at which the end-effectors of the manipulators should grip the object, so that a stable grasp is obtained ([17], [38], [67], [80]). Once, the grasp points are determined, it is necessary to compute the forces (and moments) that each individual manipulator should exert on the object to perform the desired task. This leads to the problem of load distribution ([77], [83], [87], [98]). The determination of appropriate gripping forces is critical, not only to hold the object securely and manipulate it as desired, but also to ensure that the object is not subjected to excessive internal stresses (especially if it is fragile). To realize the desired object trajectory and the desired gripping forces, the joints of the manipulators must be controlled suitably. This is done using kinematic and dynamic models of the cooperating manipulator system and by devising control methods to achieve the desired forces, positions and velocities ([42], [45], [58], [72], [74]). It is also important to model the effects of contact/impact forces on the system, in the instances when two
bodies come in contact with each other ([66], [99], [111]). This can happen when, for example, the end-effectors of manipulators initially grip an object or the legs of a locomotion vehicle come in contact with the terrain.

An overview of the research issues involved in controlling multiple manipulator systems can be found in [49]. For an extensive bibliography see for example [100], [107].

4.2 Grasping and Fine Manipulation

The primary purpose of a robot manipulator (as its name suggests) is to manipulate the objects in its world. The interaction of the manipulator with its environment takes place through its end-effector (the equivalent of the human hand). Early industrial robots were employed for simple repetitive tasks that did not require a lot of precision or flexibility. Consequently, such robots have parallel-plate grippers, resembling tongs or pliers, as end-effectors. Robotic arms that are dedicated to a specific task such as spray-painting, welding, forging, etc. are endowed with special-purpose devices for end-effectors.

As the range and complexity of the tasks that need to be handled by robotic systems grows, the issues of grasping and fine manipulation have assumed a new importance. The growing need for more versatile end-effectors stems from the fact that a robotic manipulator cannot accomplish both gross motions (such as pick-and-place operations) and fine-motion manipulations (like insertions, rotations, etc). Also, the earlier generation of grippers and special-purpose effectors lack the flexibility and dexterity which is required in performing fine-motion manipulations. The main source of inspiration in developing more universal end-effectors is the human hand, which is an extremely general-purpose end-effector. The evident versatility of the human hand has led to research in developing artificial hands which can adapt to different grasps and possess the capacity to perform fine motions. An excellent survey of the technologies which support general-purpose manipulation and a description of several representative devices can be found in [39].

The development of different kinds of end-effectors has resulted in defining different types of grasps which can be used by one or more end-effectors to manipulate an object. To facilitate grasp analysis and grasp selection, numerous attempts have been made to categorize grasps according to a variety of factors such as size and shape of object, the nature of the task, grasp stability, grasp dexterity, etc. For a detailed
analysis of grasping taxonomies, see [67]. One factor which will be of importance to the analysis in this chapter is the nature of contact between the surfaces of the end-effector and the object.

4.2.1 Types of Contact

In addition to the physical and mechanical properties of the hand and the object, it is also important to study the nature of contact between the end-effector and the object. The geometric and frictional properties of the contact determine the forces and moments that are transmitted to the object and hence, also determine the relative motion between the two bodies.

When two rigid bodies come in contact with each other, their surfaces can have three types of contact - point contact, line contact and plane contact. In [64], the constraints imposed by each type of contact with and without friction have been described in terms of the degrees of freedom of relative motion allowed and in terms of screw systems of motion and force associated with each type. Table 4.2.1 outlines the number of degrees of freedom associated with these contact types (in decreasing order) along with the corresponding relative translational and angular velocities allowed and the transmitted forces and moments. (The common tangent plane to the contacting bodies is taken to be the x-y plane. For line contacts, the line is assumed to be along the y-axis).

Notice that the velocities and forces/moments are dual to each other i.e. the directions in or about which relative velocities are permitted correspond to the directions in/about which forces/moments are not transmitted.

Similarly, Cutkosky [21] has classified contact types based on the geometry of the fingertips. A set of models has been defined that may be used to characterize fingertip geometries - pointed, hard curved, flat, elastic curved and very soft(fig. 4.3).

![Fingertip Geometries](image)

**Figure 4.3** Fingertip Geometries
Table 4.1 Basic Types of Contact

<table>
<thead>
<tr>
<th>Contact Type</th>
<th>D.O.F.</th>
<th>u</th>
<th>ω</th>
<th>f</th>
<th>m</th>
</tr>
</thead>
<tbody>
<tr>
<td>No contact</td>
<td>6</td>
<td>x, y, z</td>
<td>x, y, z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Frictionless point contact</td>
<td>5</td>
<td>x, y</td>
<td>x, y, z</td>
<td>z</td>
<td>-</td>
</tr>
<tr>
<td>Frictionless line contact</td>
<td>4</td>
<td>x, y</td>
<td>y, z</td>
<td>z</td>
<td>x</td>
</tr>
<tr>
<td>Friction point contact</td>
<td>3</td>
<td>-</td>
<td>x, y, z</td>
<td>x, y</td>
<td>-</td>
</tr>
<tr>
<td>Frictionless plane contact</td>
<td>3</td>
<td>x, y</td>
<td>z</td>
<td>z</td>
<td>x, y</td>
</tr>
<tr>
<td>Soft finger</td>
<td>2</td>
<td>-</td>
<td>x, y</td>
<td>x, y, z</td>
<td>z</td>
</tr>
<tr>
<td>Friction line contact</td>
<td>1</td>
<td>-</td>
<td>y</td>
<td>x, y, z</td>
<td>x, z</td>
</tr>
<tr>
<td>Friction plane contact</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>x, y, z</td>
<td>x, y, z</td>
</tr>
</tbody>
</table>

Kinematic constraints and friction conditions for each type are studied. The various fingertip geometries in this analysis correspond to one of Salisbury’s contact types (see Table 4.2.1). For example, the pointed finger and the curved finger correspond to the friction point contact while the flat finger makes the friction plane contact. The difference between the pointed and the curved finger is that the contact point of the former remains fixed on the object while the latter makes a rolling motion on the object surface. Elastic fingertip compliance is incorporated in the very soft model and the soft curved fingertip has elastic coupling and rolling motion.

In subsequent analysis in this chapter, we will mostly concentrate on point contact with friction and the soft finger contact in Salisbury’s contact types and the hard curved finger in Cutkosky’s classification, as these contacts correspond to relative rolling motion between the contacting bodies. In order to ensure that there is no relative slipping between the contacting bodies, the contact forces must lie within the friction cone at each point of contact. The friction cone is a geometric interpretation of Coulomb’s Law, which requires that the magnitude of the tangential component of the contact force be less than or equal to the product of the coefficient of friction μ and the magnitude of the normal component of the contact force [64].
4.2.2 Rigid/Non-rigid grasps

As mentioned earlier, before the advent of artificial multifingered hands, the most common end-effector for a robotic manipulator was a parallel-jawed gripper which could clamp onto an object. Thus, the contacts between the gripper and the object were of the planar type with friction. According to Table 4.2.1, this kind of contact permits no relative motion between the contacting surfaces and allows the transmission of all forces and moments.

Hence, until a few years ago, the analyses of grasping and load distribution among cooperating manipulators assumed the presence of **Rigid Grasps** i.e. those that correspond to no relative motion between the contacting bodies and allow all forces and moments to be exerted across the contact. In particular, this assumption means that the contact points remain fixed on the end-effector and object surfaces. Hence, once the object has been held by the end-effectors, the grasp remains unchanged for the entire task.

However, many tasks require a more flexible grasp. In the simple act of picking up a pen to write with, we humans change from one grasp to another by rolling or sliding our fingers with respect to the pen. We would find it very awkward to screw or unscrew the lid of a jar, if we had to keep our fingers fixed on the lid. Thus, a wide variety of tasks require a changing grasp, where the end-effectors move with respect to the object. Moreover, the shape and size of many objects may preclude the possibility of clamping them rigidly. For instance, large (with respect to the manipulators) convex objects without handles cannot be grasped rigidly without squeezing or deforming them. Hence, in order to expand the range of tasks accomplished and objects manipulated, it is necessary to explore **Non-Rigid Grasps** i.e. grasps that permit relative motion between the end-effectors and the object. Such a grasp is relevant, not only in the case of dexterous manipulation with multifingered hands, but also when robotic arms handle an object using an enveloping grasp i.e. when the “effector” is distributed over, say, the end link of the arm and contact with the object may occur at any point on the link surface.

Non-rigid grasps are effected by essentially two kinds of relative motion between the end-effector and the object - Rolling and Sliding. The former occurs when there is relative angular velocity between the bodies in contact and the latter corresponds to relative translational velocity. Of course, the relative motion can be a combination of both rolling and sliding, too.
In this work, we address the problem of control of cooperative manipulation when the end-effectors maintain rolling contact with the object. In terms of the contact types described earlier, a rolling constraint could correspond to point contact with friction or a soft finger contact (Table 4.2.1) or a hard curved finger (fig. 4.3). It is well-known that three-dimensional rolling constraint equations are non-holonomic in nature i.e. the constraints imposed on the velocities of the contacting bodies are non-integrable. However, by commanding rolling motion instead of sliding motion, some advantages gained are [53]

- Wear and tear of the contacting bodies is much less.

- Maintaining a rolling contact is easier in the sense that the coefficient of friction need not be exactly known. Rolling motion can be achieved by exerting forces sufficiently close to the center of the friction cone.

- The nonholonomic nature of the rolling constraint makes the set of configurations reachable by rolling larger than that reachable by sliding.

### 4.3 Kinematics of Rolling Contact

In recent years, there have been several studies developing instantaneous kinematic equations relating relative rigid body motions to changes in the contact coordinates. In [14] and [15], the authors derive relationships between velocities and higher order derivatives for planar and spatial curves in point contact. Montana ([68], [69]) has derived a set of input-output equations which describe how the points of contact on the surfaces of the contacting bodies evolve in time in response to relative motion between the bodies, at the velocity level. Corresponding second-order relations have been obtained in [89]. The problem of determining the existence of an admissible path between two contact configurations and determining such a path, for rolling constraint has been studied in [53].

In this section, we briefly describe the first-order contact kinematics derived by Montana [68]. The basic formulations used by Montana in deriving these equations are interpreted here, in terms of differential geometric concepts such as shape operators and connection forms. This interpretation gives a better physical understanding of the quantities involved in the kinematic contact equations. These equations can be used to obtain the relevant relations for the particular case of rolling contact by setting the relative translational velocities to zero.
4.3.1 Some Differential Geometry

The definitions and notation in this section are taken from [53], [69] and [82].

An open subset of \( \mathbb{R}^2 \) is denoted by \( U \), with any point of \( U \) being denoted by \((u, v) \in \mathbb{R} \times \mathbb{R}\).

![Coordinate System Mapping](image)

**Figure 4.4** Coordinate System Mapping

**Definition 4.1** A surface \( S \) in \( \mathbb{R}^3 \) is a subset \( S \subset \mathbb{R}^3 \) such that for every point \( s \in S \), there exists an open subset \( S_s \) of \( S \) with the property

1. \( s \in S_s \)
2. \( S_s \) is the image of a \( C^3 \) map \( f : U \to \mathbb{R}^3 \)
3. \( f_u(u, v) \) and \( f_v(u, v) \) are linearly independent in \( \mathbb{R}^3 \) for all \((u, v) \in U\), where
   \[
   f_u(u, v) = \frac{\partial}{\partial u} [f(u, v)] ; \quad f_v(u, v) = \frac{\partial}{\partial v} [f(u, v)]
   \]
4. \( f : U \to S_s \subset \mathbb{R}^3 \) is a diffeomorphism.

\( S_s \) is called a coordinate patch of \( S \) and the pair \((f, U)\) is called a (local) coordinate system for \( S \) with coordinates \( u \) and \( v \). The collection of coordinate patches \( \{S_s\} \) which covers \( S \) i.e. \( S = \bigcup S_s \), is called an atlas of \( S \). A surface that can be covered by coordinate patches (i.e. can be written as their union) is said to be embedded in \( \mathbb{R}^3 \) with the set of coordinate patches that covers it, being its embedding in \( \mathbb{R}^3 \).
Definition 4.2 The Gauss map of a surface $S$ is a continuous map $n : S \to S^2$, where $S^2$ is a sphere of unit radius in $\mathbb{R}^3$, such that for every $s \in S$, $n(s)$ is orthogonal to $S$ at $s$. The Gauss map may also be denoted as $n \circ f : U \to S^2$.

Definition 4.3 A coordinate system $(f, U)$ is called orthogonal if $f_u \cdot f_v = 0$ and right-handed if $(f_u \times f_v)/\|f_u \times f_v\| = n \circ f(u, v)$.

Let $(f, U)$ be an orthogonal, right-handed coordinate system for a surface patch $S_s \subset S$. Then, the Gaussian frame at a point $s \in S_s$ is defined as the coordinate frame with origin at $f(u, v)$ and coordinate axes

$$
x(u, v) = \frac{f_u}{\|f_u\|}; \quad y(u, v) = \frac{f_v}{\|f_v\|}; \quad z(u, v) = n \circ f(u, v) \quad (4.1)
$$

Definition 4.4 Let $S_s$ be a coordinate patch of $S$, with an orthogonal coordinate system $(f, U)$. At a point $s \in S_s$, with $(u, v) = f^{-1}(s)$, the Curvature form $\mathcal{K}$ is defined as the $2 \times 2$ matrix

$$
\mathcal{K} = [x(u, v), y(u, v)]^T \begin{bmatrix} z_u(u, v) & z_v(u, v) \\ \|f_u\| & \|f_v\| \end{bmatrix} \quad (4.2)
$$

The Torsion form $T$ is the $1 \times 2$ matrix

$$
T = y(u, v)^T \begin{bmatrix} x_u(u, v) \\ \|f_u\| \end{bmatrix}, \quad y_v(u, v)^T \begin{bmatrix} x_v(u, v) \\ \|f_v\| \end{bmatrix} \quad (4.3)
$$
The Metric $\mathcal{M}$ is the $2 \times 2$ matrix

$$
\mathcal{M} = \begin{bmatrix}
\|f_u\| & 0 \\
0 & \|f_v\|
\end{bmatrix}
$$

(4.4)

A physically intuitive interpretation of the curvature form $\mathcal{K}$ in (4.2) defined by Montana [69] is obtained via the concept of the shape operator of the surface $S$ which is defined as [82]

![Shape Operator Diagram](image)

**Figure 4.6** Shape Operator

**Definition 4.5** If $s$ is a point of $S$, then for each tangent vector $w$ to $S$ at $s$,

$$
S_s(w) = \nabla_w N
$$

(4.5)

where $N$ is the unit normal vector field (i.e. normalized Gauss map) on a neighborhood of $s$ in $S$ and $\nabla_w N$ is the covariant derivative of $N$ with respect to $w$ at the point $s$, i.e.

$$
\nabla_w N = \frac{d}{dt} [N(s + tw)]_{t=0}
$$

Clearly, the covariant derivative $\nabla_w N$ measures the initial rate of change of $N(s)$ as $s$ moves in the $w$ direction. Since the tangent plane of $S$ at any point $s$ consists of all vectors orthogonal to $N(s)$, the shape operator $S_s(w)$ gives an infinitesimal description of the way the surface $S$ is curving in the $w$ direction. It can be shown that the shape operator is a linear map that maps the tangent plane of $S$ to itself (fig. 4.6), and hence it can be described in terms of the basis vectors of the tangent plane at $s$ viz. $x(u, v)$ and $y(u, v)$ defined in (4.1).
Lemma 4.1  If \((f, U)\) is an orthogonal coordinate system for a coordinate patch \(S_s\) of \(S\), then for any \(s \in S_s\), the curvature form \(\mathcal{K}\) defined in (4.2), is a matrix representation of the shape operator \(S_s\) at \(s\) in terms of the basis vectors \(x(u, v)\) and \(y(u, v)\).

