INVERSE POSITION SOLUTIONS, WORKSPACE, AND
RESOLVED RATE CONTROL OF
ALL POSSIBLE 3-DOF PARALLEL PLANAR MANIPULATORS

A Thesis Presented to
The Faculty of the
Fritz J. and Dolores H. Russ
College of Engineering and Technology
Ohio University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

by
Brett Shelley
November, 1996
I wish to acknowledge the individuals who have made my pursuit of a Master of Science degree a truly rewarding experience. Dr. Robert L. Williams II has been a focused instructor and dedicated advisor. With his assistance and mentorship, an extremely challenging academic experience has also been an enjoyable one. Dr. Hajrudin Pasic’s assistance allowed me to mature as a graduate student and complete the program much earlier than I would have otherwise. Most importantly, Daniela Wehr has stood by me from afar and never doubted the outcome of my work. She has been my source of motivation. I thank her for her encouragement, patience, and love.
This thesis studies all possible configurations of a specific class of parallel planar three degree-of-freedom robots. Inverse position control, resolved-rate control, and numerically-determined workspace algorithms are presented. The position and resolved-rate algorithms may be used to control any possible configuration of the class of robots in question. The workspace algorithms enable one to determine the workspaces of any configuration and are useful in robot design.

Since this class of robots is planar, the Euler closed-loop vector equations were most efficient in developing inverse position and inverse velocity equations. Throughout the work, analytical solutions were tested in Matlab and results were simulated. Illogical joint values from inverse position solutions formed the foundation for workspace calculations.

This thesis solves the inverse position problem for all possible configurations. Results are verified using inverse position simulations found throughout this thesis. Algorithms for numerical determination of dexterous and reachable workspace are developed and simulations are presented. Furthermore, the Jacobian matrices and inverse Jacobians for each independent chain and overall robot are derived. The overall inverse Jacobian for any possible configuration is used to simulate resolved-rate control. The inverse velocity analysis identifies four additional actuated chains that make a robot uncontrollable.
# Table of Contents

1 Introduction

1.1 Background and Literature review .................................................. 1

1.2 Class of Robots ................................................................................. 2

1.3 Thesis Outline ................................................................................... 3

2 Inverse Position

2.1 Introduction ....................................................................................... 5

2.2 Definition of Variables ....................................................................... 5

2.3 Irrelevance of Actuation ................................................................. 6

2.4 Independence of Chains and Platform Geometry .......................... 7

2.5 Solving for Joint Values in an Entire Robot .................................... 9

2.6 Specific Chain Solutions ............................................................... 10

2.7 Example Chain Solutions ............................................................... 19

2.8 Complete Example Manipulators ................................................. 22

2.9 Summary ......................................................................................... 25

3 Workspace

3.1 Introduction ....................................................................................... 26

3.2 Feasibility of Analytical Solutions for All Possible Architectures .... 27

3.3 Logic Behind Algorithm ............................................................... 27

3.4 Example Causes of Illogical Joint Values ..................................... 28

3.5 Workspace Computation ............................................................... 31
LIST OF TABLES

Table 2.1: Merlet’s Actuated Serial Chain Combinations ............................................. 6
Table 2.2: All Possible Chains for Inverse Position Problem ........................................ 7
Table 2.3: Serial Chain Variable Listing .......................................................................... 10
Table 2.4: Serial Chain Solutions .................................................................................... 25
Table 3.1: Illogical Joint Values Listing ........................................................................... 31
Table 4.1: Uncontrollable Actuation Schemes .................................................................. 54
Table 4.2: Modified Merlet Table .................................................................................... 49

LIST OF FIGURES

Figure 1.1: Example 3 RRR Robot .................................................................................. 2
Figure 2.1: Serial Chain Arrangement for All Configurations ......................................... 6
Figure 2.2: Platform Geometry Diagram ......................................................................... 8
Figure 2.3: Diagram of RRR Serial Chain ....................................................................... 12
Figure 2.4: Diagram of RPR Serial Chain ....................................................................... 14
Figure 2.5: Diagram of PRR Serial Chain ....................................................................... 15
Figure 2.6: Diagram of PPR Serial Chain ....................................................................... 15
Figure 2.7: Diagram of RRP Serial Chain ....................................................................... 17
Figure 2.8: Diagram of PRP Serial Chain ....................................................................... 18
Figure 2.9: Diagram of RPP Serial Chain ....................................................................... 19
Figure 2.10: RRR Solutions ............................................................................................ 19
Figure 2.11: RPR Solution ............................................................................................... 20
Figure 2.12: RPP Solution ............................................................................................... 20
Figure 2.13: PRR Solution ............................................................................................... 21
Figure 2.14: PRP Solution ............................................................................................... 21
Figure 2.15: Example RPP-RRR-PRR Manipulator .......................................................... 23
Figure 2.16: “Ohio” Position Control and Joint Positions Example .................................. 24
Figure 3.1: PRR Chain without solution ......................................................................... 28
Figure 3.2: PPR Chain without solution..................................................29
Figure 3.3: RRR Chain without solution..................................................30
Figure 3.4: Workspace Program Algorithm..........................................33
Figure 3.5: Integration of Joint Limits....................................................34
Figure 3.6: Workspace of 3-PRP Manipulator........................................36
Figure 3.7: Reachable Workspace of 3-PRP Manipulator..........................36
Figure 3.8: Dexterous Workspace of 3-PRP Manipulator.........................36
Figure 3.9: 3-D View of 3-PRP Reachable Workspace..............................37
Figure 3.10: Workspace of Random Manipulator.....................................37
Figure 3.11: Reachable Workspace of Random Manipulator.....................37
Figure 3.12: Dexterous Workspace of Random Manipulator......................38
Figure 4.1: Comparison of Relative and Absolute Angles.........................40
Figure 4.2: Example 3-RRR Robot.........................................................49
Figure 4.3: Resolved Rate Control Block Diagram....................................51
Figure 4.4: Resolved Rate Control Simulation........................................52
CHAPTER 1

Introduction

1.1 Background and Literature Review

Parallel planar manipulators are robots that consist of separate serial chains that connect the fixed link to a moving platform. An end-effector is attached to the moving platform. The main motivation is that they provide better stiffness and accuracy than serial kinematic chains. Also, this architecture allows actuators to be fixed to the base link. This minimizes inertia of moving parts and allows the use of more powerful actuators.

The application of parallel mechanical devices in robotics was proposed approximately seventeen years ago. (MacCallion and Pham, 1979) Some configurations have been built and analyzed to demonstrate their properties. These manipulators have been used in terrestrial vehicle simulators, tire-testing machines, metal cutting, and part handling.

Since 1979, numerous works have been published analyzing the kinematics, dynamics, workspace and control of parallel manipulators (Williams, 1988) Hunt conducted preliminary studies of various parallel robot configurations. (Hunt, 1982, 1983) Cox and Tesar (1981) compared the relative merits of serial and parallel robots.

These past works have focused on only a few different architectures. For example, Aradyfio and Qiao (1985) examine the inverse kinematics solutions for three different
three degree-of-freedom planar robotic mechanisms. Also, Gosselin and Angeles (1988) study the kinematics and workspace of one planar three degree-of-freedom parallel robot.

Recently, Merlet identified all of the possible chains for a specific class of planar parallel robots. He identified eighteen possible chains and solved the direct kinematics position problem. (Merlet, 1996)

1.2 Class of Robots

The class of robots in this thesis have the following characteristics. The robots operate within a plane and are parallel. That is, a parallel robot has separate serial chains joining together to control one end-effector. Three serial chains with three degrees of freedom each connect the base link to the platform. Furthermore, one joint per chain is actuated and the remaining two in each chain are passive. The active and passive joints may be revolute (R) or prismatic (P). (See Fig. 1.1 for a general description of this class of robots.)

Figure 1.1
Example 3-RRR Robot
The degrees of freedom for the entire robot (regardless of whether joints are revolute or prismatic) is calculated below. In the equation, $M$, $N$, $J_1$, and $J_2$ represent the degrees of freedom, the number of links, the number of 1-dof joints, and the number of 2-dof joints, respectively.

$$M = 3 \cdot (N - 1) - 2 \cdot (J_1) - J_2$$

$$M = 3 \cdot (8 - 1) - 2 \cdot (9) = 3$$

1.3 Thesis Outline

Like Merlet's work, this thesis considers all possible 3-dof planar parallel manipulators. His work focuses on forward position kinematics. This work analyzes all possible chains from an inverse kinematics standpoint. This work analyzes inverse position solutions, workspaces, and inverse velocity of the manipulators.

Initially, this work develops the analytical expressions necessary to find all the possible joint values when the platform is given position and orientation. Chapter two presents this inverse kinematics position problem.