(See appendix B for proof)

Just as the curvature form \(\mathcal{K}\) represents the curving of the surface in the form of the infinitesimal rotation of the normal vector, the torsion form \(T\) contains information about the infinitesimal rotation of the vectors in the tangent plane. The torsion form can also be defined using the covariant derivative in the following way.

Let us denote the vectors \(x(u, v)\), \(y(u, v)\) and \(z(u, v)\) in (4.1) by \(E_1\), \(E_2\) and \(E_3\) respectively. The three vectors are said to constitute an adapted frame field on a region in \(S \subset \mathbb{R}^3\) since at any point \(s \in \mathbb{R}^3\) in this region, they form a coordinate frame.

**Definition 4.6**  For each tangent vector \(w\) to the surface \(S\) at the point \(s\), the connection forms of the frame field \(E_1, E_2, E_3\) are defined as

\[
\omega_{ij}(w) = \nabla_w E_i \cdot E_j(s) \quad (1 \leq i, j \leq 3)
\]

(4.6)

Since the connection forms \(\omega_{ij}\) represent the dot product of a coordinate vector \(E_j\) with the rate of change of another coordinate vector \(E_i\), \(\omega_{ij}(w)\) is the initial rate at which \(E_i\) rotates toward \(E_j\) as \(s\) moves in the \(w\) direction. It can be shown that the connection forms satisfy the condition \([82]\)

\[
\omega_{ij} = -\omega_{ji} \quad (1 \leq i, j \leq 3)
\]

(4.7)

The skew-symmetry condition (4.7) implies that

\[
\omega_{11} = \omega_{22} = \omega_{33} = 0
\]

Hence, this condition has the effect of reducing the nine connection forms \(\omega_{ij}\) for \(1 \leq i, j \leq 3\) to essentially only three, say \(\omega_{12}, \omega_{13}\) and \(\omega_{23}\).

Since the connection forms are one-forms on the surface \(S\) (i.e. they are real-valued functions on the tangent vectors to \(S\) and are linear at each point), they can be expressed in terms of their component functions along the differentials \(du\) and \(dv\), i.e.

\[
\omega_{ij} = g_{ij}^1 \, du + g_{ij}^2 \, dv
\]
Hence, each connection form can be represented as a $1 \times 2$ vector as

$$\omega_{ij} = [g_{ij}^1 \ g_{ij}^2]$$

This representation of a connection form can be used to relate the connection form $\omega_{12}$ to the torsion and metric forms defined in (4.3) and (4.4) respectively.

**Lemma 4.2** In an orthogonal coordinate system, the torsion form $T$ in eq.(4.3) is related to the connection form $\omega_{12}$ as

$$T = \omega_{12} M^{-1}$$

where the metric $M$ is as defined in (4.4).

The connection forms, by definition, are the coefficient functions in the orthonormal expansion (in terms of $E_1$, $E_2$, $E_3$) of the covariant derivative of each coordinate vector $E_i$ with respect to any tangent vector $w$ i.e.

$$\nabla_w E_i = \sum_{i=1}^{3} \omega_{ij}(w) E_j(s)$$

As mentioned earlier, the frame field $E_1$, $E_2$, $E_3$ corresponds to the vectors $x(u,v)$, $y(u,v)$ and $z(u,v)$, with $E_3$ or $z$ being the unit normal to the surface. Hence, from the definition of the shape operator (4.5) and the definitions of the connection forms (4.6), it follows that

$$S(w) = -\omega_{13}(w) E_1(s) - \omega_{23}(w) E_2(s)$$

(4.8)

Thus, the curvature form, torsion form and the metric defined in eqs.(4.2), (4.3) and (4.4) encapsulate the information of the three non-trivial connection forms $\omega_{12}$, $\omega_{13}$ and $\omega_{23}$. This information essentially captures the infinitesimal rotation of the frame field $E_1$, $E_2$, $E_3$ at any point on the surface $S$ along any tangent vector.

**4.3.2 Contact Kinematics**

We now consider a finger\footnote{The term *finger* in this analysis refers to the part of the manipulator that is in contact with the object} and an object that move while maintaining contact with each other (fig. 4.7). The reference frames $C_f$ and $C_o$ are fixed with respect to the
finger and the object respectively. Let \( S_f \in \mathbb{R}^3 \) and \( S_o \in \mathbb{R}^3 \) be the embeddings of the finger and object surfaces relative to \( C_f \) and \( C_o \) respectively. Let \( n_f \) and \( n_o \) be the (outward normal) Gauss maps for \( S_f \) and \( S_o \). Choose atlases \( \{ S_{f,i} \}_{i=1}^{m_f} \) and \( \{ S_{o,j} \}_{j=1}^{m_o} \) for the respective surfaces. Let \( (f_{f,i}, U_{f,i}) \) be an orthogonal right-handed coordinate system for \( S_{f,i} \), with Gauss map \( n_f \). Similarly, let \( (f_{o,j}, U_{o,j}) \) be an orthogonal right-handed coordinate system for \( S_{o,j} \), with Gauss map \( n_o \).

Let \( c_f(t) \in S_f \) and \( c_o(t) \in S_o \) be the positions, at time \( t \), of the point of contact relative to \( C_f \) and \( C_o \) respectively. The analysis is restricted to an interval of time \( I \), such that \( c_f(t) \in S_{f,i} \) and \( c_o(t) \in S_{o,j} \) for all \( t \in I \), for some \( i \) and \( j \). The coordinate systems \( (f_{f,i}, U_{f,i}) \) and \( (f_{o,j}, U_{o,j}) \) induce a normalized Gaussian frame at all points in \( S_{f,i} \) and \( S_{o,j} \) respectively. Hence, we can define two continuous families of coordinate frames, for each \( t \in I \), as follows. Let \( C_{tf} \) and \( C_{to} \) be coordinate frames fixed relative to \( C_f \) and \( C_o \) respectively, such that they coincide with the normalized Gaussian
frames at \( c_f(t) \) and \( c_o(t) \) at time \( t \). These frames facilitate the representation of the instantaneous relative velocities between the two bodies. Thus, the motion of the finger relative to the object can be defined in terms of the velocity of \( C_{lf} \) with respect to \( C_{lo} \). Hence, let \( v_x, v_y, \) and \( v_z \) be the components of the translational velocity and \( \omega_x, \omega_y, \) and \( \omega_z \) be the components of the angular velocity of \( C_{lf} \) with respect to \( C_{lo} \) at time \( t \).

Using the coordinate systems defined above, we can also define the parameters describing the five degrees of freedom of the evolution of the contact points. Let

\[
(u_f, v_f) = f_{f,i}^{-1}(c_f(t)) \in U_{f,i} \quad ; \quad (u_o, v_o) = f_{o,j}^{-1}(c_o(t)) \in U_{o,j}
\]

These account for four degrees of freedom. The fifth parameter is the angle of contact \( \psi(t) \), which can be defined as the angle between the \( x \)-axes of \( C_{lf} \) and \( C_{lo} \).

Let the curvature form, connection form and the metric of the finger surface and object surface relative to the coordinate systems \((f_{f,i}, U_{f,i})\) and \((f_{o,j}, U_{o,j})\) at the points \( c_f(t) \) and \( c_o(t) \), at time \( t \), be \( \mathcal{K}_f, T_f, \mathcal{M}_f \) and \( \mathcal{K}_o, T_o, \mathcal{M}_o \) respectively. Also, let

\[
R_\psi = \begin{bmatrix}
\cos \psi & -\sin \psi \\
\sin \psi & -\cos \psi
\end{bmatrix} \quad ; \quad \tilde{\mathcal{K}}_o = R_\psi \mathcal{K}_o R_\psi
\]

\((\mathcal{K}_f + \tilde{\mathcal{K}}_o)\) is called the relative curvature form.

The following kinematic equations that describe the motion of the points of contact over the surfaces of the finger and the object in response to a relative motion between the two bodies are due to Montana [68].

**Theorem 4.1** At a point of contact, if the relative curvature form is invertible, then the point of contact and angle of contact evolve according to the following equations

\[
\dot{u}_f = \mathcal{M}_f^{-1}(\mathcal{K}_f + \tilde{\mathcal{K}}_o)^{-1} \left( \begin{bmatrix}
-\omega_y \\
\omega_x
\end{bmatrix} - \tilde{\mathcal{K}}_o \begin{bmatrix}
v_x \\
v_y
\end{bmatrix} \right)
\]

\[
\dot{u}_o = \mathcal{M}_o^{-1} R_\psi (\mathcal{K}_f + \tilde{\mathcal{K}}_o)^{-1} \left( \begin{bmatrix}
-\omega_y \\
\omega_x
\end{bmatrix} + \mathcal{K}_f \begin{bmatrix}
v_x \\
v_y
\end{bmatrix} \right)
\]

\[
\dot{\psi} = \omega_z + T_f \mathcal{M}_f \dot{u}_f + T_o \mathcal{M}_o \dot{u}_o
\]

\[
v_z = 0
\]
In particular, if the bodies maintain rolling contact with each other, this implies that the relative translational velocities are zero, i.e.

\[
\begin{bmatrix}
v_x \\
v_y
\end{bmatrix} = 0
\]

(4.9)

Additionally, if the bodies are not allowed to spin with respect to each other (pure rolling motion), then

\[
\omega_z = 0
\]

(4.10)

Substituting conditions (4.9) and (4.10) in the kinematic contact equations of theorem 4.1, we obtain the first order equations for pure rolling contact as

\[
\begin{bmatrix}
\dot{\mathbf{u}}_f \\
\dot{\mathbf{u}}_o \\
\dot{\psi}
\end{bmatrix} = \begin{bmatrix}
\mathcal{M}_f^{-1} \\
\mathcal{M}_o^{-1} \\
T_f + T_o R_\psi
\end{bmatrix} \begin{bmatrix}
\mathcal{K}_f + \mathcal{\hat{K}}_o \\
\omega_y \\
\omega_x
\end{bmatrix}^{-1}
\]

(4.11)

4.4 System Kinematics and Dynamics

The kinematic contact equations in the previous section specify the evolution of the points of contact on the surfaces of the end-effector and the object in response to a relative motion between the two bodies. In order to analyze and control the manipulation of a common object by several manipulators (fingers/arms) maintaining rolling contact, these equations can be used as the bridge between the kinematic and dynamic models of the manipulators and those of the object.

The control of multifingered hands with rolling contacts was first studied in [20] where, a control algorithm which tracks the object trajectory and keeps the contact forces within the friction cone was derived. This control law is valid for the case when the number of actuators on the manipulators is equal to the sum of the task space dimension and the dimension of the internal forces (for example, two planar two-jointed fingers or three spatial three-jointed fingers). In this case, the contact positions on the finger or object surfaces are not explicitly controlled. Control of object trajectory and contact conditions for rolling contact manipulation, using a nonlinear feedback control law has been studied in [51] and [110]. For two-arm manipulation, the authors suggest the use of a modified control law that allows the critical contact force between the two contacts, as one of the outputs. In [19], a nonlinear optimization problem is solved to obtain a motion planning strategy for sliding and rolling contacts. Tactile
feedback has been employed in [62] to control the manipulation of an unknown object with rolling contact. To date, no general control algorithm for simultaneously controlling the object trajectory, internal forces and the contact conditions, has been proposed in the literature.

The objective of the work in this chapter is to obtain such a control law. This will enable the manipulators to move the object in the desired fashion, while maintaining a stable and secure grasp and ensuring that the moving grasp locations do not result in awkward or unreachable configurations. To be able to devise such an algorithm that incorporates not only the object trajectory and the internal forces experienced by the object, but also the contact variables (described in the previous section), we first develop a unified dynamic formulation for the cooperating manipulator system, that includes the dynamics of the manipulators as well as the object. This formulation facilitates the incorporation of all the variables that need to be controlled in the dynamic equations in a natural way.

The fundamental concept that we will employ in obtaining this unified formulation is to model the rolling contacts between each end-effector and the object as (additional) unactuated joints of the respective manipulators. Thus, the object is considered to be partitioned into several parts, the number being equal to the number of manipulators. Each part is augmented to one of the manipulators as an additional link of that mechanism. The "joint" between the mechanism and this new link is the rolling contact present between the actual end-effector of the manipulator and the object. Since there is no torque-impacting actuator present at this joint, the joint is "unactuated", with the motion at the joint being caused by the action of the actuators at other joints.

The idea that the rolling contact between the end-effector and the object can be modeled as a joint stems from the following [78]. When a rigid surface $S$ rolls without sliding on another surface $S_1$ (fig. 4.8), then, at every moment of time, the velocity field of the moving body $S$ is the same as if it were rotating with a certain angular velocity $\omega$ about some axis passing through the point of contact. Depending on the direction of the instantaneous axis of rotation, we can distinguish between so-called pure (or proper) rolling and pivoting. Pure rolling occurs if the instantaneous axis of rotation lies in the tangent plane; pivoting, if the instantaneous axis of rotation is normal to the tangent plane. Hence, in the general case, the rolling of a surface $S$ on a surface $S_1$ can be decomposed into pure rolling and pivoting in accordance with the
decomposition of the vector $\omega$ into a component $\omega_t$ lying in the tangent plane and a component $\omega_n$ which is normal to the contacting surfaces (fig. 4.8).

Hence, conceptually, we can think of pure rolling motion to be induced by two revolute joints at the point of contact, with their axes orthogonal to each other (and lying in the tangent plane). Similarly, rolling motion with pivoting can be thought of as being caused by a three-axis "wrist" mechanism (i.e. three revolute joints with mutually orthogonal axes) at the point of contact of the two bodies.

Thus, when the end-effector of a robotic manipulator makes a rolling contact with the object being manipulated, the relative motion between the two can be thought of as being induced by two or three revolute (unactuated) joints at the point of contact. Therefore, in order to analyze the kinematic and dynamic relations of the resulting system, we can think of the object as being split into $L$ parts (corresponding to $L$ manipulators) through its center of mass, with each part attached as an additional link to each manipulator through this unactuated joint mechanism. The equations for the overall system can then be obtained by imposing the constraint that the object-links of all the arms are "glued" together at the center of mass of the object.

The relations in the following analysis are derived on the basis of the above conceptual model. The resulting equations provide a complete model of the cooperating manipulator system including the constraints imposed by the rolling contacts.

Consider the cooperating system to consist of $L$ manipulators, as shown in fig. 4.9. The end-effector of each manipulator makes a rolling contact with the object being manipulated. Thus, the characteristics of the system are defined by three kinds of
entities - the manipulators, the object and the interface between the manipulator end-effectors and the object. By considering the rolling contacts to be unactuated joints between each end-effector and a portion of the object (as described above), we are essentially envisioning the system to be made up of $L$ similar entities or mechanisms (which shall be referred to as *virtual manipulators*) whose tips (which correspond to the object center of mass) are joined together. Each mechanism consists of links connected in series by (actuated and unactuated) joints. Hence, we can obtain the equations representing the system by, first deriving the relations for each individual virtual manipulator and then grouping these together while imposing the constraint that their tips must move together.

### 4.4.1 Kinematics of the Virtual Manipulator

In order to relate the velocity of the tip of the virtual manipulator to the velocities of its joints, we derive the Jacobian matrix for the virtual manipulator. This is done using the Jacobian of the original manipulator and the rolling contact kinematics.
Figure 4.10 Rolling contact between end-effector and object

Let $\mathbf{x}_o$ denote the position of the object center of mass in the base frame of reference and let $\mathbf{x}_f^i$ denote the position of the real end-effector of the $i$-th manipulator (fig. 4.10). Let $\mathbf{x}_{oc}^i$ and $\mathbf{x}_{fc}^i$ denote the positions of the points of contact on the object and the $i$-th end-effector respectively, expressed in the base frame. Then, the velocities at the $(i$-th) points of contact can be computed as

$$\mathbf{\dot{x}}_{fc}^i = T_j^i \mathbf{\dot{x}}_f^i ; \quad \mathbf{\dot{x}}_{oc}^i = T_o^i \mathbf{\dot{x}}_o$$

(4.12)

where, the matrices $T_j^i$ and $T_o^i$ (for the spatial case) are of the form

$$T = \begin{bmatrix} I_{3 \times 3} & S \\ 0_{3 \times 3} & I_{3 \times 3} \end{bmatrix} \quad \text{with} \quad S = \begin{bmatrix} 0 & p_z & -p_y \\ -p_z & 0 & p_x \\ p_y & -p_x & 0 \end{bmatrix}$$

(4.13)

Here, $p_x, p_y, p_z$ are the components of the vector from the end-effector or object reference frame to the respective point of contact, expressed in the base frame. For the $i$-th contact, they are functions of the contact variables $y_f^i \in \mathbb{R}^2$ and $y_o^i \in \mathbb{R}^2$ used to describe the surfaces of the end-effector and the object respectively as well as of the joint variables $\theta^i$ of the actual manipulator.

The positions and velocities of the actual joints of the $i$-th manipulator are denoted by the vectors $\mathbf{\theta}^i$ and $\mathbf{\dot{\theta}}^i$ respectively. Since the end-effector and the object maintain rolling contact, the relative translational velocity between the two bodies is zero. Let $\mathbf{\omega}^i$ be the vector of relative angular velocities between the $i$-th end-effector and the object. (We define only the relative velocity vector, and not the relative position vector, due to the fact that rolling imposes constraints on the velocities and not the
positions of the contacting bodies.) Then, the augmented joint velocity vector for the 
i-th virtual manipulator is

$$\dot{q}^i = \begin{bmatrix} \dot{\theta}_i^T \\
(\omega_r^i)^T \end{bmatrix}^T$$

Notice that in the planar case, $\omega_r^i$ will be one dimensional for each manipulator as the bodies can only roll about the axis perpendicular to the plane in which they lie. In the spatial case, if pure rolling is considered (i.e. there is no rotation about the normal to the contacting surfaces), the relative angular velocity vector $\omega_r^i$ will be two-dimensional while in the case of rolling with pivoting, it will be three-dimensional (fig. 4.8). In terms of the relative angular velocities in the equations in Theorem (4.1)

$$\omega_r^i = [\omega_x \omega_y \omega_z]^T$$

Using the relative angular velocities, the velocities at the points of contact on the finger and the object surfaces are related as

$$\dot{x}_o = \dot{x}_f + \begin{bmatrix} 0^T \\
(\omega_r^i)^T \end{bmatrix}^T$$

Using (4.12),

$$T_o^i \dot{x}_o = T_o^i \dot{x}_f + \begin{bmatrix} 0^T \\
(\omega_r^i)^T \end{bmatrix}^T$$

Then, multiplying by $(T_o^i)^{-1} = [\hat{T}_o^i \hat{T}_o^i]$, we have

$$\dot{x}_o = (T_o^i)^{-1} T_j^i \dot{x}_j + \begin{bmatrix} 0^T \\
(\omega_r^i)^T \end{bmatrix}^T$$

However, as we have seen in the earlier chapters, the velocity of the real end-effector of each manipulator can be expressed as a function of the joint positions and joint velocities using the manipulator Jacobian $J_m^i$ as

$$\dot{x}_f = J_m^i \dot{\theta}^i$$

Hence, substituting this relation in the equation above and grouping the terms together, we obtain

$$\dot{x}_o = (T_o^i)^{-1} T_j^i J_m^i \dot{\theta}^i + \begin{bmatrix} 0^T \\
(\omega_r^i)^T \end{bmatrix}^T$$

$$= (T_o^i)^{-1} T_j^i J_m^i \dot{\theta}^i + \hat{T}_o^i \omega_r^i$$

$$= \begin{bmatrix} (T_o^i)^{-1} T_j^i J_m^i \\
\hat{T}_o^i \end{bmatrix} \dot{\theta}^i$$
Thus, for each virtual manipulator, the tip velocity can be written in terms of the augmented Jacobian $J^i$ and the augmented joint velocity vector $\dot{\theta}^i$ as

$$\ddot{\mathbf{x}}_o = J^i \ddot{\theta}^i; \quad J^i = [(T_o)^{-1}T_f, J^i_m, \hat{T}_{o2}^i]$$ (4.14)

The Jacobian $J^i$ of the $i$-th virtual manipulator is a function of its joint variables $\theta^i$ and the contact variables at the $i$-th contact $u_f^i$ and $u_o^i$. Differentiating (4.14) yields

$$\ddot{x}_o = J^i \ddot{\theta}^i + J^i \left[ \begin{array}{c} \dot{\theta}^i \\ \dot{u}_o^i \\ \dot{u}_f^i \end{array} \right]$$

However, using eq.(4.11), the velocities $\dot{u}_o^i$ and $\dot{u}_f^i$ can be written in terms of the relative angular velocities $\omega^i$. Thus, we can write

$$\left[ \begin{array}{c} \dot{\theta}^i \\ \dot{u}_o^i \\ \dot{u}_f^i \end{array} \right] = \bar{M}^i \ddot{q}^i$$

where the matrix $\bar{M}$ can be written from eqs.(4.11), so that

$$\ddot{x}_o = J^i \ddot{q}^i + \tilde{J}^i \ddot{q}^i$$

with $\tilde{J}^i = J^i \bar{M}^i$.

Then, concatenating the equations for all the manipulators,

$$E \ddot{x}_o = J \ddot{q} + \tilde{J} \ddot{q}$$ (4.15)

where $E$ is a matrix of stacked identities, i.e.

$$E = \left[ \begin{array}{c} I \\ \vdots \\ I \end{array} \right]$$

$J$ and $\tilde{J}$ are block diagonal matrices with the $J^i$'s and $\tilde{J}^i$'s along their respective diagonals and $\dot{q} = [(q^1)^T \ (q^2)^T \ \ldots \ (q^L)^T]^T$. Note that the incorporation of the relative velocities in the vector $\dot{q}$ will make eq.(4.15) underdetermined if each manipulator has (at least) as many joints as the task space dimension. This affords a larger choice in planning the joint (and relative) accelerations for a given object trajectory, and will be useful when some secondary subtask has to be performed (see section 4.5).
4.4.2 Dynamics of the Virtual Manipulator

Each virtual manipulator forms an open kinematic chain, with the links and the object part connected serially via joints. The only variation of this structure from the serial open-chain robotic arm studied previously is the fact that the last "joint" (corresponding to the rolling motion) has multiple degrees of freedom in the spatial case.

Several methods of deriving the dynamic equations of an open-link serial robotic arm have been studied in the literature [94]. Two well-known methods are -
(a) *Euler-Lagrange* equations which describe the time evolution of a mechanical system subject to holonomic constraints. The determination of these equations involves the formation of the system Lagrangian, which is the difference of the kinetic and potential energy of the system

(b) *Newton-Euler formulation* which involves the motion analysis of each link and uses a forward-backward recursion to obtain the equations for the whole manipulator.

We use a graph-theoretic method proposed in [106] for general multi-body systems, to derive the dynamic equations for the virtual manipulator. This method has been used rarely (if ever) for the derivation of robot manipulator dynamics. The main motivation for the choice of this method is that it facilitates the incorporation of the nonholonomic constraints present at the rolling contact. This method is also convenient because multiple degrees of freedom of motion between adjoining bodies can be treated with ease and the equations can be derived from the known dynamic models of the cooperating manipulators.

We first derive the dynamic equations for an individual virtual manipulator. These equations are then concatenated and the constraints associated with the fact that the tips of all the virtual manipulator are coincident, are incorporated to obtain the dynamics of the whole system.

Recall that each virtual manipulator consists of the actual manipulator augmented with a portion of the object as the last link. When all the manipulators are identical (and hence have equal load-carrying capacity), the mass of the object can be divided equally among all the parts i.e. each part is assumed to have mass $m_o/L$ where $m_o$ is the mass of the object. In cases where the load-carrying capacities of the manipulators are different, each part can be assigned to have mass proportional to its manipulator's load-carrying capacity.
Some definitions and introductory comments are first presented which set up the framework for the graph-theoretic analysis of the manipulator.

In a multi-body system, two bodies are said to be contiguous if and only if they exert force on each other directly. The coupling between two contiguous bodies is called a hinge. This word is used to represent any kind of coupling allowing relative rotational and/or translational motion between contiguous bodies, and it need not necessarily be a material connection. The hinge between two contiguous bodies encapsulates all the interaction forces/moments between them so that there exists exactly one hinge for each pair of contiguous bodies. Furthermore, for each hinge there exists only one pair of contiguous bodies.

For the virtual manipulator, each adjacent pair of links forms a pair of contiguous bodies while the joint between them is the corresponding hinge. The hinge connecting the object portion to the real end-effector is characterized by the rolling motion between the two bodies. The first link is connected to the ground at one end. The ground or the base is taken to be the zero-th body. Let \( n \) be the number of bodies(links) in the manipulator (not counting the base, but including the object portion as the last link). Correspondingly, the manipulator has \( n \) hinges, counting the hinge between the base and the first link. To set up the directed system graph, bodies and hinges are labeled separately. The graph consists of points called vertices, which represent the bodies of the system, and of lines connecting the vertices, called edges, which represent the hinges. To every edge of the graph, a sense of direction is assigned, which is indicated by an arrow. This helps to distinguish the two vertices connected by any arc thus enabling the unambiguous specification of the motion and force relative to one of the bodies. Fig.(4.11) shows the system graph for a virtual manipulator consisting of three links and the object portion. The vertices are denoted by \( s_i \), \( i = 0, \ldots, n \) and the arcs by \( u_a \), \( a = 1, \ldots, n \).

![Directed System Graph for 3-link arm](image)

**Figure 4.11** Directed System Graph for 3-link arm
For each hinge $a$ where $a = 1, 2, \ldots, n$, the structure of the directed graph uniquely defines two integer functions $i^+(a)$ and $i^-(a)$. $i^+(a)$ is the index of the vertex from which the arc $u_a$ is pointing away while $i^-(a)$ is the index of the vertex toward which the arc $u_a$ is pointing. For the directed graph of fig (4.11), the two functions are

The same information can also be contained in a matrix called the incidence matrix of the directed graph, whose elements $S_{ia}$ are defined as follows, for $i = 0, \ldots, n$ and $a = 1, \ldots, n$,

$$
S_{ia} = \begin{cases} 
+1 & \text{if } i = i^+(a) \\
-1 & \text{if } i = i^-(a) \\
0 & \text{otherwise}
\end{cases}
$$

(4.16)

Each column of the incidence matrix contains exactly one element equal to $+1$ and one element equal to $-1$ and corresponds to one of the arcs of the graph. The matrix is partitioned into sub-matrices $S_0$ and $S$, with $S_0$ being the top row

$$
S_0 = [S_{01} \ldots S_{0n}]
$$

and $S$ being the $n \times n$ square matrix

$$
S = \begin{bmatrix}
S_{11} & \ldots & S_{1n} \\
& \ddots & \\
S_{n1} & \ldots & S_{nn}
\end{bmatrix}
$$

Another $n \times n$ matrix $T$ can be defined for the system graph. In this matrix, the rows correspond to the arcs while the columns correspond to the vertices of the graph. The elements $T_{ai}$ of this matrix are defined for $a, i = 1, 2, \ldots, n$ as

$$
T_{ai} = \begin{cases} 
+1 & \text{if } u_a \text{ belongs to the path between } s_0 \text{ and } s_i \\
& \text{and is directed toward } s_0 \\
-1 & \text{if } u_a \text{ belongs to the path between } s_0 \text{ and } s_i \\
& \text{and is directed away from } s_0 \\
0 & \text{if } u_a \text{ does not belong to the path between } s_0 \text{ and } s_i
\end{cases}
$$

(4.17)

<table>
<thead>
<tr>
<th>$a$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i^+(a)$</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>$i^-(a)$</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
Note that the matrices \( T \) and \( S \) are inverses of each other. These matrices help interrelate the quantities (such as motions and forces) associated with the individual bodies\(^*\) of the system, to obtain the coupled dynamic equations of the entire system.

One way to develop the dynamic equations is to apply cuts at each of the hinges and derive the equations of motion for each individual body based on Newton’s law and the law of moment of momentum for a single rigid body. This approach requires some information about the direction of constraint forces and torques in the hinges on the one hand and about the kinematics of rotational and translational motions of contiguous bodies relative to one another, on the other. In order to be able to account for arbitrary constraints at the hinges (especially the one formed due to contact between the end-effector and the object), we choose the approach using the method of analytical mechanics and d’Alembert’s principle.

For each body \( i, i = 1, \ldots, n \), define the following quantities -

- \( m_i \): Mass of the body
- \( r_i \): Radius vector of its center of mass from a fixed reference point
- \( L_i \): Absolute angular momentum with respect to its center of mass
- \( J_i \): Moment of Inertia
- \( \omega_i \): Absolute angular velocity of the body
- \( F_i \): Resultant external force acting on the body
- \( M_i \): Resultant external moment acting on the body

Then, for the system of \( n \) rigid bodies, d’Alembert’s principle can be written in the form

\[
\sum_{i=1}^{n} \left[ \delta r_i \cdot (F_i - m_i \dot{r}_i) + \delta \pi_i \cdot (M_i - \dot{L}_i) \right] + \delta W = 0
\]

(4.18)

where \( \dot{L}_i = J_i \cdot \dot{\omega}_i + \omega_i \times J_i \cdot \omega_i \). The vectors \( \delta r_i \) and \( \delta \pi_i \) describe variations of the position and orientation of body \( i \) with respect to the inertial reference frame, with time \( t \) fixed [106]. Note that body \( 0 \) is excluded as it is assumed to be stationary.

The term \( \delta W \) represents the total virtual work done in the hinges of the system.

In the presence of constraints in the hinges, the variations \( \delta r_i \) and \( \delta \pi_i \) are not independent of each other. Hence, it is necessary to express these variations in terms

\(^*\)If the last row and column in \( T \) and \( S \) corresponding to the last hinge and body in the virtual manipulator is removed, then these matrices will represent the actual manipulator. By thus, partitioning the matrices, the dynamic equations of the virtual manipulator may be written in terms of the dynamics of the original manipulator, if already known.
of variations of the generalized coordinates which are independent. This is a purely kinematic problem. Note that the dynamics of the system is entirely contained in eq. (4.18) which can be rewritten in matrix form as

\[
\delta \mathbf{r}^T \cdot (\mathbf{F} - m \ddot{\mathbf{r}}) + \delta \pi^T \cdot (\mathbf{M} - \mathbf{J} \cdot \dot{\omega} - \mathbf{V}) + \delta W = 0
\]  

(4.19)

where

\[
\begin{align*}
\delta \mathbf{r} & = [\delta r_1 \ldots \delta r_n]^T \\
\delta \pi & = [\delta \pi_1 \ldots \delta \pi_n]^T \\
\delta \mathbf{F} & = [\delta F_1 \ldots \delta F_n]^T \\
\mathbf{M} & = [M_1 \ldots M_n]^T \\
\mathbf{V} & = [V_1 \ldots V_n]^T \\
\mathbf{J} & = \begin{bmatrix} J_1 & \cdots & J_n \end{bmatrix} \\
\mathbf{m} & = \begin{bmatrix} m_1 & \cdots & m_n \end{bmatrix}
\end{align*}
\]

We will now investigate the kinematics of two contiguous bodies relative to one another (in terms of the generalized coordinates), with special attention to the case when the two bodies are the end-effector and the object.