Chapter three addresses a method to determine the workspace (dexterous and reachable workspace) for all possible architectures of planar, three degree-of-freedom parallel robots. The workspace in this thesis is defined as the region of the plane that the manipulator can reach given a fixed orientation $\phi$. The dexterous workspace is defined as a region in the plane where a given manipulator can reach with all orientations or a given range of orientations. Finally, the reachable workspace is defined as the region within the plane that the manipulator can reach with any orientation.
Chapter four develops velocity expressions, the Jacobian and inverse Jacobian matrices for each individual chain, and the manipulator’s inverse Jacobian matrices. In Merlet’s work, he suggested that any actuation scheme could be employed as long as the resulting “free joints” in any one chain were not of the PP type. This chapter demonstrates that four more limits to actuation exist beyond the three cases that Merlet identified. Section 4.7 uses resolved-rate velocity control and the above equations to exemplify possible applications of the analytical equations.

Computer simulations of results are applied throughout this work. In the case of determining workspaces for any possible combination of chains, numerical analysis is essential. Regardless of the topic, computerized analysis is an essential tool in verifying the general case equations developed in chapters two, three, and four.

Chapter five draws conclusions from the present research and addresses recommended future work.
CHAPTER 2
Inverse Kinematics Position Solutions

2.1 Introduction

Merlet's Article "Direct Kinematics of Planar Parallel Manipulators" identifies eighteen different possible kinematic arrangements for each serial chain of the 3 DOF planar parallel robot. He solves the forward position kinematics problem for all combinations of arrangements, including all possible actuation schemes. This chapter focuses on developing inverse position solutions for all of these kinematic configurations.

The inverse position problem is defined as follows: Given any platform position and orientation (x, y, and θ), determine the actuated joint values necessary to achieve the position and orientation.

The robot inverse position solution is achieved by independently solving three serial inverse position problems, one for each chain. The resulting solutions are verified using Matlab programs that solve joint values for each chain and graphically simulate the resulting chain configuration.

2.2 Definition of Variables

The following symbols appear in analytical solutions and programs. They are defined as follows: Figure 2.1 illustrates these definitions. The definitions are applicable to all seven serial chains and any overall robot configuration.

- $A_i$: point where $i^{th}$ chain is attached to base link
- $B_i$: point where link's $L_{1i}$ and $L_{2i}$ are attached
- $C_i$: point where $i^{th}$ independent chain is attached to platform
- $L_{1i}$: length of the link attached to base point $A_i$
- $L_{2i}$: length of the link connecting point $B_i$ to point $C_i$
- $L_{3i}$: distance between $C_i$ and end effector (x,y)
θ_{1i}- counterclockwise angle formed between X-axis of the inertial frame and a vector drawn from point A_i along L_{1i}.
θ_{2i}- counterclockwise angle formed between a vector drawn from point A_i along L_{1i} and a vector drawn from point B_i drawn along L_{2i}.
θ_{3i}- counterclockwise angle formed between a vector drawn from point B_i along L_{2i} and a vector drawn from point C_i to end effector (x,y).
ψ_{i}- angle drawn between a vector from end effector to C_i and from inertial X axis drawn from end effector.

2.3 Irrelevance of actuation joints

The eighteen possible kinematic chains of the three-dof parallel, planar manipulators are given in Table 2.1. Underlined letters refer to the actuated revolute or prismatic joint.

<table>
<thead>
<tr>
<th>RRR</th>
<th>RRR</th>
<th>RRR</th>
<th>RPR</th>
<th>RPR</th>
<th>RPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPP</td>
<td>RPP</td>
<td>PRR</td>
<td>PRR</td>
<td>PRR</td>
<td>PRR</td>
</tr>
<tr>
<td>PRP</td>
<td>PPR</td>
<td>PPR</td>
<td>RRP</td>
<td>RRP</td>
<td>RRP</td>
</tr>
</tbody>
</table>

No PPP configurations are possible in this class of robots since the link L_{3i} is fixed on the platform.

In the inverse position problem, the actuation scheme will not affect the joint values necessary to obtain x,y, and φ. Unlike the forward kinematics problem, actuation
on each chain is irrelevant, the joint values will be the same. As a result, the eighteen possible serial chains from forward kinematics yields only seven possible serial chains for inverse kinematics. The seven chains are:

| Table 2.2 |
| All Possible Serial Chains for Inverse Position Problem |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| RRR       | RPR       | RPP       | PRR       | PRP       | PPR       | RRP       |

2.4 Independence of Chains and Platform Geometry

The pose (x, y, φ) of any point on the platform is achieved by three independent kinematic chains connected to the platform at points C₁, C₂, and C₃. These chains are attached to the fixed link at points A₁, A₂, and A₃. Since a given pose determines the exact position of points Cᵢ, the joint values of each chain are solved independently of the other chains on the manipulator.

The point Cᵢ of the robot is fixed by the geometry of the platform and inverse kinematic input. The designer chooses a point on the platform where a tool would normally be attached. For the sake of simplicity, the center of a triangle may be chosen as the end-effector (x, y). The orientation angle φ is defined as zero when the base of the triangle is in a horizontal position. The designer chooses three points on the platform where the independent chains will attach the platform to base points. These points are C₁, C₂, and C₃ in Fig. 2.2.
Since the platform is solid, fixed angles, $\psi_1, \psi_2,$ and $\psi_3,$ exist between the end effector $(x, y)$ and points $C_{1,2,3}$ on the platform. They are defined as the angle formed between the positive X-axis of the universal frame translated to the end-effector and a vector drawn from $(x, y)$ to $C_i$ when $\phi = 0.$ In Fig. 2.2, $\psi_1=210^\circ,$ $\psi_2=90^\circ,$ and $\psi_3=-30^\circ.$ Furthermore, the length of the vector from the tool to $C_i$ is fixed and is defined as $L_{3i}.$

Determination of point $C_i$ is then accomplished as follows.

\begin{align*}
C_{ix} & = x + L_{3i} \cos(\phi + \psi_i) \\
C_{iy} & = y + L_{3i} \sin(\phi + \psi_i)
\end{align*}

With these equations and the independence of chains in the inverse position problem, the joint values can be solved between points $A_i$ and $C_i.$ (Refer to Fig. 2.1 and 2.2) The diagram in Fig. 2.1 represents all seven possible independent serial chains.
2.5 Solving for Joint Values in an Entire Robot

In the general case for an entire robot, one must be able to solve the joint values of each chain independently. Each chain has three degrees of freedom. That is, each chain has three unknowns. The designer and the control input fixes points $A_i$ and $C_i$. For the entire robot, one must solve for the three variable joints in each of the three independent chains to solve the entire inverse position problem. This may be done by considering representation of each chain's known point $C_i$ using the Euler identity (Eq. 2.3). This identity allows a planar vector to be $C_i$ to be represented as in Eq. 2.4.

$$e^{\pm j\theta} = \cos \theta \pm j \sin \theta \quad 2.3$$

$$C_i = A_i + L_1 e^{j\theta_1} + L_2 e^{j(\theta_1 + \theta_2)} \quad 2.4$$

Breaking down Eq. 2.2 into the $x$ and $y$ components allows $C_i$ to be represented:

$$A_x + L_1 c_{11} + L_2 c_{12} = C_x \quad 2.5$$

$$A_y + L_1 s_{11} + L_2 s_{12} = C_y \quad 2.6$$

Furthermore, the angle $\theta_1 + \theta_2 + \theta_3$ points directly to the end-effector $(x,y)$ from point $C_i$ when the angles are added from the inertial X-axis. The angle $\phi + \psi_i$ points directly at $C_i$ using the X-axis. These two angles differ by 180°. Thus, the following relationship exists:

$$\theta_1 + \theta_2 + \theta_3 + \pi = \phi + \psi_i \quad 2.7$$

Equations 2.5-2.7 represent three equations that can be used to solve for three unknowns. Solving for a specific chain such as the RRR chain or the RPR chain is a matter of determining solutions for the unknowns. These solutions are found below for each of the seven serial chains.
2.6 Specific Chain Solutions

Again, each of the seven chains has three unknowns to solve in the inverse position problem. The unknowns for each chain are summarized in the Table 2.3.