Fig. (4.12) shows two bodies \(i^+(a)\) and \(i^-(a)\) coupled by hinge \(a\) \((a = 1, \ldots, n)\). Body-fixed frames \(\mathbf{e}_{i^+(a)}\) and \(\mathbf{e}_{i^-(a)}\) are located at the respective centers of mass \(C_{i^+(a)}\) and \(C_{i^-(a)}\). The number of degrees of freedom of motion of the bodies relative to each other is \(n_a\) for hinge \(a\). Note that \(1 \leq n_a \leq 6\). An equal number of generalized coordinates \(q_{a1}, \ldots, q_{an_a}\) is required to describe the motion of the bodies relative to one another.

To describe the position of body \(i^-(a)\) relative to body \(i^+(a)\), it is sufficient to specify two quantities as functions of the generalized coordinates -

(a) The radius vector in the \(C_{i^+(a)}\) frame of a single point \((P)\) fixed on body \(i^-(a)\), and

(b) The transformation matrix \(G_a\) between the frames at \(C_{i^+(a)}\) and \(C_{i^-(a)}\) which is defined by the equation \(\mathbf{e}_{i^-(a)} = G_a \mathbf{e}_{i^+(a)}\).

The point \(P\) fixed on body \(i^-(a)\) is called the hinge point. The location of the hinge point on body \(i^-(a)\) is specified by the body-fixed vector \(\mathbf{c}_{i^-(a)}\) which originates from \(C_{i^-(a)}\) (fig. 4.12). The variable radius vector from \(C_{i^+(a)}\) to the hinge point is split into two parts. One part is the vector \(\mathbf{c}_{i^+(a)}\) which is fixed in body \(i^+(a)\) while the other part is called the vector \(\mathbf{z}_a\). Only the vector \(\mathbf{z}_a\) is a function of the
Figure 4.12  Kinematics of contiguous bodies

generalized coordinates $q_{a1}, \ldots, q_{an}$. (For each hinge $a$, the vector $\xi_a$ is zero if $i$ equals neither $i^+(a)$ nor $i^-(a)$. Otherwise $\xi_a$ is the vector from the center of mass of body $i$ to the hinge point $a$ on this body.)

In our virtual manipulator, we have essentially two kinds of hinges - one is the actual joint of the robotic manipulator which connects two adjacent links and the other is the hinge formed by the rolling contact between the end-effector and the object. The actual joints of the robotic manipulator are modeled as one degree-of-freedom revolute or prismatic joints. For revolute joints, the hinge point is taken to be the geometric center of the joint, so that the hinge vector $z_a$ is zero. For prismatic joints, the hinge vector will be along the direction of motion of the joint with its length equal to the joint variable. The body-fixed frame for each link of the manipulator is fixed according to the Denavit-Hartenberg convention [94]. Then, the transformation matrix $G_a$ for each of the actual joints can be written using the Denavit-Hartenberg parameters.

Of special interest, is the hinge formed by the rolling contact between the end-effector and the object. For this hinge, the body $i^-(a)$ is the real end-effector of the manipulator while the body $i^+(a)$ is the portion of the object which has been
augmented to form the virtual manipulator. By our discussion earlier, this hinge instantaneously behaves as a multiple degree-of-freedom revolute joint at the point of contact of the two bodies. Note that although the point of contact on the two bodies changes from instant to instant, at any given instant, the relative motion between the two bodies is equivalent to that which would have been obtained had the bodies been connected by (one or more) single degree-of-freedom revolute joint(s) at the point of contact. Hence, similar to the revolute joints of the manipulator, the hinge point for the rolling contact hinge at any instant, is the point of contact between the two bodies and the hinge vector $\vec{z}_a$ is zero. The transformation matrix $G_a$ for this hinge is obtained via the transformation matrices between the body reference frames ($C_i^-(a)$ and $C_i^+(a)$) and the body-fixed frames at the respective points of contact which coincide with the respective Gaussian frames at the instant under consideration.

Having obtained a description of the position ($\vec{z}_a$) and orientation ($G_a$) of body $i^-(a)$ in the reference frame of body $i^+(a)$, the velocity distribution of body $i^-(a)$ is uniquely defined by two vectors viz., the velocity of the hinge point $a$ and the angular velocity of $i^-(a)$, both relative to $e^{i^+(a)}$. The first vector is the derivative of $\vec{z}_a$ in the base $e^{i^+(a)}$, denoted by $\dot{\vec{z}}_a$, with the open circle distinguishing it from the derivative of $\vec{z}_a$ in inertial space. Thus, we can write,

$$\vec{z}_a = \sum_{i=1}^{n_a} k_{ai} \vec{q}_{ai} \quad \dot{\vec{z}}_a = \sum_{i=1}^{n_a} \dot{k}_{ai} \vec{q}_{ai} + s_a$$

where

$$k_{ai} = \frac{\partial \vec{z}_a}{\partial q_{ai}}; \quad s_a = \sum_{i=1}^{n_a} \sum_{j=1}^{n_a} \frac{\partial^2 \vec{z}_a}{\partial q_{ai} \partial q_{aj}} \dot{q}_{ai} \dot{q}_{aj}$$

Similarly, the relative angular velocity $\Omega_a$ for each hinge can be written in the general form

$$\Omega_a = \sum_{i=1}^{n_a} p_{ai}(q_{a1}, \ldots, q_{ana}) \dot{q}_{ai} \quad \dot{\Omega}_a = \sum_{i=1}^{n_a} p_{ai} \dot{q}_{ai} + w_a$$

where

$$w_a = \sum_{i=1}^{n_a} \sum_{j=1}^{n_a} \frac{\partial p_{ai}}{\partial q_{aj}} \dot{q}_{ai}$$

For revolute joints of the manipulator, since $z_a$ is zero, the vectors $\dot{z}_a$ and $\ddot{z}_a$ will also be zero. Also, $p_{a1} = [0 \ 0 \ 1]$, as the Denavit-Hartenberg convention makes the revolution of the joint about the z-axis of the link. On the other hand, for prismatic joints, the vector $k_{a1} = [0 \ 0 \ 1]$, since the motion of the prismatic joint will be along
the z-axis. In this case, the vectors $\Omega_a$ and $\dot{\Omega}_a$ will be zero, as the links connected by a prismatic joint do not have any angular velocity relative to each other.

For the rolling contact hinge, once again, $\dot{z}_a$ and $\ddot{z}_a$ are zero, since $\dot{z}_a = 0$. Note that this is a mathematical representation of the fact that the relative translational velocity between the two bodies connected by this hinge is zero. Earlier, we represented the relative angular velocity of the two bodies at the point of contact using the vector $\omega_c$. This vector can be used to define the generalized coordinates of the rolling contact hinge, as it exactly characterizes the degrees of freedom of relative motion allowed between the two contacting bodies. Thus, the vectors $\Omega_a$ and $\dot{\Omega}_a$ for this hinge can be written as

$$\Omega_a = \sum_{i=1}^{n_a} p_{ai} \omega_{ri} ; \quad \dot{\Omega}_a = \sum_{i=1}^{n_a} \dot{p}_{ai} \omega_{ri}$$

The information contained in the expressions for $\dot{z}_a$ and $\Omega_a$ can be conveniently written in matrix form as

$$k^T = \begin{bmatrix}
  k_{11} & k_{1,2} & \cdots & 0 \\
  k_{21} & k_{2,2} & \cdots & \vdots \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & \cdots & k_{n,1} & k_{n,n} \\
\end{bmatrix}$$

$$p^T = \begin{bmatrix}
  p_{11} & p_{1,2} & \cdots & 0 \\
  p_{21} & p_{2,2} & \cdots & \vdots \\
  \vdots & \vdots & \ddots & \vdots \\
  0 & \cdots & p_{n,1} & p_{n,n} \\
\end{bmatrix}$$

Using the various quantities described above, the variations $\delta_T$ and $\delta_\pi$ can be expressed in terms of the generalized coordinates of the various hinges. By substituting these expressions in (4.19), the equations of motion of the virtual manipulator can be obtained in the final form as [106]

$$A^i \ddot{q}^i = \ddot{B}^i$$

where

$$A^i = [p \times (C + Z)T - kT] \cdot m \cdot [p \times (C + Z)T - kT]^T + (p_T) \cdot L \cdot (p_T)^T$$
\[ \vec{B}^i = \left[ \vec{p} \times T(C + Z)T - kT \right] \cdot (F - mU) - pT \cdot \left[ M + J \cdot (T^T J) - V \right] - k_x X - pY \] (4.23)

(In the above expressions, the superscript \( i \) has been omitted for the terms on the right-hand side, for enhancing readability). Most of the terms on the right-hand sides have been defined in eqs. ((4.16), (4.17), (4.19), (4.20) and (4.21). The matrices \( C \) and \( Z \) are formed respectively from the vectors \( \vec{e}_{iz} \) and \( \vec{z}_a \) defined earlier for each hinge \( a \). Thus,

\[
C = \begin{bmatrix}
C_{11} & \cdots & C_{1n} \\
\vdots & & \vdots \\
C_{n1} & \cdots & C_{nn}
\end{bmatrix}
\]

where the elements

\[
C_{ia} = S_{iz} \vec{e}_{iz} \quad \text{for } i, a = 1, \ldots, n
\]

For ease of defining the matrix \( Z \), we use a set of scalars \( S_{iz}^+ \)

\[
S_{iz}^+ = \begin{cases} 
+1 & \text{if } i = i^+(a) \\
0 & \text{otherwise}
\end{cases} \quad \text{(4.24)}
\]

Then,

\[
Z = \begin{bmatrix}
Z_{11} & \cdots & Z_{1n} \\
\vdots & & \vdots \\
Z_{n1} & \cdots & Z_{nn}
\end{bmatrix}
\]

with the elements

\[
Z_{ia} = S_{iz}^+ \vec{z}_a \quad \text{for } i, a = 1, \ldots, n
\]

\( X \) and \( Y \) are column matrices whose elements \( X_a \) and \( Y_a \) are the force and the torque acting at each hinge \( a \). The convention used in defining these quantities is that \(+X_a\) and \(+Y_a\) are acting on body \( i^+(a) \) while \(-X_a\) and \(-Y_a\) are acting on body \( i^-(a) \).

In order to write the equations in the standard form that is used for robotic manipulators, we split the vector \( \vec{B}^i \) as

\[
\vec{B}_i = \vec{\tau}^i - \vec{B}^i
\]

where \( \vec{\tau}^i \) the vector of torques/forces at the (actuated and unactuated) joints. Thus, the dynamic model for the \( i \)-th manipulator can be written in the form

\[
\mathbf{A^i} \ddot{q}^i + \mathbf{B}^i = \vec{\tau}^i \quad \text{(4.25)}
\]
As is evident from eqs. (4.22) and (4.23), $A^i$, the inertia matrix, is a function of the joint and contact positions while $B^i$ is the vector of Coriolis, Centrifugal and gravity terms, which are functions of position and velocity variables.

### 4.4.3 Concatenating the equations

Having derived the equations of motion for the individual virtual manipulators, the equations for the whole system can be obtained by concatenating the equations for the individual arms and imposing the constraint that the tips of all the manipulators (which correspond to the object center of mass) have to move together.

From eq. (4.14), we know that the tip velocity for each virtual manipulator can be written in terms of its joint coordinates as

$$\dot{x}_o = J^i \dot{q}^i$$

where $J^i$ is as defined in eq. (4.14). Hence, for all the tips to move together$^{11}$,

$$J^1 \dot{q}^1 = J^2 \dot{q}^2 = \ldots = J^L \dot{q}^L$$  \hspace{1cm} (4.26)

In matrix-vector form, this constraint may be written (for example) as

$$\begin{bmatrix}
J^1 & 0 & \ldots & -J^L \\
0 & J^2 & \ldots & -J^L \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \ldots & J^{L-1} & -J^L
\end{bmatrix} \dot{q} \triangleq H \dot{q} = 0$$  \hspace{1cm} (4.27)

It must be noted that there are various ways of translating the equations in (4.26) to the matrix form $H$ in (4.27). The manner in which the constraint equations are represented in $H$ determines the physical interpretation of the Lagrange multipliers which get associated with these constraints.

Let $A = \text{diag}\{A^1, A^2, \ldots, A^L\}$ and $B = [(B^1)^T (B^2)^T \ldots (B^L)^T]^T$. Then the dynamic equations for the system can be written as

$$A \ddot{q} + B + H^T \mathcal{F} = \tau$$  \hspace{1cm} (4.28)

where $\tau = [(\tau^1)^T (\tau^2)^T \ldots (\tau^L)^T]^T$ and $\mathcal{F}$ is the vector of Lagrange multipliers that arises due to the imposition of constraint (4.27). The components of $\mathcal{F}$ are the (internal) forces (and moments)$^{11}$ exerted by the object parts on each other.

$^{11}$Notice that we have implicitly assumed this constraint in writing eq. (4.14), by not using the superscript $i$ with the velocity $\dot{x}_o$.

$^{11}$In the following analysis, unless otherwise specified, the term “force(s)” includes both forces and moments.
As mentioned above, the exact physical interpretation of these forces depends on the composition of the constraint matrix $H$. To better understand this physical interpretation, consider the following.

![Diagram of robots and forces](image)

**Figure 4.13  Internal Forces at Object Center of Mass**

The tip of the $i$-th virtual manipulator exerts a force $E^i$ on the rest of the system, to which it is connected via the tips of the other virtual manipulators (fig. 4.13). The joint torques required to produce this force at the $i$-th virtual end-effector are given by\cite{94}

$$\tau^i_F = [J^i]^T E^i$$

Hence, the dynamic equations (4.25) for the $i$-th virtual manipulator get modified as

$$\tau^i = A^i \ddot{q}^i + B^i + \tau^i_F$$

$$= A^i \ddot{q}^i + B^i + [J^i]^T E^i$$

Then, concatenation of all the equations for $i = 1, \ldots, L$ yields

$$\tau = A\ddot{q} + B + \tau_F$$

(4.29)
Here $A$ and $B$ are as defined before. The vector $\tau_F^i$ is given by

$$\tau_F = \begin{bmatrix}
\tau_F^1 \\
\vdots \\
\tau_F^L
\end{bmatrix} = \begin{bmatrix}
[J^1]^T \\
\vdots \\
[J^L]^T
\end{bmatrix} \begin{bmatrix}
\bar{F}^1 \\
\vdots \\
\bar{F}^L
\end{bmatrix} \quad (4.30)$$

Due to the interaction of all the manipulators, the forces $\bar{F}^i$ are not independent of each other. Their mutual dependence can be determined from the fact that the tip of each virtual manipulator is the object center of mass. The net force at this point is equal to the inertial force required to move the object, which is the sum of the inertial forces required to move each object part. Since these inertial forces have already been taken into account in the dynamic equations (4.25), the resultant of all the forces $\bar{F}^i$ must be zero. Thus, in particular

$$\bar{F}^L = -(\bar{F}^1 + \bar{F}^2 + \ldots + \bar{F}^{L-1}) \quad (4.31)$$

Then, replacing $\bar{F}^L$ in (4.30) with the expression in (4.31),

$$\tau_F = \begin{bmatrix}
(J^1)^T \\
\vdots \\
(J^L)^T
\end{bmatrix} \begin{bmatrix}
\bar{F}^1 \\
\vdots \\
\bar{F}^{L-1}
\end{bmatrix} = \begin{bmatrix}
I \\
\vdots \\
-I
\end{bmatrix} \begin{bmatrix}
\bar{F}^1 \\
\vdots \\
\bar{F}^{L-1}
\end{bmatrix} = H^T \bar{F}$$

Substituting this expression for $\tau_F$ in eq.(4.29), we obtain the same equations as in (4.28). Thus, the elements of the Lagrange multiplier vector $\bar{F}$ in eq.(4.28) constitute the forces exerted by the object parts $1$ through $L - 1$ at the object center of mass, when $H$ is as defined in (4.27). In section 4.4.4, we will relate these forces to the forces exerted by the actual end-effectors of the manipulators on the object at the respective points of contact.
In (4.28), note that the torque vector \( \tau \) contains the torques corresponding to the actuated joints of the manipulator as well as the unactuated joints at the rolling contacts. The values of the torque inputs at the unactuated joints is zero. Hence, the equations in (4.28) can be rearranged according to the actuated and unactuated components as

\[
A_a \ddot{\mathbf{q}} + B_a + H_a^T F_a = \tau_a \quad (4.32)
\]
\[
A_u \ddot{\mathbf{q}} + B_u + H_u^T F_u = \tau_u = 0 \quad (4.33)
\]

The rows corresponding to the unactuated components (eq. 4.33) represent dynamic constraints arising due to the presence of the rolling contacts. In fact, the satisfaction of these constraints implies that the moments, about the directions corresponding to the relative angular velocities present between the contacting bodies, are not transmitted through the contact.