Table 2.3
Serial Chain Variable Listing

<table>
<thead>
<tr>
<th>Chain</th>
<th>$\theta_{1i}$</th>
<th>$\theta_{2i}$</th>
<th>$\theta_{3i}$</th>
<th>$L_{1i}$</th>
<th>$L_{2i}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPR</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RPP</td>
<td>x</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RRP</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>PRR</td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>PRP</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PPR</td>
<td></td>
<td></td>
<td>x</td>
<td></td>
<td>x</td>
</tr>
<tr>
<td>RRR</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A. RRR: The problem is defined as follows: Given $x$, $y$, and $\phi$, determine joint angles $\theta_1$, $\theta_2$, and $\theta_3$. To solve for $\theta_1$, it is necessary to eliminate $\theta_2$. To eliminate $\theta_2$, Eqs. 2.5 and 2.6 are squared and added after isolating $(\theta_1 + \theta_2)$ terms. Squared, they are:

$$(L_2 c_{12})^2 = A_x^2 + C_x^2 + L_1^2 c_{12} - 2C_x A_x - 2C_y L_1 c_1 + 2A_x L_1 c_1$$  \hspace{1cm} 2.8

$$(L_2 s_{12})^2 = A_y^2 + C_y^2 + L_1^2 s_{12} - 2C_y A_y - 2C_x L_1 s_1 + 2A_y L_1 s_1$$  \hspace{1cm} 2.9

By adding the two equations together and rearranging, $\theta_2$ is eliminated:

$$L_2^2 - A_x^2 - C_x^2 - A_y^2 - C_y^2 - L_1^2 + 2C_x A_x + 2C_y A_y + (2C_x L_1 - 2A_x L_1) c_1 + (2C_y L_1 - 2A_y L_1) s_1 = 0$$  \hspace{1cm} 2.10
To simplify, the following terms are defined:

\[
G = L_z^2 - A_x^2 - C_x^2 - A_y^2 - C_y^2 - L_i^2 + 2C_xA_x + 2C_yA_y \tag{2.11}
\]

\[
E = 2C_xL_i - 2A_xL_i \tag{2.12}
\]

\[
F = 2C_yL_i - 2A_yL_i \tag{2.13}
\]

The new simplified equation is:

\[
Ec_1 + Fs_1 + G = 0 \tag{2.14}
\]

The tangent of the half angle equations are now used to solve for \( \theta_1 \):

\[
t = \tan \left(\frac{\theta}{2}\right) \tag{2.15}
\]

\[
\cos \theta = \frac{1 - t^2}{1 + t^2} \tag{2.16}
\]

\[
\sin \theta = \frac{2t}{1 + t^2} \tag{2.17}
\]

The equation becomes:

\[
(G - E)t^2 + 2Ft + (G + E) = 0 \tag{2.18}
\]

Using the quadratic formula, \( t \) is solved:

\[
t = \frac{-2F \pm \sqrt{(2F)^2 - 4(G - E)(G + E)}}{2(G - E)} \tag{2.19}
\]

\[
t = \frac{-F \pm \sqrt{F^2 - G^2 + E^2}}{G - E} \tag{2.20}
\]

With two solutions for \( t \), \( \theta_1 \) is solved by inverting Eq. 2.15:

\[
\theta_1 = 2 \tan^{-1}(t_{1,2}) \tag{2.21}
\]

\( \theta_1 \) has two solutions which correspond to the “elbow-up” and “elbow-down” solutions (see Fig. 2.5, Section 2.7). To solve for \( \theta_2 \), one returns to Eqs. 2.5 and 2.6. If the equations are rearranged, and the y-equation is divided by the x-equation, then one has:

\[
L_2s_{12} = C_y - (A_y + L_is_i) \tag{2.22}
\]

\[
L_2c_{12} = C_x - (A_x + L_ic_i) \tag{2.23}
\]
\[ \tan(\theta_1 + \theta_2) = \frac{C_y - (A_y + L_1 s_1)}{C_x - (A_x + L_1 c_1)} \]

Using the two argument inverse tangent function, the value of \( \theta_2 \) is determined:

\[ \theta_2 = \alpha \tan 2\left( C_y - A_y - L_1 s_1, C_x - A_x - L_1 c_1 \right) - \theta_1 \]

\( \theta_3 \) is then solved using Eq. 2.7:

\[ \theta_3 = \psi + \phi - \theta_1 - \theta_2 - \pi \]

**Figure 2.3**

Diagram of RRR Serial Chain

**B. RPR**: The RPR kinematic chain consists of a revolute joint at the base \((A_x, A_y)\), a prismatic joint replacing a fixed \(L_1\) or \(L_2\), and a revolute joint on the platform at \((C_x, C_y)\).

The inverse kinematics position problem is defined as follows: Given \(x, y, \phi\), determine \(\theta_1, L_1\) (or \(L_2\)), and \(\theta_3\) necessary to achieve that position and orientation. In the derivation that follows, \(L_1\) is chosen as the prismatic joint. Eqs. 2.5 and 2.6 are rearranged to begin isolating knowns from unknowns:

\[ L_1 c_1 + L_2 c_{12} = C_x - A_x \]
\[ L_1 s_1 + L_2 s_{12} = C_y - A_y \]

The sum of angles formulas are then used to expand the equations and solve for \(L_1\):

\[ L_1 c_1 + L_2 (c_1 c_2 - s_1 s_2) = C_x - A_x \]
\[ L_1 s_1 + L_2 (c_1 s_2 + s_1 c_2) = C_y - A_y \]
L_1 is then substituted into Eq. 2.30 and simplified:

\[
L_1 = \frac{C_x - A_x - L_2 (c_1 c_2 - s_1 s_2)}{c_1} \tag{2.31}
\]

Eq. 2.36 has the same form as Eq. 2.14 where:

\[
E = C_y - A_y \tag{2.37}
\]
\[
F = A_x - C_x \tag{2.38}
\]
\[
G = -L_2 s_2 \tag{2.39}
\]

\( \theta_1 \) may then be solved using Eqs. 2.15-2.21. \( L_1 \) is solved using Eq. 2.31 and \( \theta_3 \) is solved using Eq. 2.26. Using this method, two solutions are determined. One of these solutions corresponds to a negative prismatic joint length. This solution is normally discarded.

Thus, one solution exists.
C. PRR: The inverse kinematics position problem is defined as follows: Given x, y, and \( \phi \), calculate the joint values \( L_{i1}, \theta_{2i}, \) and \( \theta_{3i} \), necessary to achieve the given position and orientation of the platform.

In this chain, a prismatic joint \( L_1 \) is fixed at the base creating a fixed angle \( \theta_1 \). The joints connecting \( L_1 \) to \( L_2 \) and \( L_3 \) to the platform are revolute. Again, in any pose, the points \( C_i \) are fixed. The solution to these three joints is derived as follows:

The derivation for this solution is identical to the RPR manipulator from Eqs. 2.27-2.35.

\[
\left( C_x - A_x \right) s_1 + L_2 s_2 = c_1 \left( C_y - A_y \right) \tag{2.35}
\]

In this case however, the variable to be solved is \( \theta_2 \), not \( \theta_1 \). Thus, \( \theta_2 \) may be solved by rearranging the equation and implementing an inverse sine function.

\[
s_2 = \frac{c_1 \left( C_y - A_y \right) - \left( C_x - A_x \right) s_1}{L_2} \tag{2.40}
\]

\[
\theta_2 = \sin^{-1} \left( \frac{c_1 \left( C_y - A_y \right) - \left( C_x - A_x \right) s_1}{L_2} \right) \tag{2.41}
\]

Two solutions exist, a "near" and "far" solution. The "near" solution for \( \theta_2 \) corresponds to the above equation. Its far solution corresponds to the angle in the second and third quadrant. Thus, the far solution is calculated using:

\[
\theta_{2,FAR} = \pi - \theta_2 \tag{2.42}
\]

\( L_1 \) and \( \theta_3 \) are solved respectively by using Eq. 2.31 and by rearranging Eq. 2.7.
D. PPR: The inverse kinematics position problem is: Given \(x, y,\) and \(\phi,\) determine \(L_1,\) \(L_2,\) and \(\theta_3.\) The angles \(\theta_1\) and \(\theta_2\) are fixed by the designer. The derivation for this solution follows Eqs. 2.27-2.35. In this chain, the unknown is \(L_2\) and is solved by rearranging Eq. 2.35.

\[
L_2 = \frac{c_1 \left(C_y - A_y\right) - \left(C_x - A_x\right) s_1}{s_2}
\] \hspace{1cm} 2.43

\(L_1\) and \(\theta_3\) are solved using Eqns. 2.31 and 2.26. One solution exists when negative prismatic joint lengths are discarded.
E. RRP: The inverse position problem of the RRP chain is: Given \( x, y, \) and \( \phi \), calculate the joint values \( \theta_1, \theta_2, \) and \( L_3 \). The solution is nearly identical to the PRR solution. The derivation of the solution begins with Eqs. 2.27 and 2.28. These two equations contain three unknowns, thus an additional equation is needed. From Eq. 2.7, one can write \( \theta_2 \) in terms of \( \theta_1 \) and known terms:

\[
\theta_2 = \psi_i + \phi - \theta_3 - \theta_1 - \pi
\]

2.44

The terms \( \psi, \phi, \pi, \) and \( \theta_3 \) are fixed by the designer and the controller input. Equations are simplified by substituting the term \( \zeta \) for the terms as follows:

\[
\zeta = \psi_i + \phi - \theta_3 - \pi
\]

2.45

\[
\theta_2 = \zeta - \theta_1
\]

2.46

This substitution allows Eqs. 2.26 and 2.27 to be rewritten as:

\[
L_1c_1 + L_2 \cos(\theta_1 + \zeta - \theta_1) = C_x - A_x
\]

2.47

\[
L_1s_1 + L_2 \sin(\theta_1 + \zeta - \theta_1) = C_y - A_y
\]

2.48

which simplifies to

\[
L_1c_1 + L_2 c \zeta = C_x - A_x
\]

2.49

\[
L_1s_1 + L_2 s \zeta = C_y - A_y
\]

2.50

These two equations now have two unknowns. \( L_2 \) is now solved for in Eq. 2.47 and substituted into Eq. 2.48.

\[
L_2 = \frac{C_x - A_x - L_1 c_1}{c \zeta}
\]

2.51

\[
L_1s_1 + \left( \frac{C_x - A_x - L_1 c_1}{c \zeta} \right) s \zeta = C_y - A_y
\]

2.52

The terms are expanded and collected:

\[
L_1s_1 c \zeta + C_x - A_x - L_1 c_1 s \zeta = C_y c \zeta - A_y c \zeta
\]

2.53
By letting E, F, and G be assigned the following variables, one can solve for $\theta_1$ using Eqs. 2.11-2.21.

\[
E = L_1 s_z \\
F = -L_2 c_z \\
G = C_y c_z - A_y c_z + A_x - C_x
\]

The two solutions correspond to a "shorter" and "longer" prismatic joint length.

**Figure 2.7**
Diagram of RRP Serial Chain

**F: PRP:** The inverse position problem is defined as follows: Given x, y, and $\phi$, calculate $L_1$, $L_2$, and $\theta_2$ necessary to reach the position and orientation. In this chain, $\theta_1$ and $\theta_3$ are fixed while $L_1$ and $L_2$ are prismatic joints. The solution is derived as follows:

$\theta_2$ may be solved directly from Eq. 2.44. The two unknowns, $L_1$ and $L_2$ are solved using Eqs. 2.27 and 2.28. The same method for the RRP chain is used up to Eq. 2.53.

\[
L_1 s_z c_z + C_x - A_x - L_2 c_z = C_y c_z - A_y c_z
\]
L_2 is solved using Eq. 2.49. One solution exists.

Figure 2.8
Diagram of PRP Serial Chain

G. RPP: The inverse kinematics position problem of the RPP manipulator is defined as follows:

Given x, y, and \( \phi \), determine \( \theta_1 \), L_1 and L_2. In this chain, angles \( \theta_2 \) and \( \theta_3 \) are fixed. In the RPP manipulator, \( \theta_1 \) is directly determined by Eq. 2.7.

\[
\theta_1 = \psi_i + \phi - \theta_2 - \theta_3 - \pi
\]  

Again, L_2 is solved for in Eq. 2.5 and substituted into Eq. 2.6. L_1 is then solved and substituted back into Eq. 2.60 to solve for L_2.

\[
L_1 = \frac{C_x c_\zeta - A_x c_\zeta - C_x + A_x}{s_1 c_\zeta - c_1 s_\zeta} \quad 2.58
\]

\[
L_2 = \frac{C_x A_x - L_1 c_1}{c_{12}} \quad 2.59
\]

\[
A_y s_1 + L_1 s_1 \left[ \frac{C_x - A_x - L_1 c_1}{c_{12}} \right] s_{12} = C_y \quad 2.61
\]

\[
A_y c_{12} + L_1 s_1 c_{12} + (C_x - A_x - L_1 c_1) s_{12} = C_x c_{12} \quad 2.62
\]

\[
L_1 s_1 c_{12} - L_1 c_1 s_{12} = C_y c_{12} - A_y c_{12} - C_x s_{12} + A_x s_{12} \quad 2.63
\]

\[
L_1 = \frac{C_x c_{12} - A_y c_{12} - C_x s_{12} + A_x s_{12}}{s_1 c_{12} - c_1 s_{12}} \quad 2.64
\]
2.7 Example Chain Solutions

Using Matlab, one may input an end effector pose and calculate joint positions using the analytical solutions developed in the above sections. Furthermore, these solutions may be graphically simulated to view solutions. Example outputs follow.

RRR Example:

Designer Variables:  
\[ L_1 = 7; \quad L_2 = 6; \quad L_3 = 3; \]

Input Pose:  
\[ x = 10; \quad y = 10; \quad \phi = 20^\circ; \]

Output Joint Values:

<table>
<thead>
<tr>
<th>Elbow-Down Branch</th>
<th>Elbow-Up Branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \theta_1 = 15.32^\circ )</td>
<td>( \theta_1 = 72.00^\circ )</td>
</tr>
<tr>
<td>( \theta_2 = 61.97^\circ )</td>
<td>( \theta_2 = -61.97^\circ )</td>
</tr>
<tr>
<td>( \theta_3 = -27.28^\circ )</td>
<td>( \theta_3 = 39.97^\circ )</td>
</tr>
</tbody>
</table>
RPR Example:

Designer Variables:  \( L_2 = 4 \);  \( \theta_2 = -75^\circ \);  \( L_3 = 3 \);
Input Pose:  \( x = 10 \);  \( y = 10 \);  \( \phi = 0^\circ \);

Output Joint Values:

\[ \theta_1 = 68.99^\circ \]  \( L_1 = 9.55 \)  \[ \theta_3 = 36.00^\circ \]

Figure 2.11
RPR Position Solution

RPP Example

Designer Variables:  \( \theta_2 = -55^\circ \);  \( \theta_3 = 15^\circ \);  \( L_3 = 3 \);
Input Pose:  \( x = 10 \);  \( y = 10 \);  \( \phi = 0^\circ \);

Output Joint Values:

\[ \theta_1 = 70.00^\circ \]  \( L_1 = 7.68 \)  \( L_2 = 4.94 \)

Figure 2.12
RPP Position Solution
PRR Example:

Designer Variables: \( \theta_1 = 65^\circ; \) \( L_2 = 5; \) \( L_3 = 3; \)
Input Pose: \( x = 10; \) \( y = 10; \) \( \phi = 12^\circ; \)

Output Joint Values:

Near Solution

\( L_1 = 7.13 \)
\( \theta_2 = -47.13^\circ \)
\( \theta_3 = 24.13^\circ \)

Far Solution

\( L_1 = 13.93 \)
\( \theta_2 = -132.86^\circ \)
\( \theta_3 = 109.86^\circ \)

Figure 2.13
PRR Position Solutions

"Near Solution"  "Far Solution"

PRP Example

Designer Variables: \( \theta_1 = 80^\circ \) \( \theta_3 = 15^\circ \) \( L_3 = 3; \)
Input Pose: \( x = 10; \) \( y = 10; \) \( \phi = 0^\circ; \)

Output Joint Values: \( \theta_2 = -65.00^\circ \) \( L_1 = 6.95 \) \( L_2 = 6.41 \)

Figure 2.14
PRP Position Solution
2.8 Complete Example Manipulators

To solve for all the joint values in any given parallel manipulator, the solutions developed for each individual chain are applied to an entire robot. In the example Matlab output seen below, the resulting joint values are used to graphically simulate the position of all the joints in a given robot. That is to say, if a designer builds a robot and a controller gives the robot a specific pose to achieve, then the individual chain solutions are applied to all three independent chains and the nine unknowns are solved for.

In an actual robot, the controller will only need the three actuated joint values to control the pose of the end effector. However, in a graphical Matlab simulation, all nine independent solutions are needed.

A manipulator’s number of possible inverse solutions ranges from one to eight and is a function of the number of solutions for each chain in a given architecture. A 3-RRR manipulator may have eight \(2^3\) solutions, whereas a 3-PPR manipulator may only have one \(1^3\) solution. The following shows the solution and joint values of a RPP-RRR-PRR robot. It has four \(1 \times 2 \times 2\) solutions.

**RPP-RRR-PRR Manipulator**

Pose of end effector: \(x=10;\ y=10;\ \phi=0^\circ;\)

Designer Fixed Variables:
- 1st Chain: \(\theta_{21}=-50^\circ;\ \theta_{31}=19^\circ;\ L_{31}=3\)
- 2nd Chain: \(L_{12}=8.3;\ L_{22}=5.9;\ L_{32}=3\)
- 3rd Chain: \(\theta_{13}=150^\circ;\ L_{23}=4.2;\ L_{33}=3\)
- Platform: \(\psi_1=210^\circ;\ \psi_2=90^\circ;\ \theta_3=330^\circ\)

Number of Solutions Possible: 4
The above solutions are generated for a whole robot using a Matlab program “posmain.m.” Using the same algorithms, it is also possible to analyze the motion of the robot with the program “movemain.m.” This type of simulation is useful when used to analyze joint motion with a given end-effector path. Analysis of joint motion may indicate whether a singularity exists within the workspace boundary.