Alternatively, we can write the dynamic equations more compactly in the form

\[
A \ddot{\mathbf{q}} + B + H^T F = R \tau_a \quad (4.34)
\]

where \( R \) is a matrix that filters out the unactuated components. Thus, if the number of actuated joints is \( m \) and the dimension of \( \ddot{\mathbf{q}} \) is \( n \) (\( m < n \)), then the matrix \( R \) has dimensions \( n \times m \), with its diagonal elements equal to 1 and all non-diagonal elements equal to zero.

Eqs. (4.15) and (4.34) represent the complete kinematic and dynamic model of the system, including the constraints arising from the rolling contacts.

### 4.4.4 Static Force Analysis

We now relate the internal forces \( F \) in eq.(4.28) to the forces exerted by the manipulators on the object at the contact points. This is done by writing the static force/moment balance at the center of mass of the object and then transforming the vectors back to the contact points. Some of this analysis is similar to that in [95], with the difference being that in [95], the case of two robots grasping an object rigidly is considered. Here, we treat the case of multiple robots with rolling contacts between the end-effectors and the object.

Let \( P \) be the inertial force and moment vector at the center of mass of the object. This force is produced by the cooperating manipulators by applying forces to the object via their end-effectors at the respective points of contact. As seen in section
4.2.1, the type of contact between the end-effector and object determines what forces and/or moments are transmitted to the object. The two types of (point) contact that enable the bodies to roll without slipping relative to one another are point contact with friction and soft finger contact. The former allows the exertion of forces only (i.e. no moments are transmitted) while the latter additionally permits the exertion of a moment about the normal to the common tangent plane at the point of contact.

Let \( \mathbf{F}_p^i \) represent the forces exerted on the object by the \( i \)-th end-effector at the corresponding contact point (fig. 4.14). Then, assuming a frictional point contact, the force (and moment) contribution of each manipulator at the center of mass of the object can be computed as

\[
P^i = \begin{bmatrix} I_{3 \times 3} \\ -S \end{bmatrix} \mathbf{F}_p^i \triangleq W^i \mathbf{F}_p^i
\]  

(4.35)

where \( S \) is as defined in (4.13). In case of a soft finger contact, the matrix \( W^i \) will contain an additional column corresponding to the moment that can be exerted by the finger about the normal. \( \mathbf{F}_p^i \) will be four-dimensional in that case. The following analysis is applicable to either contact, the only difference being in the dimensions of the matrices.

The total object inertial force can be expressed in terms of the end-effector forces as

\[
P = \sum_{i=1}^{L} P^i = \begin{bmatrix} W^1 & \cdots & W^L \end{bmatrix} \begin{bmatrix} \mathbf{F}_p^1 \\ \vdots \\ \mathbf{F}_p^L \end{bmatrix} \triangleq W \mathbf{F}_p
\]  

(4.36)

where \( W = [(\mathbf{F}_p^1)^T \cdots (\mathbf{F}_p^L)^T]^T \). The matrix \( W \) is known in the literature as the Grasp Matrix or Grip Matrix. It is a function of the location of the contact points on the surface of the object. It thus incorporates the knowledge of the grasp geometry [64]. If the number of forces and moments that can be transmitted through each contact is \( d \), then the dimensions of \( W \) for the spatial case are \( 6 \times Ld \) (\( d = 3 \) for point contact with friction, \( d = 4 \) for soft finger contact).

Consider the static force/moment balance equation at the object center of mass

\[
P = \sum_{i=1}^{L} P^i = [I_{6 \times 6} \cdots I_{6 \times 6}] \begin{bmatrix} P^1 \\ \vdots \\ P^L \end{bmatrix} \triangleq W_c \mathbf{P}_c
\]  

(4.37)
$W_e \in \mathbb{R}^{6 \times 6_L}$ is the modified grasp matrix when the end-effector forces $F_p$ are expressed (as $P_e$) in the object frame of reference. Let $V_e = \text{Null}(W_e)$ ($V_e \in \mathbb{R}^{6\times6(L-1)}$). Then, the vector $P_e$ can be expressed in terms of the object inertial force and a null space component $P_N$ as

$$P_e = W_e^+ P + V_e P_N$$

$$= \begin{bmatrix} W_e^+ & V_e \end{bmatrix} \begin{bmatrix} P \\ P_N \end{bmatrix} \triangleq U \begin{bmatrix} P \\ P_N \end{bmatrix}$$

$$\Rightarrow \begin{bmatrix} P \\ P_N \end{bmatrix} = U^{-1} P_e \triangleq \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} P_e$$

Thus, the null space component can be determined as

$$P_N = X_2 P_e$$  \hspace{1cm} (4.38)

The null space component $P_N$ represents the components of the end-effector forces $P_e$ that get cancelled out at the object center of mass. Hence, from the discussion in the previous section, these forces are physically similar to the internal forces $F$ in the
dynamic equations (4.34), in that they represent the force (and moment) components at the object center of mass that do not contribute to the net inertial force on the object. In fact, the two vectors $\mathbf{F}$ and $\mathbf{P}_N$ are equal when the null space matrix $V_c$ is chosen to have the same (block) form as the (transposed) constraint matrix $H^T$ (with the $J^i$'s in $H$ replaced by identity matrices in $V_c$). Thus, when $H$ is as in eq.(4.27), $V_c$ is chosen as

$$
V_c = \begin{bmatrix}
I_{6\times6} & 0 & . & 0 \\
. & I_{6\times6} & . & . \\
. & . & I_{6\times6} & . \\
-I_{6\times6} & . & . & -I_{6\times6}
\end{bmatrix}
$$

With $V_c$ chosen in this manner, the internal force vector $\mathbf{F}$ can be expressed in terms of the vector of end-effector forces $\mathbf{F}_p$ as shown below. From (4.38), (4.37) and (4.35),

$$
\mathbf{F} = \mathbf{F}_N = X_2 \mathbf{P}_c = X_2 \begin{bmatrix}
W^1 & \\
& . \\
& \\
& W^L
\end{bmatrix} 
\begin{bmatrix}
\mathbf{F}_p^1 \\
. \\
. \\
\mathbf{F}_p^L
\end{bmatrix} = \tilde{W} \mathbf{F}_p
$$

The matrix $\tilde{W}$ has dimensions $6(L - 1) \times Ld$.

4.5 Exploiting redundancy in system equations

As stated earlier, the complete kinematic and dynamic model of the cooperating manipulator system with rolling contact between the end-effectors and the object is represented by the following equations [(4.15), (4.32), (4.33)] -

$$
J\ddot{q} = E\ddot{x}_o - \dot{J}\dot{q} \quad (4.40)
$$

$$
A_u\ddot{q} + B_u + H_u^T F = \tau_u \quad (4.41)
$$

$$
A_u\ddot{q} + B_u + H_u^T F = \tau_u = 0 \quad (4.42)
$$

When the system is controlled by independent joint control of the manipulators, eqs. (4.40) and (4.42) can be used to compute the joint accelerations and forces required to perform the desired task, which is specified in terms of the object trajectory
and the internal forces that need to be exerted on the object. One straightforward way to compute these vectors is to solve the following equation for $\ddot{q}$ and $E$.

$$
\begin{bmatrix}
J & 0 \\
A_u & H_u^T
\end{bmatrix}
\begin{bmatrix}
\ddot{q} \\
\dot{E}_{\dot{z}_o}
\end{bmatrix} =
\begin{bmatrix}
E\ddot{\ddot{z}}_o - \dot{J}\ddot{q} \\
-\dot{B}_u
\end{bmatrix}
$$

(4.43)

However, if each manipulator has (at least) as many actual joints as the corresponding task space dimension (i.e. 3 in the planar case and 6 in the spatial case), then the incorporation of the relative velocity parameters in the above equations lends the system some degrees of redundancy, thus making the set of equations in (4.43) underdetermined. In this case, these extra degrees of freedom can be exploited to plan the joint trajectories of the robots to perform some additional subtask while manipulating the object along the desired trajectory. In [48], the redundant degrees of freedom have been used to satisfy grasping conditions for a three-fingered hand. Recently, in [18], the inverse kinematic problem for two cooperating manipulators, with one manipulator maintaining rolling contact with the object, was solved by incorporating the rolling degrees of freedom as additional revolute joints in the kinematic formulation. However, these works do not include any dynamic analysis of the manipulators.

Below, we show how the desired joint accelerations and force can be computed to satisfy an additional subtask such as collision avoidance. Once the joint accelerations and force are computed, hybrid position and force control techniques may be utilized to control the robotic manipulators individually.

We assume that the dimension of the joint acceleration vector $\ddot{q}$ is greater than the dimension of the stacked task space vector $E\ddot{\ddot{z}}_o$. This will be the case whenever the cooperating robot manipulators have enough joints to be able to place the end-effector at any position and orientation in their task space. In this situation, the augmentation of the joint variables with the relative velocity variables will make the kinematic equation (4.40) underdetermined. Thus, for example, when 3 cooperating planar manipulators with 3 joints each handle a common object with rolling contacts, the dimension of $\ddot{q}$ will be 12 while that of $E\ddot{\ddot{z}}_o$ will be 9. Thus, the joint space will have 3 extra degrees of freedom.

As we have seen in chapter 3, when a set of equations is underdetermined, an additional subtask can be specified by adding a null space term to the solution. Thus, in this case, given the desired object trajectory $\ddot{z}_{odes}$, the desired joint accelerations
can be computed as

$$\ddot{q}_{des} = J^+ \left( E_{\ddot{x}_{obs}} - \ddot{J} \ddot{q} \right) + (I - J^+ J) \xi_1 \quad (4.44)$$

The matrix \((I - J^+ J)\) projects any vector onto the null space of \(J\). \(\xi_1\) can be any vector which is chosen to satisfy a secondary performance criterion. For instance, as in the case of redundant manipulator kinematics, suppose there is an obstacle in the path of one of the manipulators. Or, there may be a possibility of one of the manipulator links colliding into the object itself. In such a case, a suitable point on one of the manipulator links, say \(x_{obs}\) can be assigned a trajectory such that the imminent collision is avoided. Since \(x_{obs}\) will be a function of the joint variables, it can be expressed in terms of the joint accelerations as

$$\ddot{x}_{obs} = J_{obs} \ddot{q} + \dot{J}_{obs} \dot{q} \quad (4.45)$$

Then, the vector \(\xi_1\) for this subtask can be computed as

$$\xi_1 = (I - J^+ J) \left( J_{obs} \left[ \ddot{x}_{obs} - J_{obs} J^+ (E_{\ddot{x}_{obs}} - \ddot{J} \ddot{q}) - \dot{J}_{obs} \dot{q} \right] \right)$$

Substituting this expression in (4.44) and simplifying [59] yields,

$$\ddot{q}_{des} = J^+ \left( E_{\ddot{x}_{obs}} - \ddot{J} \ddot{q} \right) + \left[ J_{obs} (I - J^+ J) \right]^+ \left[ \ddot{x}_{obs} - J_{obs} J^+ (E_{\ddot{x}_{obs}} - \ddot{J} \ddot{q}) - \dot{J}_{obs} \dot{q} \right]$$

Having computed the desired joint acceleration, the equation (4.42) which represents the rolling contact constraint can be used to compute the desired internal force vector. Note that in the example of 3 planar cooperating manipulators cited above, \(F\) is 6-dimensional while \(H_u^T \in \mathbb{R}^{3 \times 6}\). Hence, an underdetermined system of equations is formed by

$$H_u^T F = -A_u \ddot{q} - B_u$$

which can be solved for the desired force as

$$F_{des} = \left( H_u^T \right)^+ (A_u \ddot{q}_{des} - B_u) + \left( I - \left( H_u^T \right)^+ H_u^T \right) \xi_2 \quad (4.46)$$

Once again, the vector \(\xi_2\) gets projected onto the null space of the matrix \(H_u^T\). Hence, \(\xi_2\) can be chosen such that the internal force vector \(F_{des}\) satisfies a grasping or squeezing criterion. Several techniques have been studied in the literature for computing the vector \(\xi_2\) to ensure that a proper grasp is maintained (e.g. see [1], [77], [88], [98]).
Once the desired acceleration and force vectors $\ddot{\mathbf{q}}_{\text{des}}$ have been determined, hybrid position/force control laws can be designed that compute torque inputs $\boldsymbol{\tau}$ to the joints of the manipulators ([46], [65], [86], [109]), such that the specified joint accelerations and forces are achieved. This, in turn, will imply not only that the object is moved and grasped in the desired fashion, but also that the subtask specified by (4.45) is performed simultaneously. An example of this technique used to avoid a collision between a manipulator link and the object is shown in section 4.7.

### 4.6 Control Algorithm

In the previous section, we discussed one control method wherein the joints of the manipulators are controlled directly and independently. The desired joint trajectories and the end-effector forces are computed from eqs.(4.44) and (4.46) respectively, given the desired task specifications. Such a control law will track the joint trajectories and will use errors in the joint space as feedback terms. However, this control scheme does not provide direct control over the trajectory of the object or the forces experienced by the object. Hence, if there are modeling/parameter errors in the mapping from the joint parameters to the object parameters (which there usually will be), then correctly tracking the desired joint trajectory will not imply that the object is manipulated as desired. Hence, in order to ensure that the object follows the specified trajectory and is grasped as desired, we need a control law that uses the object parameters directly as feedback. The formulation of such a control law is the objective of this section. Using the kinematic and dynamic equations derived in the previous sections, we present here, a control algorithm that enables simultaneous control of not only, the object trajectory and internal forces, but also the contact variables (either $\mathbf{u}_f$ or $\mathbf{u}_c$).