**Position Control Simulation Example:** The analytical solutions derived above may be used to control a manipulator. In this simulation, a program “ohiosim.m” gives an end-effector a series of poses to reach. The required joint positions are calculated for each pose and the controller sends instructions to the actuators to reach these poses. In the simulation below, the end-effector traces a path spelling Ohio. The actuated joint positions through the motion may also be plotted as seen below.
Figure 2.16
"Ohio" Position Control and Joint Positions Example
2.9 Summary

The chapter has introduced the process of solving the inverse kinematics problem for all possible three-dof planar parallel robots. Since the actuation scheme does not effect the inverse position solutions, the eighteen kinematic chains identified by Merlet reduce to seven basic chains. Given any pose for a planar manipulator, the three coordinates $C_1$, $C_2$, and $C_3$ are fixed. The joint values in each chain are then solved independently of the other two chains. Each separate chains has a different solution derived from the same three equations. In each case there are three unknowns. Verification of the analytical solutions developed is accomplished in the example solutions in sections 2.7 and 2.8.

For an entire manipulator, the number of inverse solutions (N) follows the following formula:

$$N = (# \text{solutions 1st Chain}) \times (# \text{solutions 2nd Chain}) \times (# \text{solutions 3rd Chain})$$

Each particular chain may have as many as two solutions. In some cases, one of these solutions corresponds to a negative prismatic length and is discarded. Therefore, the table below summarizes the number of possible solutions.

<table>
<thead>
<tr>
<th>Serial Chains</th>
<th>Number of Possible Solutions</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRR</td>
<td>2</td>
</tr>
<tr>
<td>RPR</td>
<td>1</td>
</tr>
<tr>
<td>RPP</td>
<td>1</td>
</tr>
<tr>
<td>PRR</td>
<td>2</td>
</tr>
<tr>
<td>PRP</td>
<td>1</td>
</tr>
<tr>
<td>PPR</td>
<td>1</td>
</tr>
<tr>
<td>RRP</td>
<td>2</td>
</tr>
</tbody>
</table>
CHAPTER 3

Workspace Calculations

3.1 Introduction

One of the most important restrictions in the design of a three-dof parallel manipulator is the workspace. The workspace is represented as a planar region which can be attained by a reference point \((x,y)\) on the platform, for a given fixed orientation of the platform (Craig, 1989). Chapter three addresses the problem of determining the workspace for any given architectures of a 3-dof parallel planar manipulator.

The workspace of all possible architecture of planar parallel robots may be determined with varying levels of precision. Calculated workspaces of some example robots are shown using a numerical method. The method is explained in detail and a user-friendly program to determine workspace for any given architecture is presented.

Using variations of this computational method, three types of workspace may be calculated. First, one may calculate a workspace for a platform's given orientation \(\phi\). Second, it is possible to calculate reachable workspace. Reachable workspace is defined as regions in the plane the end effector can reach with any platform orientation \(\phi\). Lastly, one is able to calculate a mechanism's dexterous workspace. The dexterous workspace (a subset of the reachable workspace) is defined as the region in the plane that the end effector can reach for all \(\phi\) in a range of platform orientations.

As a robot is designed for a specific function, one attempts to maximize workspace with the given constraints. The program developed to determine workspaces allows
complete modification of a robot to calculate a modified robot's new workspace boundary(ies).

3.2 Feasibility of Analytical Solutions for all Possible Architectures

Gosselin, et. al. 1996, analyzes the workspace of a 3-PRR robot. The analysis involves seventeen equations and is calculated using the following algorithm.

(1) Find the curves defining the boundary of the three regions corresponding to the constraints associated with each of the legs independently.
(2) Find intersections of the curves defining the boundary of the three regions.
(3) Divide the curves in elementary portions.
(4) Test each of the elementary portions to determine which ones are part of the envelope of the global workspace. (Gosselin)

Again, this chapter focuses on calculating the workspace of all possible architectures of a planar parallel manipulator. Without considering actuation, $343 (7^3)$ combinations are possible. Developing analytical solutions for each possible architecture using the above method is unrealistic. A numerical method is more efficient and more flexible than 343 analytical solutions.

3.3 Logic behind Algorithm

If a point $(x, y)$ with a given orientation ($\phi$) is within a manipulator's workspace, then each independent kinematic chain is able to independently reach that point. When inverse position solutions do not exist for a given point, then illogical joint values are returned. That is, joint angles are typically returned as complex conjugate and prismatic joint lengths may be negative. A robot's workspace may be graphically determined by exploiting these illogical joint values.
3.4 Example Causes of Illogical Joint Values

Example 1: PRR and RRP chains:

These chains consist of a prismatic joint and two revolute joints. Two inverse position solutions may exist. When a given pose \((x,y,\phi)\) is outside of the workspace, then no solution is possible.

![Figure 3.1: PRR Serial Chain Outside Workspace](image)

With the chosen \((x,y,\phi)\) and chain as seen in Fig. 3.1, no solution exists. Application of the algorithms of Chapter two for the PRR chain will give the following \(\theta_2\) results in radians:

\[ \theta_2 = -1.57 + 0.8378i \]

The values of \(L_1\) and \(\theta_3\) (derived from \(\theta_2\)) will have corresponding complex conjugate values.
Example 2: PPR, PRP, RPP, and RPR Solutions

In Example 1, complex conjugate numbers signal that a given chain is unable to reach a position \((x,y,\phi)\). In Example 2, the negative length of a prismatic joint signals the lack of a solution. The PPR, PRP, and RPP chains consist of two prismatic joints and one revolute joint.

When the solution algorithm for the PPR chain is applied to Fig. 3.2, the following results occur:

\[
\begin{align*}
L_2 &= 8.965 \\
L_1 &= -7.32 \\
\theta_3 &= 15_{\text{deg}}
\end{align*}
\]

![Figure 3.2 PPR Serial Chain Outside Workspace](image)

The joint values are not complex conjugate. However, the joint value \(L_1\) is negative which signals that no solution exists. That is, negative prismatic joint lengths are normally discarded. The RPP, PRP, and RPR chains will also return negative prismatic joint lengths when no solution exists.
Example 3: RRR Chain

When a given pose \((x,y,\phi)\) is outside the workspace, then the RRR algorithm developed in Chapter two will result in the following solutions: (See Fig. 3.3 for specifications.)

**Figure 3.3**

**RRR Serial Chain Outside Workspace**

\[
\begin{align*}
\theta_1 &= -1.6952 - 0.6130i \\
\theta_2 &= 0 + 0.6130i \\
\theta_3 &= 0.1244
\end{align*}
\]

Again, complex conjugate joint angles signal the lack of a solution.

**Example Summary**

When the end-effector's pose is outside the manipulator's workspace of the particular chain, then joint values will still be returned using the algorithms of chapter two. However, these values will be impossible for a real system to achieve. These impossible values manifest themselves in two ways, imaginary angles and negative lengths. The seven possible chains typically give the following output when they cannot reach a particular position:
### Table 3.1
"Illogical" Joint Values Listing

<table>
<thead>
<tr>
<th>Chain\Joint Output</th>
<th>Imaginary Joint Lengths and Angles</th>
<th>Negative Lengths</th>
</tr>
</thead>
<tbody>
<tr>
<td>RRR</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>RPR</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>RPP</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>PRR</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PRP</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>PPR</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>RRP</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

### 3.5 Workspace Computation

The results of inverse position calculations can be used to graphically plot the workspaces, reachable and dexterous, for any planar parallel robot configuration. If all joint values for all three chains are "logical" for a given pose in a given robot, then that pose is within the workspace. If one or more joint values are illogical, then the pose is outside of the workspace. A testing algorithm that distinguishes poses with "good" joint values from poses with "bad" joint values can then be used to calculate workspace. The algorithm follows the following sequence.

1. User "builds" robot by fixing joint lengths, angles, base links, and offset angles.

2. Test setup: Intervals and test range in X and Y directions are specified. A range of orientation angles (\( \phi \)) is also specified for reachable and dexterous workspace calculations.

3. Joint values for each point are calculated.

4. All nine resulting joint values, three for each independent chain, are tested to determine if each chain can reach the given point within the testing range.

5. If all joint values are "good," then the point with the given orientation is within the workspace. It is saved as being within the workspace. If one or more of the joint values is illogical, then the point is not saved.
6. The saved points are plotted to define the workspace.

The computer program flowchart for determining workspace is found on the following page.
Figure 3.4
Workspace Program Algorithm

User Builds Mechanism

Base Point For Each Chain Chosen

Offset Angle from Center For Each Chain Chosen

Three of Six Variables are Fixed: L1, L2, L3, θ1, θ2, or θ3

Range and Interval of Tested Area Specified

Joint Values Calculated for Each Point Within Range

Imaginary Number (i) added to Negative Lengths and Joints Exceeding Mechanical Limits

Seeded "IF" statements test all nine Joint Values for presence of Imaginary Numbers.

X, Y Coordinates with no Imaginary Numbers are Saved in Array.

Array is Plotted for Workspace Visualization.
3.6 Joint Limits in Workspace Calculation

When studying any robot configuration, it is necessary to consider realistic joint limits. For example, without imposing joint limits, the 3-RPR mechanism has an infinite analytical workspace outside circles of radius $L_{2i}$ around the base points $A_i$. However, it is hard to imagine a robot with a 120 mm diameter hydraulic piston that extends out 1500 meters. Joint limits play a major role in determining the workspace of any manipulator.

Figure 3.5
Integration of Joint Limits

For a given chain, the solution joint values might be: $\theta_1=60^\circ$, $L_1=8$, and $\theta_3=-40^\circ$. $\theta_1$ and $L_1$ are shown in Fig. 3.5. If the mechanical limits of the prismatic joint $L_1$ is 1.5-6, then an analytical solution exists, but the mechanism cannot really reach point $(x,y)$. The workspace programs incorporate joint limits for some chains. If joint values are outside the mechanical limits of a particular joint, then an imaginary number may be added to the joint value to make the solution "illogical."
The result is seen in Fig. 3.5. Prismatic Joint Values below 1.5 and above 6 become complex conjugate. The main workspace program receives the complex conjugate joint value and plots no point although an analytical solution does exist. The same type algorithm may be used with revolute joints.

3.7 Example Workspace Calculations

Matlab programs were written to calculate workspaces for any given three dof, parallel, planar, robot configuration. The first program, "wsmain.m," calculates the workspace for a given orientation. The second program, "ws3dmain.m," calculates the union of all workspaces over a given range of platform orientations to find the reachable workspace. The third program, "wsint.m" calculates the intersection of all workspaces in a given range of platform orientations to find the dexterous workspace.

Example 1:

A randomly-chosen 3-PRP manipulator’s workspace may be calculated using the three workspace programs. Figure 3.6 indicates the workspace of a given manipulator with an orientation of 30 degrees. Figure 3.7 shows the reachable workspace of the manipulator over a range of 0 to 60 degrees. Figure 3.8 shows the dexterous workspace of the manipulator over a range of 0 to 30 degrees.
3-D Reachable Workspace Plot: The reachable workspace is plotted using a three-dimensional plot where the planar region for each increment of $\phi$ is plotted in the x and y
coordinates. The increments of $\phi$ are plotted along the z-axis. Figure 3.7 is merely a view of the three dimensional plot in Fig. 3.9 viewed from above.

**Figure 3.9**
3-D View of 3-PRP Reachable Workspace

Example 2: The workspaces of a RPR-RPR-PRR robot with a given orientation, and a $\pm 30^\circ$ reachable and dexterous workspace are plotted in Figs. 3.10, 3.11, and 3.12.

**Figure 3.10**
Example Workspace of Random Robot

**Figure 3.11**
Example Reachable Workspace of Random Robot
3.8 Summary

Chapter three addresses the difficulty of developing purely analytical solutions for all possible architectures of planar parallel manipulators and the need for a numerical method. It notes that inverse position solutions yield "illogical" joint values when a point \((x, y, \phi)\) is outside the workspace of a given chain. Specific examples of why this occurs are presented. The method of integrating mechanical joint limits into the workspace calculations is also discussed.

The algorithm for a workspace program is then introduced. The tests specific points in a region of the plane and plots points with "logical" joint values. This same algorithm is slightly modified to produce reachable workspace and dexterous workspace calculations. Examples of the program plots are found throughout the chapter.
CHAPTER 4

INVERSE VELOCITY

4.1 Introduction

Once position and workspace analysis is completed, the next step is to determine the velocities of all links and points of interest in the manipulator. One needs to know the velocities in a manipulator, both to calculate the stored kinetic energy, and also as a step on the way to the determination of the manipulator’s accelerations which are needed for dynamic force calculations. Furthermore, analytical velocity expressions may be used in resolved-rate control of a manipulator. This chapter considers the velocities of the end effector, and joint velocities.

Each separate kinematic chain leading to the end effector will have its own specific velocity matrix. These velocity matrices are calculated using the Euler identity. Results are used to calculate the Jacobian and inverse Jacobian matrix of each independent chain as it relates to the velocity of the end-effector (x, y) on the platform. These Jacobians are essential to control the robot using resolved rate control.

4.2 Velocity Calculations using Complex Numbers

In the three-dimensional case, a “velocity propagation” method determines end-effector velocities. However, planar robots operate in a plane. Thus, vectors may be represented using the Euler identity as in Chapter Two.
\[ e^{z+i\theta} = \cos \theta \pm j \sin \theta \quad 4.1 \]

Representing the positions in the seven independent chain as complex numbers allows the velocity matrices and respective Jacobians to be determined quickly in a uniform manner. The derivation of velocity relationships and Jacobian matrices is now presented. Using the Euler identity, the position vector loop-closure equation connecting each fixed pivot to the end-effector for each of the three chains is:

\[
\begin{bmatrix} x \\ y \end{bmatrix} = E = A_i + L_{i1}e^{j\theta_1} + L_{i2}e^{j(\theta_1+\theta_2)} + L_{i3}e^{j(\theta_1+\theta_2+\theta_3)} \quad 4.2
\]

Before taking the derivative of Eq. 4.2, this expression is simplified by using absolute angles.

Up to this point, joint angles have been relative angles. That is, \( \theta_2 \) has been defined as the angle between the link \( L_1 \) and \( L_2 \). Use of absolute angles \( \theta, \alpha, \) and \( \beta \) simplifies the following calculations. \( \theta \) and \( \theta_1 \) both represent the angle between link \( L_1 \) and \( X_0 \). Angles \( \alpha \) and \( \beta \) represent the angles between link \( L_2 \) and link \( L_3 \) and \( X_0 \), respectively, so \( \alpha = \theta_1 + \theta_2 \) and \( \beta = \theta_1 + \theta_2 + \theta_3 \). The relationship between relative and absolute angles is seen below.

**Figure 4.1**

**Absolute Angles Diagram**

![Diagram showing comparison of relative and absolute angles](image)
Equation 4.2 is rewritten using absolute angles as:

\[ E = A_i + L_1 e^{j\theta_1} + L_2 e^{j\alpha_i} + L_3 e^{j\beta_i} \quad 4.3 \]

Since Eq. 4.3 represents end effector position for any chain, taking its derivative yields velocity. In the general case, joints may be prismatic or revolute. As a result, the product rule must be used when taking derivatives to account for possible changing link lengths. The general time derivative for any serial chain is:

\[
\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \dot{E} = L_1 je^{j\theta_1}(\dot{\theta}_1) + (\dot{L}_1) e^{j\theta_1} + L_2 je^{j\alpha_i}(\dot{\alpha}_i) + (\dot{L}_2) e^{j\alpha_i} + L_3 je^{j\beta_i}(\dot{\beta}_i) + (\dot{L}_3) e^{j\beta_i} \quad 4.4
\]

(Note: The second subscript in the terms refer to its respective chain in a given manipulator.) By using Eq. 4.1 and dividing the equation into real and imaginary parts, the velocity in respective x and y directions is obtained. Note that the time derivative of \( L_{3i} = 0 \) because \( L_{3i} \) is always a rigid platform link. Using this method, the general case representations of velocity using absolute angles are:

\[
\begin{align*}
\dot{x} &= -L_{1i} s\theta_i \dot{\theta}_i + \dot{L}_{1i} c\theta_i - L_{2i} s\alpha_i \dot{\alpha}_i + L_{2i} c\alpha_i - L_{3i} s\beta_i \dot{\beta}_i \\
\dot{y} &= L_{1i} c\theta_i \dot{\theta}_i + \dot{L}_{1i} s\theta_i + L_{2i} c\alpha_i \dot{\alpha}_i + \dot{L}_{2i} s\alpha_i + L_{3i} c\beta_i \dot{\beta}_i
\end{align*} \quad 4.5\quad 4.6
\]

In the planar case (not spatial large angle motions), the angular velocity relationship is obtained by differentiation of Eq. 2.5 with respect to time:

\[
\omega_z = \dot{\phi} = \dot{\theta}_1 + \dot{\theta}_2 + \dot{\theta}_3 = \dot{\beta} \quad 4.7
\]

4.3 Jacobians

The Jacobian matrices of each independent chain are determined using the generalized velocity expressions. For a serial robot chain, the Jacobian matrix maps joint
rates with Cartesian rates \( \{ \dot{x}, \dot{y}, \dot{\omega}_z \}^T \). The inverse Jacobian matrix is used to calculate joint velocities given specific Cartesian velocities.

Each individual chain Jacobian is determined from 4.5 and 4.6 by canceling terms that do not apply to a specific chain. For example, the PPR’s Jacobian is derived as follows:

In the PPR chain, \( \theta \) and \( \alpha \) are fixed. The terms involving a change of \( \alpha \) and \( \theta \) with respect to time go to zero. Thus, 4.5 and 4.6 are reduced to:

\[
\begin{align*}
\dot{x} &= \dot{L}_1 c \theta_i + \dot{L}_2 c \alpha_i - L_3 s \beta_i \dot{\beta} \\
\dot{y} &= \dot{L}_1 s \theta_i + \dot{L}_2 s \alpha_i + L_3 c \beta_i \dot{\beta}
\end{align*}
\]

The Jacobian is derived by writing these equations in Matrix form with Eq. 4.7 in matrix form.