In order to facilitate the derivation of the control law, we first express the internal force $\boldsymbol{F}$ in terms of the object dynamics. The object dynamics can be written as [20]

$$
\begin{bmatrix}
m_o I & 0 \\
0 & I_b
\end{bmatrix}
\begin{bmatrix}
\dot{\mathbf{q}}_o \\
\dot{\omega}_o
\end{bmatrix}
+ \begin{bmatrix}
0 \\
\omega_o \times I_b \omega_o
\end{bmatrix}
= \begin{bmatrix}
\mathbf{f}_o \\
\mathbf{\tau}_o
\end{bmatrix}
= \mathbf{P}
$$

where $m_o I \in \mathbb{R}^{3 \times 3}$ is the diagonal object mass matrix, $I_b = R_o I_o R_o^T$ is the object inertia matrix expressed in the base frame, with $I_o$ being the constant object inertia matrix, expressed in the object frame. $\mathbf{f}_o$ and $\mathbf{\tau}_o$ are the force and moment.
components of the vector \( P \) applied at the object center of mass. Note that

\[
\begin{bmatrix}
\dot{\mathbf{x}}_o \\
\dot{\mathbf{y}}_o \\
\dot{\mathbf{z}}_o
\end{bmatrix} = \begin{bmatrix}
\mathbf{v}_o \\
\mathbf{u}_o \\
\mathbf{w}_o
\end{bmatrix}; \quad \begin{bmatrix}
\ddot{\mathbf{x}}_o \\
\ddot{\mathbf{y}}_o \\
\ddot{\mathbf{z}}_o
\end{bmatrix} = \begin{bmatrix}
\dot{\mathbf{v}}_o \\
\dot{\mathbf{u}}_o \\
\dot{\mathbf{w}}_o
\end{bmatrix}
\]

In order to determine the error in the orientation of the object at a given time, it is necessary to parametrize the orientation locally using a vector \( \gamma_o \in \mathbb{R}^3 \). For instance, the roll, pitch and yaw angles of the object may be used to obtain this parametrization. The angular velocity of the object can then be expressed in terms of the parametrization using a linear transformation \( Q(\gamma_o) \) as [106]

\[
\omega_o = Q(\gamma_o)\dot{\gamma}_o
\]

The orientation error can now be expressed in terms of the vectors \( \gamma_o \) and \( \dot{\gamma}_o \). Thus, we now express the object velocity and acceleration as

\[
\begin{bmatrix}
\dot{\mathbf{x}}_o \\
\dot{\mathbf{y}}_o \\
\dot{\mathbf{z}}_o
\end{bmatrix} = \begin{bmatrix}
\mathbf{v}_o \\
\dot{\gamma}_o \\
\dot{\gamma}_o
\end{bmatrix}; \quad \begin{bmatrix}
\ddot{\mathbf{x}}_o \\
\ddot{\mathbf{y}}_o \\
\ddot{\mathbf{z}}_o
\end{bmatrix} = \begin{bmatrix}
\dot{\mathbf{v}}_o \\
\dot{\gamma}_o \\
\dot{\gamma}_o
\end{bmatrix}
\]

and the object dynamics can be compactly written as

\[
P = M_o \ddot{\mathbf{x}}_o + \mathcal{N}_o
\]

where

\[
M_o = \begin{bmatrix}
m_o I & 0 \\
0 & I_b
\end{bmatrix} \begin{bmatrix}
I & 0 \\
0 & Q
\end{bmatrix}; \quad \mathcal{N}_o = \begin{bmatrix}
0 \\
(Q\dot{\gamma}_o) \times I_b \dot{\gamma}_o + I_b \ddot{\gamma}_o
\end{bmatrix}
\]

Also, from (4.36), \( E_p \) can be expressed as

\[
E_p = W^t P + V^t f
\]

where \( WV = 0 \). Thus, from (4.39), (4.47) and (4.48),

\[
E = \bar{W} \left( W^t M_o \ddot{\mathbf{x}}_o + W^t \mathcal{N}_o + V^t f \right)
\]

Substituting this expression for \( E \) in the dynamic equations (4.34),

\[
A \ddot{\mathbf{q}} + H^t \bar{W} W^t M_o \ddot{\mathbf{x}}_o + \left( B + H^t \bar{W} W^t \mathcal{N}_o \right) + H^t \bar{W} V^t f = R_{\mathcal{E}_a}
\]

Substituting \( A_o = H^t \bar{W} W^t M_o \) and \( B_o = B + H^t \bar{W} W^t \mathcal{N}_o \) and multiplying by \( A^{-1} \), we get

\[
\ddot{\mathbf{q}} + A^{-1} A_o \ddot{\mathbf{x}}_o + A^{-1} B_o + A^{-1} H^t \bar{W} V^t f = A^{-1} R_{\mathcal{E}_a}
\]

(4.49)
The first-order kinematic contact equations [69] relate the relative angular velocity \( \omega_r \) between the \( i \)-th end-effector and the object to the contact velocities \( \dot{u}_o^i \) and \( \ddot{u}_o^i \) on their respective surfaces. Hence, by inserting the appropriate zero columns corresponding to \( \dot{q}_i \) and stacking the equations for all the contacts, we can write the relation between \( \ddot{u}_o \) and \( \ddot{q} \) as

\[
\ddot{u}_o = J_u \ddot{q}
\]  

(4.50)

where the elements of \( J_u \) are functions of \( u_o \) and \( u_f \). We then obtain the second-order relation by differentiating (4.50) (or using the second-order kinematics derived in [89])

\[
\dddot{u}_o = J_u \dddot{q} + \dot{J}_u \ddot{q} + \ddot{J}_u \dot{q} \triangleq J_u \dddot{q} + \dot{N}_u \quad \text{(4.51)}
\]

(In the control algorithm, we can choose to control either the object contact variables \( u_o \) or the end-effector contact variables \( u_f \). We have chosen \( u_o \) in this derivation but equations similar to (4.50) and (4.51) can be written for \( u_f \) and \( \ddot{u}_f \).

Also, differentiating the constraint (4.27),

\[
H \ddot{q} + \dot{H} \dot{q} = 0 \quad \Rightarrow \quad H \ddot{q} = -\dot{H} \dot{q} \triangleq -N_c
\]  

(4.52)

Let

\[
L \triangleq \begin{bmatrix} J_u \\ H \end{bmatrix}; \quad N_1 \triangleq \begin{bmatrix} N_u \\ N_c \end{bmatrix}
\]  

(4.53)

Hence, from eqs. (4.51) and (4.52),

\[
L \dddot{q} = \begin{bmatrix} J_u \\ H \end{bmatrix} \dddot{q} = \begin{bmatrix} \dddot{u}_o \\ 0 \end{bmatrix} - \begin{bmatrix} N_u \\ N_c \end{bmatrix} = \begin{bmatrix} \dddot{u}_o \\ 0 \end{bmatrix} - N_1
\]  

(4.54)

Pre-multiplying (4.49) by \( L \) and substituting from (4.54),

\[
\begin{bmatrix} \dddot{u}_o \\ 0 \end{bmatrix} + LA^{-1}A_o \dddot{x}_o + LA^{-1}B_o - N_1 + LA^{-1}H^T \overline{W} V \ddot{f} = LA^{-1}R \tau_a
\]

\[
\Rightarrow \begin{bmatrix} I \\ 0 \end{bmatrix} \dddot{x}_o + (LA^{-1}A_o) \dddot{x}_o + (LA^{-1}B_o - N_1) + (LA^{-1}H^T \overline{W} V \ddot{f} \ddot{f}) = (LA^{-1}R) \tau_a
\]

or, grouping the terms together,

\[
Y \begin{bmatrix} \dddot{u}_o \\ \dddot{x}_o \\ \ddot{f} \end{bmatrix} + B_1 = M \tau_a
\]  

(4.55)
where

\[ Y \triangleq \begin{bmatrix} Y_1 & Y_2 & Y_3 \end{bmatrix} \]

\[ Y_1 \triangleq \begin{bmatrix} I \\ 0 \end{bmatrix} \quad ; \quad Y_2 \triangleq LA^{-1}A_o \quad ; \quad Y_3 \triangleq LA^{-1}HTWV \]

\[ B_1 \triangleq LA^{-1}B_o - N_1 \quad ; \quad M \triangleq LA^{-1}R \]

Note that all the terms in \( B_1 \) are functions of position and velocity variables.

### 4.6.1 Dimensional Analysis

In order to be able to control all the components of \( \ddot{q}_o, \dddot{q}_o \) and \( f \), the number of actuators present in the system must be greater than or equal to the sum of the dimensions of the three variables. Thus, for instance, in the case of three planar mechanisms manipulating a common object, we have \( \ddot{u}_o \in \mathbb{R}^3, \dddot{u}_o \in \mathbb{R}^3 \) and \( f \in \mathbb{R}^3 \). Hence, each manipulator should have at least 3 degrees of freedom.

In the spatial case, the contact variable \( \ddot{u}_c \) at each contact is two-dimensional. Thus, for example, with a four-fingered hand, \( \ddot{u}_o \in \mathbb{R}^6, \dddot{u}_o \in \mathbb{R}^6 \) and \( f \in \mathbb{R}^6 \). Therefore, each finger would have to have 5 actuators for all the components to be controlled. However, the hands currently in existence, usually have 3 or 4 degrees of freedom per finger. Hence, we may choose to control the contact variables for two of the four fingers. This will be useful, for example, when the manipulation is primarily two-fingered, but to re-position the two fingers accurately to a new grasp, two more fingers are employed. On the other hand, if the mechanisms involved are robotic arms with 6 joints each, then in fact, the number of actuators will be more than the number of controlled variables, and the system will be redundant. In this case, the redundancy can be exploited to satisfy some optimization criterion.

Below, we show how a control law can be formulated for the case when the number of actuators is equal to the number of controlled variables. The extension to the other cases is straightforward (see appendix C).

### 4.6.2 Control Law

The control objective is to make the object and the contact points on the object track given desired trajectories \( x_{od}(t) \) and \( u_{od}(t) \) respectively, while maintaining a desired internal force specified by \( f_{od}(t) \). Thus, the control algorithm ensures not only that the object is manipulated as desired, but also that the grasp locations on the object
change as desired. Note that in the spatial case, the system is nonholonomic and hence cannot be stabilized to an isolated equilibrium solution using smooth feedback [6]. Hence, the control law has to be utilized for stabilization to trajectories or equilibrium manifolds.

Define the following error vectors -

\[
\varepsilon_x = \xi_o - \xi_{od} \quad ; \quad \varepsilon_u = u_o - u_{od} \quad ; \quad \varepsilon_f = \mathbf{f} - \mathbf{f}_d
\]

In the following discussion, we consider the case when the number of actuators is equal to the number of controlled variables. Hence, without loss of generality we can assume the matrices \( Y \) and \( M \) in eq.(4.55) to be square, since the number of rows of \( H \) included in the matrix \( L \) can be adjusted to make this so.

**Theorem 4.2** If (a) the grasp locations are distinct and (b) the manipulators do not pass through a singular configuration, then the control law specified by (4.56) realizes the desired object trajectory, contact trajectories and the internal forces.

\[
\tau_a = M^{-1} \left\{ \begin{bmatrix} \ddot{u}_{od} - K_{vu} \dot{u}_u - K_{pu} \varepsilon_u \\ \ddot{\xi}_o - K_{vz} \dot{\xi}_x - K_{pz} \varepsilon_x \\ -K \int \varepsilon_f \end{bmatrix} + B_1 \right\}
\]

(4.56)

**Proof** Substitute the control law (4.56) in eq.(4.55) to obtain

\[
Y \begin{bmatrix} \dot{\varepsilon}_u + K_{vu} \dot{\varepsilon}_u + K_{pu} \varepsilon_u \\ \dot{\varepsilon}_x + K_{vz} \dot{\varepsilon}_x + K_{pz} \varepsilon_x \\ \varepsilon_f + K_i \int \varepsilon_f \end{bmatrix} = 0
\]

\( Y \) is square and by the assumptions (a) and (b), it is non-singular, hence it is invertible. Thus,

\[
\begin{align*}
\dot{\varepsilon}_u + K_{vu} \dot{\varepsilon}_u + K_{pu} \varepsilon_u &= 0 \\
\dot{\varepsilon}_x + K_{vz} \dot{\varepsilon}_x + K_{pz} \varepsilon_x &= 0 \\
\varepsilon_f + K_i \int \varepsilon_f &= 0
\end{align*}
\]

Hence, with appropriate selection of \( K_{vu}, K_{pu}, K_{vz}, K_{pz} \) and \( K_i \), the errors \( \varepsilon_u, \varepsilon_x \) and \( \varepsilon_f \) go to zero. \( \square \)
4.7 Simulations

Here, we demonstrate the algorithms described in this chapter by means of simulations.

4.7.1 Kinematic Simulations

Example 1:

We first consider a two-arm planar manipulator system, with each manipulator having 3 revolute joints and unit link lengths. The circular object has unit radius and rolling contact is maintained between the last link of each manipulator and the object. Simulations were carried out on this system to show how the redundancy in the kinematic and dynamic model can be used to plan the contact and joint trajectories such that a secondary criterion of collision avoidance is satisfied.

In fig. 4.15, the joint trajectories are computed using eq.(4.43). The object has to be moved along a straight line with velocity of 0.6 units/sec in the negative x-direction with zero acceleration. It is seen that the second link of the manipulator on the left collides with the object, when this trajectory is used.

However, by specifying a secondary collision avoidance criterion, in the form of a velocity in the positive y-direction for a point fixed on joint 2 of the left manipulator, the collision can be avoided as seen in fig. 4.16.

Example 2:

Figs. 4.17 and 4.18 demonstrate another instance of the same. The task is the same, but the initial configuration of the manipulators is different from the example above. Once again, by incorporating the additional subtask, the top manipulator is able to avoid a collision of the object with one of its links.

4.7.2 Control algorithm simulations

For demonstrating the control algorithm in section 4.6.2, we consider a three-fingered planar manipulation system. Once again, each manipulator has three revolute joints, with the third link of each being in contact with the round object of unit radius. Errors of the order of 10-15% were assumed in the controller parameters. The details of the simulation are enumerated in appendix D.

Example 3:

In the first example, the task consists of shifting the object to specified x and y coordinates, while rotating it back and forth. This kind of task may be required
Figure 4.15 Example 1: System evolution with no subtask specified
Figure 4.16  Example 1: System evolution with collision avoidance subtask
Figure 4.17  Example 2: System evolution with no subtask specified
Figure 4.18  Example 2: System evolution with collision avoidance subtask
Figure 4.19  Example 3: Evolution of system
Figure 4.20 Example 3: Object trajectories
Figure 4.21  Example 3: Contact trajectories
Figure 4.22  Example 3: Internal force components
say, when the manipulators need to move a stirring rod in a trough. The desired object trajectory is thus specified as constant $x$ and $y$ coordinates and a sinusoidal orientation $\phi$. The contact points are specified to move with constant velocity. The evolution of the system trajectory is shown in Figs. 4.19(a)-(f). The convergence of the object and contact trajectories and the internal forces is shown in Figs. 4.20-4.22.

Example 4:
In the second example, the desired object trajectory again consists of a small shift in its $x$ and $y$ coordinates but the orientation is now made to change linearly. This kind of task will be required when a particular object, such as a component to be assembled, is to be rotated through a certain angle. The desired contact trajectories are specified to be sinusoidal with equal amplitude and frequency. This has several advantages. It ensures that the contact on the link does not run off the end of the link. Also, the contact point for each manipulator remains within its reachable workspace. In addition, the grasp locations on the object do not get too close or too far apart from each other. The internal force vector is specified to ensure that the grasp has a squeezing effect on the object. Figs. 4.23(a)-(f) depict the evolution of the system. Figs. 4.24-4.26 show the errors in tracking the object trajectory, contact trajectories and the internal forces, converging to zero. (Error plots for the same results have been plotted in [34]).