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\omega_z
\end{bmatrix} =
\begin{bmatrix}
c \theta & c \alpha & -L_3 s \beta \\
s \theta & s \alpha & L_3 c \beta \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\dot{L}_1 \\
\dot{L}_2 \\
\dot{\beta}
\end{bmatrix}
\]

The 3 x 3 matrix that maps joint velocity into Cartesian velocity in Eq. 4.10 is the PPR’s Jacobian. Each possible chain’s velocity equation is presented below. The Jacobian matrix is always the 3 x 3 matrix mapping joint velocity into Cartesian velocity.

\[
\begin{bmatrix}
\dot{x} \\
\dot{y} \\
\omega_z
\end{bmatrix} =
\begin{bmatrix}
-L_1 s \theta & -L_2 s \alpha & -L_3 s \beta \\
L_1 c \theta & L_2 c \alpha & L_3 c \beta \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
\dot{\theta} \\
\dot{\alpha} \\
\dot{\beta}
\end{bmatrix}
\]
4.4 Determination of Inverse Jacobian for Entire Robot

The seven "absolute angle" Jacobians are simpler than they would be using "relative angles." This allows one to more efficiently calculate the inverse of the
Jacobians. The inverse of these Jacobians, listed below, map end effector velocity into joint velocity:

\[ J^{-1}_{RRR} = \begin{bmatrix}
\frac{ca}{L_1s(\alpha - \theta)} & \frac{sa}{L_1s(\alpha - \theta)} & -L_3s(\alpha - \beta) \\
-c\theta & -s\theta & L_3s(\theta - \beta) \\
0 & 0 & 1
\end{bmatrix} \quad 4.17 \]

\[ J^{-1}_{RPR} = \begin{bmatrix}
s\theta & c\theta & L_3c(\theta - \beta) \\
-L_4 & L_4 & -L_4 \\
0 & 0 & 1
\end{bmatrix} \quad 4.18 \]

\[ J^{-1}_{RPP} = \begin{bmatrix}
0 & 0 & 1 \\
\frac{s\alpha}{s(\alpha - \theta)} & \frac{-c\alpha}{s(\alpha - \theta)} & \frac{L_4c(\alpha - \theta)}{s(\alpha - \theta)} \\
-s\theta & c\theta & -L_4 \\
\frac{s(\alpha - \theta)}{s(\alpha - \theta)} & \frac{s(\alpha - \theta)}{s(\alpha - \theta)} & \frac{s(\alpha - \theta)}{s(\alpha - \theta)}
\end{bmatrix} \quad 4.19 \]

\[ J^{-1}_{PRR} = \begin{bmatrix}
\frac{ca}{c(\theta - \alpha)} & \frac{sa}{c(\theta - \alpha)} & -L_4s(\alpha - \beta) \\
-s\theta & c\theta & -L_4c(\theta - \beta) \\
\frac{L_2c(\theta - \alpha)}{L_2c(\theta - \alpha)} & \frac{L_2c(\theta - \alpha)}{L_2c(\theta - \alpha)} & 1
\end{bmatrix} \quad 4.20 \]

\[ J^{-1}_{PBP} = \begin{bmatrix}
\frac{-sa}{s(\theta - \alpha)} & \frac{ca}{s(\theta - \alpha)} & -L_2 \\
0 & 0 & 1 \\
\frac{s(\theta - \alpha)}{s(\theta - \alpha)} & \frac{s(\theta - \alpha)}{s(\theta - \alpha)} & \frac{L_2c(\alpha - \theta)}{s(\theta - \alpha)}
\end{bmatrix} \quad 4.21 \]
\[ J^{-1}_{PPR} = \begin{bmatrix}
  \frac{s\alpha}{s(\alpha-\theta)} & \frac{-c\alpha}{s(\alpha-\theta)} & \frac{L_3 c(\alpha-\beta)}{s(\alpha-\theta)} \\
  \frac{-s\theta}{s(\alpha-\theta)} & \frac{c\theta}{s(\alpha-\theta)} & \frac{-L_3 c(\beta-\beta)}{s(\alpha-\theta)} \\
  0 & 0 & 1
\end{bmatrix} \]

\[ J^{-1}_{RRP} = \begin{bmatrix}
  \frac{-s\alpha}{L_1 c(\theta-\alpha)} & \frac{c\alpha}{L_1 c(\theta-\alpha)} & \frac{-L_2}{L_1 c(\theta-\alpha)} \\
  \frac{0}{c(\theta-\alpha)} & \frac{0}{c(\theta-\alpha)} & \frac{1}{c(\theta-\alpha)} \\
  \frac{c\theta}{s(\theta-\alpha)} & \frac{s\theta}{c(\theta-\alpha)} & \frac{L_2 s(\alpha-\theta)}{c(\theta-\alpha)}
\end{bmatrix} \]

Again, the Jacobian matrix maps joint velocities into end-effector velocities and its inverse maps end-effector velocities into joint velocities. In the equations below, \( \{\dot{X}\} \)

\( \{\dot{X}\} \) represents the end effector velocity vector \( \{\dot{x}, \dot{y}, \dot{\omega}_z\}^T \) and \( \{\Theta\} \) represents the joint position vector.

\[ \{\dot{X}\} = [J]\{\dot{\Theta}\} \quad 4.24 \]

\[ \{\dot{\Theta}\} = [J^{-1}]\{\dot{X}\} \quad 4.25 \]

The \( i^{th} \) row of the inverse Jacobian represents the mapping of the end-effector's velocity to the \( i^{th} \) actuated joint. Therefore, for any robot configuration, its overall inverse Jacobian (M) may be constructed from the rows corresponding to the actuated joints in each independent chain. For example, in a 3-RRR robot (underlined R indicates that the three revolute joints attached to base link are actuated), the Jacobian equation will be:

\[ \begin{bmatrix}
  \dot{\theta}_{1st\_Chain} \\
  \dot{\theta}_{2nd\_Chain} \\
  \dot{\theta}_{3rd\_Chain}
\end{bmatrix} = \begin{bmatrix}
  Row_{1st\_chain}^{1st\_chain} & \dot{x} \\
  Row_{2nd\_chain}^{2nd\_chain} & \dot{y} \\
  Row_{3rd\_chain}^{3rd\_chain} & \omega_z
\end{bmatrix} \]
Row$1_{1st\_chain}$ represents the first row of the inverse Jacobian for the first chain. Symbolically, it is:

$$Row1_{1st\_Chain} = \begin{bmatrix}
\frac{c\alpha_1}{L_{11}s(\alpha_1 - \theta_1)} & \frac{s\alpha_1}{L_{11}s(\alpha_1 - \theta_1)} & -L_{31}s(\alpha_1 - \beta_1)
\end{bmatrix}$$  \hspace{1cm} (4.27)

Row$1_{2nd\_chain}$ represents the first row of the inverse Jacobian for the second chain. It equals:

$$Row1_{2nd\_Chain} = \begin{bmatrix}
\frac{c\alpha_2}{L_{12}s(\alpha_2 - \theta_2)} & \frac{s\alpha_2}{L_{12}s(\alpha_2 - \theta_2)} & -L_{32}s(\alpha_2 - \beta_2)
\end{bmatrix}$$  \hspace{1cm} (4.28)

Row$1_{3rd\_chain}$ represents the first row of the inverse Jacobian for the third chain.

$$Row1_{3rd\_Chain} = \begin{bmatrix}
\frac{c\alpha_3}{L_{13}s(\alpha_3 - \theta_3)} & \frac{s\alpha_3}{L_{13}s(\alpha_3 - \theta_3)} & -L_{33}s(\alpha_3 - \beta_3)
\end{bmatrix}$$  \hspace{1cm} (4.29)

The resulting inverse Jacobian (M) for a 3-RRR is constructed below.

$$M = \begin{bmatrix}
\frac{c\alpha_1}{L_{11}s(\alpha_1 - \theta_1)} & \frac{s\alpha_1}{L_{11}s(\alpha_1 - \theta_1)} & -L_{31}s(\alpha_1 - \beta_1) \\
\frac{c\alpha_2}{L_{12}s(\alpha_2 - \theta_2)} & \frac{s\alpha_2}{L_{12}s(\alpha_2 - \theta_2)} & -L_{32}s(\alpha_2 - \beta_2) \\
\frac{c\alpha_3}{L_{13}s(\alpha_3 - \theta_3)} & \frac{s\alpha_3}{L_{13}s(\alpha_3 - \theta_3)} & -L_{33}s(\alpha_3 - \beta_3)
\end{bmatrix}$$  \hspace{1cm} (4.30)

The inverse Jacobian is constructed using this method for any manipulator configuration being studied. Again, the row corresponding to the actuated joint in the $i^{th}$ chain's inverse Jacobian is "pulled out" to construct the $i^{th}$ row of the overall robot's inverse Jacobian.
4.5 Effect of Actuation Schemes on Invertibility of Matrix M

The matrix M maps end effector velocity into joint velocity. If the resulting matrix M is invertible, then one may solve the direct kinematics velocity problem for the overall robot. That is, the inverse of the matrix M and the joint velocity vector will determine the end-effector velocity. Symbolically, this statement is represented:

\[
\{ \dot{\mathbf{x}} \} = [M^{-1}](\dot{\theta})
\]  

4.31

M may not be invertible based on a manipulator's actuation scheme. Numerous combinations of three independent chains will produce M matrices with columns of zero. For example, consider a 3-RRR robot (where the 3rd revolute joint of each chain is actuated.) The matrix M would be:

\[
M_{3-RRR} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}
\]

4.32

Clearly, [M] is singular and cannot be inverted.