4.8 Conclusions
We have developed a new dynamic formulation for a cooperating manipulator system with rolling contacts between the end-effectors and the object. This formulation models the rolling degrees of freedom as unactuated joints of the manipulator. One of the advantages of this formulation is that it results in a kinematically redundant set of equations, thus enabling the satisfaction of subtask criteria in planning the motion of the joints and the contacts. The dynamic equations developed with this concept (eq. (4.32)) relate the actuator torques to the joint accelerations (including the relative accelerations at the contacts) and the internal forces at the center of the object. They also yield dynamic constraints arising from the presence of the rolling contacts (eq. (4.33)). The main advantage of this new formulation is that it can be used for the specification of a general control law that enables simultaneous tracking of the object trajectory, the internal forces in the object and the contact trajectories. Although we have shown simulations of a planar system for ease of presentation, the
Figure 4.23  Example 4: Evolution of system
Figure 4.24  Example 4: Object trajectories
Figure 4.25  Example 4: Contact trajectories
Figure 4.26  Example 4: Internal force components
technique and equations are fully general and can be applied to a spatial system as well.
Chapter 5

Summary and Future Work

What a piece of work is a man!
William Shakespeare, Hamlet

The development of new algorithms for the kinematic and dynamic control of robotic manipulator systems has been the focus of this dissertation. We have dealt with both single-arm and multiple cooperating manipulator systems. The techniques described in this work can be applied to systems such as redundant manipulators and artificial multifingered hands. In this chapter, we summarize the various results presented in the thesis and discuss avenues for future work.

In our earlier work [25], the damped least squares method which is a velocity inverse kinematic method (i.e. a technique for computing joint velocities for specified end-effector velocities) yielding bounded joint velocities in all regions of the manipulator workspace was studied. The trust region algorithms proposed for overdetermined systems were adapted to the underdetermined inverse kinematic equations, to obtain a method for optimally damping the joint velocities such that the error in the task space is minimized. We also integrated this method with the performance of an additional subtask such as obstacle avoidance, for redundant manipulators. In the course of these studies, it was noticed that the damped least squares technique fails to converge when the desired end-effector position is outside the reachable workspace of the manipulator. In this case, no joint configuration can place the end-effector at the desired position and the use of the damped least-squares method can give rise to oscillations near the workspace boundary.

In chapter 2 of this work, we propose the use of an adaptive algorithm which switches from the damped least-squares method to a second-order model of the kinematic function, when the damped least-squares method is unable to converge. The second-order model is able to converge to an exact singularity and hence yields good results, when employed close to a minimizer. We propose a switching strategy which ensures that the second-order model is employed only when the end-effector position is unreachable and convergence to an external singularity is desired.
The methods described above are based on optimizing objective functions which are formulated using the Euclidean norm or the two-norm, which is the norm universally used in inverse kinematic computations. However, in working with these algorithms, it became clear that when a choice of inverse kinematic solutions is available (i.e. in the case of redundant manipulators), optimizing the Euclidean norm does not always give a good measure of performance. In particular, when low magnitudes of joint velocities are desired, a more appropriate norm to be minimized is the infinity norm. In chapter 3 of this thesis, we investigate several algorithms for the computation of joint velocity vectors for redundant manipulators, with a minimum infinity norm, a measure of performance that has hitherto been unused in the robotics literature. We also look at adapting these algorithms to optimize other sub-task performance criteria such as obstacle avoidance or joint limit avoidance for such manipulators. We present several examples comparing the use of this norm in inverse kinematic computations to that of the Euclidean norm, which demonstrate the fact that better control over individual joint velocities is achieved by utilizing the infinity norm.

The work in this chapter, while presenting new techniques for inverse kinematics computations for redundant manipulators, opens a whole new arena of robotic research in optimizing non-Euclidean norm-based performance criteria. As discussed in section 3.12, uniqueness and continuity of such solutions, when applied to inverse kinematics of robotic manipulators, needs to be investigated. Another interesting question that arises is whether some measure can be determined, based on say, the manipulator Jacobian, which can indicate the extent of the difference in optimizing criteria with different norms in different regions of the manipulator workspace. We hope that the algorithms and examples presented in chapter 3 will spawn investigations of new algorithms for inverse kinematics computations using non-Euclidean norms.

The second half of this thesis, contained in chapter 4, deals with the control of multiple manipulator systems formed by several manipulators cooperatively handling a common object. Specifically, we investigate dynamic modeling and control issues for the case when there exists rolling contact between the end-effectors and the object. This type of study should be especially useful in devising control methods for artificial multifingered hands, which are becoming increasingly popular as versatile end-effectors for robotic manipulators. The results can also be applied to a system in which several manipulators cooperatively handle an object with an enveloping
grasp i.e. a grasp where the end-effector is distributed over, say, the end link of the manipulator.

The core contribution of this chapter is a new dynamic formulation and control algorithm for such a cooperating system. The dynamic formulation is derived using a graph-theoretic framework for expressing the couplings between the links of the manipulators and the manipulators and the object. This derivation is based on the concept of modeling the rolling contacts as unactuated joints of the respective manipulators. This new dynamic formulation enables the design of a general control law that can be used to explicitly control the trajectory of the object, the internal forces generated within the object and the trajectories of the contact points on the object/end-effectors. To the best of our knowledge, no such general algorithm for controlling all three types of variables has been proposed previously. In addition, we also demonstrate how the redundancy resulting from incorporating the relative velocity between the end-effectors and the object into the kinematic and dynamic equations, can be utilized to perform an additional subtask such as collision avoidance.

An immediate extension of this work is to determine if the same concept of modeling the relative motion between the end-effectors and the object can be extended to the case of sliding motions and then, to a combination of roll-slide motions. The unified dynamic formulation derived in this work can be used to design other control laws as well. For instance, in applications where the uncertainty in the manipulator models is large, the design of control laws which incorporate adaptive and robustness issues needs to be investigated. On the practical side, in order to promote the implementation of these control algorithms, it will be necessary to augment the robotic system with accessories such as camera vision system and tactile sensors, to obtain real-time data regarding the actual positions of the manipulators and the object. Sensor fusion techniques which can combine this real-time data to determine differential geometric and other information required for such systems and, sophisticated hardware for processing the large amount of information in real time will be needed to make these algorithms practically feasible. Although research efforts in all these areas are prevalent currently, there is also a need for efforts to combine these with the results obtained in the analytical and control areas.

In conclusion, this dissertation has contributed new techniques in the following areas of research in robotic systems -

- Inverse kinematic computations of robotic manipulators
- An adaptive algorithm which is insensitive to the reachability of the desired end-effector position
- Introduction of minimum-effort inverse kinematics for redundant manipulators

- Dynamics and control of multiple cooperating manipulator systems with rolling contact.
  - A new unified dynamic formulation for such systems
  - Trajectory planning method incorporating subtask performance
  - A general control algorithm that explicitly controls the object trajectory, internal forces and contact trajectories
Appendix A

Proofs of Lemmas 2.1, 2.2 and 3.1

A.1 Lemmas 2.1 and 2.2

![Schematic representation](image)

Figure A.1 Schematic representation

**Lemma A.1** For any point \( \tilde{x} \neq \tilde{x}^* \) on \( S \) i.e. satisfying (2.18),
\[
\|\tilde{x}^d - \tilde{x}\| < \|\tilde{x}^d - \tilde{x}^*\| = \|R^*\|
\]

**Proof** Substituting the value of \( \tilde{x} \) from (2.17) in eq.(2.18),
\[
\begin{align*}
\|\tilde{x}^* + \alpha(\tilde{x}^d - \tilde{x}^*) - \tilde{x}\| &= \alpha\|R^*\| \\
\|\alpha \tilde{x}^d + (1 - \alpha)\tilde{x}^* - \tilde{x}\| &= \alpha\|R^*\| \\
\|\alpha \tilde{x}^d + (1 - \alpha)(\tilde{x}^d - R^*) - \tilde{x}\| &= \alpha\|R^*\| \\
\|(\tilde{x}^d - \tilde{x}) - (1 - \alpha)R^*\| &= \alpha\|R^*\|
\end{align*}
\]
(A.1)

(In A.1, we have used \( \tilde{x}^d - \tilde{x}^* = R^* \Rightarrow \tilde{x}^* = \tilde{x}^d - R^* \). Thus, using \( \|a-b\| \geq \|a\| - \|b\| \)
in A.2, we obtain
\[
\begin{align*}
\|\tilde{x}^d - \tilde{x}\| - (1 - \alpha)\|R^*\| &\leq \alpha\|R^*\| \\
\|\tilde{x}^d - \tilde{x}\| &\leq \|R^*\|
\end{align*}
\]
(A.3)
Equality will hold in the above expression if and only if, the two vectors are scalar multiples of each other i.e. for some non-zero $\beta \in \mathbb{R}$,

$$\bar{x}^d - \bar{x} = \beta \bar{R}^* = \beta (\bar{x}^d - \bar{x}^*)$$

Thus, for $\|\bar{x}^d - \bar{x}\| = \|\bar{R}^*\|$, we must have $\beta = 1$ i.e. $\bar{x} = \bar{x}^*$. Hence, for all $\bar{x} \neq \bar{x}^*$,

$$\|\bar{x}^d - \bar{x}\| < \|\bar{R}^*\| = \|\bar{x}^d - \bar{x}^*\|$$

(A.4)

\[ \square \]

Lemma A.2

$\hat{x}(s^*)^T \bar{R}^* = 0$

Proof From eq.(2.18),

$$[\hat{x} - \bar{x}(s)]^T[\hat{x} - \bar{x}(s)] = \alpha^2 \|\bar{R}^*\|^2$$

Hence, differentiating with respect to $s$, we get

$$2[\hat{x}(s)]^T[\hat{x} - \bar{x}(s)] = 0$$

(A.5)

At $s = s^*$, $\bar{x}(s) = \bar{x}(s^*) = \bar{x}^*$. Thus,

$$\hat{x} - \bar{x}(s^*) = \hat{x} - \bar{x}^*$$

$$\quad = \bar{x}^* + \alpha(\bar{x}^d - \bar{x}^*) - \bar{x}^*$$

$$\quad = \alpha(\bar{x}^d - \bar{x}^*) = \alpha \bar{R}^*$$

(A.6)

Thus, from eqs.(A.5) and (A.6), at $s = s^*$,

$$[\hat{x}(s^*)]^T \bar{R}^* = 0$$

\[ \square \]

A.2 Alignment Lemma 3.1

Lemma A.3 A vector $z \in l^n_\infty$ is aligned with a given nonzero vector $w \in l^n_1$, if and only if $z$ is of the form

$$z_i = \alpha \cdot \text{sgn}[w_i] \quad \text{if} \quad w_i \neq 0$$

$$= \alpha_i \quad \text{if} \quad w_i = 0$$

(A.7)

where $\alpha \geq 0$ and $|\alpha_i| \leq \alpha$. 

Proof

(In the following, WLOG, we assume that $w$ has all non-zero components. Otherwise, the number of indices in the summations reduces accordingly)

One way is trivial i.e. if $z$ is as in eq. (A.7), then $z$ is aligned with $w$. Thus,

$$w^T z = \sum_{i=1}^{n} w_i z_i = \alpha \sum_{i=1}^{n} w_i \text{sgn}(w_i) = \alpha \sum_{i=1}^{n} |w_i| = \|z\|_{\infty} \|w\|_1$$

To show the converse, consider a vector $z \in \ell^n_1$, with $\infty$-norm equal to $\alpha$, that is aligned with the given $w$. Then, by the alignment condition,

$$\|z\|_{\infty} \|w\|_1 = \sum_{i=1}^{n} z_i w_i$$

$$\Rightarrow \alpha \sum_{i=1}^{n} |w_i| = \sum_{i=1}^{n} z_i (\text{sgn}(w_i) |w_i|)$$

$$\Rightarrow \sum_{i=1}^{n} \left( \alpha - z_i \text{sgn}(w_i) \right) |w_i| = 0 \quad (A.8)$$

However,

$$\|z\|_{\infty} = \alpha \Rightarrow \alpha \geq z_i \text{sgn}(w_i), \forall i \Rightarrow \alpha - z_i \text{sgn}(w_i) \geq 0, \forall i$$

Hence, for the equality (A.8) to be satisfied with non-zero $w_i$,

$$\alpha = z_i \text{sgn}(w_i), \forall i \Rightarrow z_i = \alpha \text{sgn}(w_i), \forall i$$
Appendix B

Proofs of Differential Geometric Lemmas

Lemma B.1 If \((f, U)\) is an orthogonal coordinate system for a coordinate patch \(S_s\) of \(S\), then for any \(s \in S_s\), the curvature form \(\mathcal{K}\) defined in (4.2), is a matrix representation of the shape operator \(S_s\) at \(s\) in terms of the basis vectors \(x(u, v)\) and \(y(u, v)\).

Proof Since the shape operator \(S\) on a surface \(M\) maps the tangent plane of \(M\) \((T_p(M))\) to itself, for any vector \(v\) in \(T_p(M)\) at the point \(p\) of \(M\), the vector \(S(v)\) can be expressed in terms of the frame field \(E_1, E_2, E_3\) at the point \(p\) as

\[
S(v) = (S(v) \cdot E_1) E_1 + (S(v) \cdot E_2) E_2
\]  
(B.1)

Suppose \(f(u, v) : U \rightarrow M\) forms an orthogonal coordinate patch on the surface \(M\). Then \(f_u\) and \(f_v\) are tangent vectors to \(M\) at each point and the frame field at each point can be computed as

\[
E_1 = \frac{f_u}{\|f_u\|} \quad ; \quad E_2 = \frac{f_v}{\|f_v\|} \quad ; \quad E_3 = \frac{f_u \times f_v}{\|f_u \times f_v\|}
\]

The vector \(E_3\) at each point is the unit normal vector \(N\) to the surface \(M\) at that point.

The following real-valued functions are defined on \(U\).

\[
E = f_u \cdot f_u \quad ; \quad F = f_u \cdot f_v \quad ; \quad G = f_v \cdot f_v
\]  
(B.2)

\[
l = S(f_u) \cdot f_u \quad ; m = S(f_u) \cdot f_v \quad ; n = S(f_u) \cdot f_v
\]  
(B.3)

Using these functions and (B.1), the vectors \(S(E_1)\) and \(S(E_2)\) can be expressed as

\[
S(E_1) = \left(\frac{l}{E}\right) E_1 + \left(\frac{m}{\sqrt{EG}}\right) E_2
\]  
(B.4)

\[
S(E_2) = \left(\frac{m}{\sqrt{EG}}\right) E_1 + \left(\frac{n}{G}\right) E_2
\]  
(B.5)
Hence, the shape operator transformation at any point can be represented by the matrix

\[
S = \begin{bmatrix}
\frac{l}{\sqrt{EG}} & \frac{m}{\sqrt{EG}} \\
\frac{m}{\sqrt{EG}} & \frac{n}{G}
\end{bmatrix}
\]  \hspace{1cm} (B.6)

If we write the definition of the curvature form in eq.(4.2) in terms of the functions defined above, we obtain,

\[
\mathcal{K} = \begin{bmatrix}
\left( \frac{f_u}{\sqrt{E}} \right) \cdot \left( \frac{N_u}{\sqrt{E}} \right) & \left( \frac{f_u}{\sqrt{E}} \right) \cdot \left( \frac{N_v}{\sqrt{G}} \right) \\
\left( \frac{f_v}{\sqrt{G}} \right) \cdot \left( \frac{N_u}{\sqrt{E}} \right) & \left( \frac{f_v}{\sqrt{G}} \right) \cdot \left( \frac{N_v}{\sqrt{G}} \right)
\end{bmatrix}
\]  \hspace{1cm} (B.7)

Since \( f_u, f_v \) denote the velocity vectors of the \( u \)-parameter and \( v \)-parameter curves respectively, on the surface \( M \), using the definition of the covariant derivative, it can be shown that

\[
N_u = S(f_u) \quad ; \quad N_v = S(f_v)
\]

Hence,

\[
l = S(f_u) \cdot f_u = N_u \cdot f_u
\]

\[
m = S(f_u) \cdot f_v = N_u \cdot f_v \quad \text{and} \quad m = S(f_v) \cdot f_u = N_v \cdot f_u
\]

\[
n = S(f_v) \cdot f_v = N_v \cdot f_v
\]

Thus, the curvature form can be expressed in terms of the real-valued functions in (B.2) and (B.3) as

\[
\mathcal{K} = \begin{bmatrix}
\frac{l}{\sqrt{EG}} & \frac{m}{\sqrt{EG}} \\
\frac{m}{\sqrt{EG}} & \frac{n}{G}
\end{bmatrix}
\]  \hspace{1cm} (B.8)

From eqs.(B.6) and (B.8), \( S = \mathcal{K} \). \( \square \)

**Lemma B.2**  In an orthogonal coordinate system, the torsion form \( T \) in eq.(4.3) is related to the connection form \( \omega_{12} \) as

\[
T = \omega_{12} \mathcal{M}^{-1}
\]

where the metric \( \mathcal{M} \) is as defined in (4.4).
Proof It can be shown (using the structural equations obeyed by the differential forms on a surface [7]) that the connection form $\omega_{12}$ can be written in terms of the lengths $\sqrt{E}$ and $\sqrt{G}$ of the tangent vectors $f_u$ and $f_v$ as [82]

$$\omega_{12} = \frac{(-\sqrt{E})_u}{\sqrt{G}} du + \frac{(\sqrt{G})_u}{\sqrt{E}} dv$$

Hence, in vector form, $\omega_{12}$ may be written as

$$[\omega_{12}] = \begin{bmatrix} \frac{(-\sqrt{E})_u}{\sqrt{G}} & \frac{(\sqrt{G})_u}{\sqrt{E}} \end{bmatrix}$$ (B.9)

(From (B.2), $E = f_u \cdot f_u$ and $G = f_v \cdot f_v$.)

Using the definition of the torsion form $\mathcal{T}$ in (4.3) and eqs.(B.2), $\mathcal{T}$ can be written in terms of $E, G$ as

$$\mathcal{T} = \left( \frac{f_v}{\sqrt{G}} \right) \cdot \begin{bmatrix} (E_1)_u & (E_1)_v \end{bmatrix}$$ (B.10)

For any vector $v = [v_1, v_2, v_3]$ which is a function of some scalar variable $z$, consider the component-wise differentiation

$$\frac{\partial}{\partial z} \left( \frac{v_i}{\sqrt{v \cdot v}} \right) = \frac{1}{v \cdot v} \left[ \sqrt{v \cdot v} \frac{\partial v_i}{\partial z} - \frac{1}{2} v_i (v \cdot v)^{-1/2} \left( \sum_{i=1}^{3} 2 v_i \cdot \frac{\partial v_i}{\partial z} \right) \right]$$

Thus, we obtain the following

$$\frac{\partial}{\partial z} \left( \frac{v}{\sqrt{v \cdot v}} \right) = \frac{v_z}{\sqrt{v \cdot v}} - \left( \frac{v \cdot v_z}{(v \cdot v)^{3/2}} \right) v$$

Using the above expansion, we get

$$(E_1)_u = \frac{\partial}{\partial u} \left( \frac{f_u}{\sqrt{f_u \cdot f_u}} \right) = \frac{f_{uu}}{\sqrt{f_u \cdot f_u}} \left( \frac{f_u \cdot f_{uu}}{f_u \cdot f_u} \right) f_u$$ (B.11)

$$(E_1)_v = \frac{\partial}{\partial v} \left( \frac{f_u}{\sqrt{f_u \cdot f_u}} \right) = \frac{f_{uv}}{\sqrt{f_u \cdot f_u}} \left( \frac{f_u \cdot f_{uv}}{f_u \cdot f_u} \right) f_u$$ (B.12)

Substituting expressions (B.11) and (B.12) in (B.10) and noting that, for an orthogonal coordinate system, $f_v \cdot f_u = 0$,

$$\mathcal{T} = \begin{bmatrix} \frac{f_v \cdot f_{uu}}{E\sqrt{G}} & \frac{f_v \cdot f_{uv}}{G\sqrt{E}} \end{bmatrix}$$ (B.13)
Once again, for a vector \( v = [v_1 \ v_2 \ v_3] \) which is a function of a scalar variable \( z \), the following expression can be derived.

\[
(\sqrt{v \cdot v})_z = \frac{\partial}{\partial z} \left( \sqrt{\sum_{i=1}^{3} v_i^2} \right) = \frac{1}{2} \left( \sum_{i=1}^{3} v_i^2 \right)^{-1/2} \sum_{i=1}^{3} 2v_i \cdot \frac{\partial v_i}{\partial z} \\
= \frac{1}{\sqrt{v \cdot v}} \left( \sum_{i=1}^{3} v_i \cdot \frac{\partial v_i}{\partial z} \right) = \frac{v \cdot v_z}{\sqrt{v \cdot v}}
\]

Thus, using the result above,

\[
(\sqrt{E})_v = \frac{\partial}{\partial u} (\sqrt{f_u \cdot f_u}) = \frac{f_u \cdot f_{uv}}{\sqrt{E}} \tag{B.14} \\
(\sqrt{G})_u = \frac{\partial}{\partial u} (\sqrt{f_v \cdot f_v}) = \frac{f_v \cdot f_{uv}}{\sqrt{G}} \tag{B.15}
\]

However,

\[ f_u \cdot f_v = 0 \quad \Rightarrow \quad \frac{\partial}{\partial u} (f_u \cdot f_v) = 0 \quad \Rightarrow \quad f_u \cdot f_{uv} = -f_{uu} \cdot f_v \]

Substituting in eq.(B.14), we obtain

\[
(\sqrt{E})_v = \frac{-f_{uu} \cdot f_v}{\sqrt{E}} \quad \Rightarrow \quad f_v \cdot f_{uv} = \frac{\sqrt{E}}{v} (\sqrt{E})_v \tag{B.16}
\]

Also, directly from eq.(B.15)

\[
f_v \cdot f_{uv} = \sqrt{G} (\sqrt{G})_u \tag{B.17}
\]

Now, substituting these expressions in the equation (B.13),

\[
T = \begin{bmatrix}
-\sqrt{E}(\sqrt{E})_v & \sqrt{G}(\sqrt{G})_u \\
E \sqrt{G} & G \sqrt{E}
\end{bmatrix} \\
= \begin{bmatrix}
-\sqrt{E}g_v & -\sqrt{G}u_v \\
\sqrt{EG} & \sqrt{EG}
\end{bmatrix} \\
= \begin{bmatrix}
-\sqrt{E}g_v & -\sqrt{G}u_v \\
\sqrt{G} & \sqrt{E}
\end{bmatrix} \begin{bmatrix}
\frac{1}{\sqrt{E}} & 0 \\
0 & \frac{1}{\sqrt{G}}
\end{bmatrix} \tag{B.18}
\]

Thus, from eqs.(4.4), (B.9) and (B.18),

\[
T = [\omega_{12}] \mathcal{M}^{-1}
\]
Appendix C

Control Algorithm for redundant systems

In section 4.6.2, a control law for controlling the object trajectory, internal forces and contact variables was presented for the case when the number of controlled variables is equal to the number of actuators.

However, in many cases, the number of controlled variables may either be less than or greater than the number of actuators. In the former case, we can devise a straightforward extension of the control law in section 4.6.2, as shown below. This case can occur, for example, when three robotic manipulators with 6 joints each, are used to manipulate a common object. In this case, the number of actuators (and hence the dimension of the vector $\tau$) is 18. There will be 6 contact variables to be controlled in addition to the 6 positional degrees of freedom of the object and the three-dimensional force vector $\vec{f}$. Thus, the total number of controlled variables is 15.

Consider the dynamic equations of the system in the form (4.55)

$$ Y \begin{bmatrix} \ddot{u}_o \\ \ddot{\vec{r}}_o \\ \vec{f} \end{bmatrix} + B_1 = M \ddot{x}_o $$  \hspace{1cm} (C.1)

where

$$ Y \triangleq \begin{bmatrix} Y_1 & Y_2 & Y_3 \end{bmatrix} $$

$$ Y_1 \triangleq \begin{bmatrix} I \\ 0 \end{bmatrix} ; \quad Y_2 \triangleq LA^{-1}A_o \; ; \quad Y_3 \triangleq LA^{-1}H^T\tilde{W}V $$

$$ B_1 \triangleq LA^{-1}B_o - N_1 \; ; \quad M \triangleq LA^{-1}R $$

In order to ensure that the matrix $Y$ is square, in this case, instead of including the entire constraint matrix $H$ in $L = \begin{bmatrix} J_u^T & H^T \end{bmatrix}^T$ (see eq.(4.53)), only an appropriate number of rows from $H$ must be included. In the example cited above, the matrix $J_u$ has 6 rows, hence 9 rows of the $12 \times 18$ matrix $H$ are used to form the matrix $L$. With this choice of $L$, the matrix $Y$ in eq.(C.1) is square with dimensions $15 \times 15$. However,
because the number of actuators is greater than 15, the matrix $M$ is underdetermined i.e. has more columns than rows.

Hence, the control torques $\tau$ are chosen as

$$\tau_a = M^+ \left\{ Y \begin{bmatrix} \ddot{u}_o - K_v \dot{e}_u - K_p e_u \\ \ddot{v}_o - K_v \dot{e}_v - K_p e_v \\ \int_{\tau_d} - K_i \int e_f \end{bmatrix} + B_1 \right\}$$

(C.2)

Here, $M^+$ is the pseudoinverse of the matrix $M$. As long as $M$ has full row rank, $MM^+ = I$. Note that, instead of the pseudoinverse, any righthanded inverse of $M$ can be used to compute the torque vector. Substituting this control law in (C.1),

$$Y \begin{bmatrix} \ddot{u}_o \\ \ddot{v}_o \\ \int_{\tau_d} \end{bmatrix} + B_1 = MM^+ \left\{ Y \begin{bmatrix} \ddot{u}_o - K_v \dot{e}_u - K_p e_u \\ \ddot{v}_o - K_v \dot{e}_v - K_p e_v \\ \int_{\tau_d} - K_i \int e_f \end{bmatrix} + B_1 \right\}$$

$$\Rightarrow Y \begin{bmatrix} \dot{e}_u + K_v \dot{e}_u + K_p e_u \\ \dot{e}_v + K_v \dot{e}_v + K_p e_v \\ e_f + K_i \int e_f \end{bmatrix} = 0$$

Since $Y$ is a square matrix and will be nonsingular as long as the grasp points are distinct and the manipulators are not at singular configurations, it can be inverted, to obtain the same error equations as before i.e.

$$\dot{e}_u + K_v \dot{e}_u + K_p e_u = 0$$
$$\dot{e}_v + K_v \dot{e}_v + K_p e_v = 0$$
$$e_f + K_i \int e_f = 0$$

Hence, with appropriate selection of $K_v$, $K_p$, $K_i$, the errors $e_u$, $e_v$ and $e_f$ go to zero.

On the other hand, when the sum of the dimensions of the contact variables, the object’s position and the internal forces is greater than the number of actuators, all the variables cannot be controlled independently. In this case, depending on the application, the appropriate variables to be controlled can be selected. Usually, we would always like to control the object’s trajectory and the internal forces it experiences due to the grasp. Hence, when the number of actuators is not enough, the number of contact variables being explicitly controlled can be reduced. For instance, in repositioning a particular grasp of a multifingered hand, an additional finger can
be employed. In this case, it is not necessary to control the trajectory of this finger exactly as it is only being used for support.

Once again, it is the choice of $L$ (eq. 4.33) that needs to be modified to achieve a control law for this case. When the number of actuators were extra, the number of rows of the constraint matrix $H$ which are included in $L$ were reduced. Now, since the number of contact variables being controlled are being reduced, we reduce the corresponding rows of the matrix $J_u$ included in $L$. Thus, only those rows of $J_u$ which correspond to the $u^i$'s being controlled are incorporated in $L$. The number of $u^i$'s chosen is such that the sum of the dimensions of the three controlled variables $u_o$, $\bar{x}_o$ and $\bar{f}$ is equal to the total number of actuators in the system.

With this choice of $L$, the matrices $Y$ and $M$ in eq.(C.1) will now be square and hence the same control law as in theorem 4.2 can be used.
Appendix D

Parameters used in Controller simulations

For the actual system, the links of all the manipulators were taken to be of unit length and unit mass. The center of mass of each link was taken to be at the midpoint of the link. The object was also assumed to have unit mass and unity radius, with its center of mass at its geometric center.

Table D.1  Link lengths and masses

<table>
<thead>
<tr>
<th>MANIPULATOR</th>
<th>LINK</th>
<th>LENGTH</th>
<th>MASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0.8159</td>
<td>1.2515</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0.8449</td>
<td>1.0577</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>1.1181</td>
<td>1.0333</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0.8136</td>
<td>0.8684</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0.8795</td>
<td>0.7411</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>1.2546</td>
<td>0.9922</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0.8351</td>
<td>1.1988</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0.8787</td>
<td>0.9198</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0.7911</td>
<td>0.8472</td>
</tr>
</tbody>
</table>
The parameters used in the controller were chosen randomly, to have about a 15% average difference from those that were used for the system. The lengths and masses of the manipulator links are enumerated in Table D.

The controller gains were as follows -

\[
K_{v_2} = \begin{bmatrix}
9 & 0 & 0 \\
0 & 9 & 0 \\
0 & 0 & 9
\end{bmatrix} \quad K_{p_x} = \begin{bmatrix}
25 & 0 & 0 \\
0 & 25 & 0 \\
0 & 0 & 25
\end{bmatrix}
\]

\[
K_{v_1} = \begin{bmatrix}
16.2 & 0 & 0 \\
0 & 12.8 & 0 \\
0 & 0 & 10.8
\end{bmatrix} \quad K_{p_u} = \begin{bmatrix}
81 & 0 & 0 \\
0 & 64 & 0 \\
0 & 0 & 36
\end{bmatrix}
\]

\[
K_i = \begin{bmatrix}
25 & 0 & 0 \\
0 & 25 & 0 \\
0 & 0 & 25
\end{bmatrix}
\]
Bibliography


[51] V. Kumar, X. Yun, E. Paljug, and N. Sarkar. Control of contact conditions for


[53] Z. Li and J. Canny. Motion of two rigid bodies with rolling constraint. IEEE

[54] A. Liegeois. Automatic supervisory control of the configuration and behavior of
multibody mechanisms. IEEE Transactions on Systems, Man and Cybernetics,

using {1}-inverses. IEEE Transactions on Systems, Man and Cybernetics, SMC-

[56] D.G. Luenberger. Optimization by vector space methods. John Wiley & Sons,

[57] D.G. Luenberger. Linear and nonlinear programming. Addison-Wesley, Reading,

with closed kinematic chain mechanisms. IEEE Journal of Robotics and Au-

[59] A.A. Maciejewski and C.A. Klein. Obstacle avoidance for kinematically re-
dundant manipulators in dynamically varying environments. The International

[60] A.A. Maciejewski and C.A. Klein. Numerical filtering for the operation of
robotic manipulators through kinematically singular configurations. Journal of

[61] A.A. Maciejewski and C.A. Klein. The singular value decomposition: Compu-
tation and application to robotics. The International Journal of Robotics