Various combinations of actuated joints in a robot cause M to become singular by creating columns of zeroes. Note that in all problem cases, the resulting M matrix is identical to 4.32.

Merlet (1996) touches on the issue of actuation. His rule on actuation states that “each joint of a chain may be actuated [provided] the chain obtained when locking the actuated joint is not of the PP type.”

This is not entirely correct. He is correct in stating that a mechanism with three identical "unlocked PP" chains is free to move in the plane. However, other general combinations exist that also allow a "locked mechanism" to translate. That is to say,
based on actuation schemes, certain mechanisms will have more than three degrees of freedom. Again, these schemes are distinguished because the robot's resulting inverse Jacobian (M) is identical to 4.32.

These combinations are created when the matrix M has one or more columns of zeroes. If this is the case, then [M] is not invertible and the mechanism has gained an additional degree of freedom. Manipulators with identical chains will have singular M matrices in the combinations shown in the table below. The first three actuated-chain combinations follow Merlet's rule. The remaining four chains are "additions" to the rule.

Table 4.1

"Uncontrollable Actuation Schemes"

<table>
<thead>
<tr>
<th>RPP</th>
<th>PRP</th>
<th>PPR</th>
<th>RRR</th>
<th>RPR</th>
<th>PRR</th>
<th>RRP</th>
</tr>
</thead>
</table>

In the general case, different chains may be used to create a manipulator. To ensure that a general manipulator has zero mobility when the actuators are locked, it is necessary to ensure that the M matrix created does not have columns of zeroes and is not identical to 4.32. Manipulators with at least one of its three chains not listed in the above table will not gain an extra degree of freedom. For example, an RRR, RRR, RRR-manipulator's M matrix will not have any columns of zeroes. It will not have an extra degree of freedom although two of its independent chains are listed above. Again, as long as no columns of zeroes are created, then a "locked" mechanism will have zero mobility.

Consider the normal case in which all three chains in a given robot are of an identical type. In this case, only fourteen possible actuation schemes exist instead of the eighteen that Merlet identified. In the table below, the actuation schemes that give a manipulator an additional degree of freedom are italicized. The actuation schemes without
any obvious singularity problems are in boldface. (Note that these fourteen cases may
have various singularity conditions, but they are not always singular.)

<table>
<thead>
<tr>
<th>RRR</th>
<th>RRR</th>
<th>RRR</th>
<th>RPR</th>
<th>RPR</th>
<th>RPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>RPP</td>
<td>RPP</td>
<td>PRR</td>
<td>PRR</td>
<td>PRR</td>
<td>PRP</td>
</tr>
<tr>
<td>PRP</td>
<td>PPR</td>
<td>PPR</td>
<td>RPP</td>
<td>RRP</td>
<td>RRP</td>
</tr>
</tbody>
</table>

Table 4.2
Modified Merlet Table

4.6 Velocity Analysis

A Matlab program calculates joint velocities for a manipulator given Cartesian
velocities of the end-effector. Furthermore, the program identifies when a mechanism has
gained an additional degree of freedom.

Figure 4.2
3-RRR Robot

Example 1: The joint velocities of a 3-RRR manipulator with actuated base joints may
be calculated separately using each chain’s inverse Jacobian or using the manipulator’s
inverse Jacobian. If a mechanism’s end effector (see Figure 4.2) is given a velocity of 5
units/second in the x direction, 4 units/second in the y-direction, and an angular velocity of
8 degrees/second, then the following actuated joint velocities are calculated using each
separate chain and the inverse Jacobian (M) of the whole manipulator. The results are identical.

Using M
\[ \dot{\theta}_1 = 0.6294 \]
\[ \dot{\theta}_2 = -0.6919 \]
\[ \dot{\theta}_3 = -0.6249 \]

Using \( J^{-1} \) SEPARATE CHAINS
\[ \dot{\theta}_1 = 0.6294 \]
\[ \dot{\theta}_2 = -0.6919 \]
\[ \dot{\theta}_3 = -0.6249 \]

Again, the inverse of M is the manipulator’s overall Jacobian J. If M is not singular, then M may be inverted and used to map joint velocity into end-effector velocity. The resulting mapped Cartesian velocity matches the original given velocity and is \( x = 5.000, \ y = 4.000, \) and \( \omega_z = 8.000 \). If M is singular, then the Matrix is not invertible and the original velocity may not be calculated. This corresponds to the situation where the robot has gained an additional degree of freedom.

4.7 Resolved-Rate Velocity Control

One method of controlling a manipulator is using resolved-rate control. The method proceeds as follows and is seen in the below figure. In this type of control, the robot receives Cartesian velocity instructions, \( \dot{X} \), for the end-effector. It maps this Cartesian vector into a joint velocity vector using the manipulator’s inverse Jacobian (M in this chapter). The joint velocities are then integrated into joint positions. The controller continuously attempts to satisfy all commanded joint positions.

Two loops run in this closed-loop control system. The inner loop ensures that the joint positions are continuously correct. Incorrect positions cause the controller to move the joints to the appropriate positions. This loop might run ten times faster than the outer loop. The outer loop sends back new joint values to recalculate the inverse Jacobian.
Correct application of this procedure yields the desired end-effector position and orientation as a function of time.

![Resolved-RateControl Block Diagram](image)

Using Matlab, one may simulate an off-line resolved-rate control system. In a program whose output is seen in Fig. 4.4, the user gives an end effector velocity for any robot configuration, and the resulting positions as a function of time are simulated. In the simulation program, each chain's inverse Jacobian is utilized. That is, passive joints also need to be calculated. These passive joints would assume the correct positions without control in an actual manipulator.

In an actual robot, the controller must only send instructions to the three actuators to control the robot. However, a position sensor should be located at each joint if possible, since the robot's inverse Jacobian is a function of its active and passive joint positions.

**Workspace interior singularities:**

Within a given workspace boundary, singularities may exist. Generally, these are caused by two or more axes lining up. (Craig, 1989) These interior singularities are not easily seen in the inverse Jacobian as is the case when a column of zeros exists. When
these singularities exist, they cause one or more joint velocities to go to infinity. In the simulation program “velsim.m,” this phenomenon manifests itself by causing significant errors in joint positions. Thus, the program may also be used to analyze specific paths within a given manipulator’s workspace.

Example 2

In the figure below, simulation results of the resolved-rate control program are shown. In this example, the 3-PRP manipulator is given a velocity vector of \([5 \ 4 \ 25]^T\). The units in the vector are in units per second and degrees per second. The program uses the resolved-rate control algorithm and simulates the movement of the mechanism as a function of time. Clearly, the mechanism is following the specified path.

Figure 4.4
Robot Motion Using Resolved Rate Control
4.8 Summary:

This chapter calculates the analytical velocity equations for the seven possible chains of the parallel planar manipulators being studied. These velocity equations are derived using the Euler method. Each chain’s Jacobian and inverse Jacobian matrices are then derived from these velocity matrices.

A method for deriving the Jacobian and inverse Jacobian matrix for any overall manipulator is then shown. To simplify the process, absolute angles and the Euler method are used to determine the inverse Jacobian of each chain. Then, the “actuated row” of each chain’s inverse Jacobian is used to build the overall manipulator’s inverse Jacobian.

An important result is presented in Section 4.5. Although Merlet identified eighteen possible actuated chains that can form a manipulator, it appears that he did not recognize the fact that various actuation schemes can increase a manipulator’s degrees of freedom so that a mechanism with three locked actuators is still free to move. These actuation schemes cause a manipulator’s inverse Jacobian (M) to be singular under all conditions and thus increase the manipulator’s degree of freedom. Note that a singularity in a parallel manipulator’s Jacobian causes a degree of freedom to be gained instead of lost.
CHAPTER 5

Conclusions and Future Research

This work analyzes the position, workspace, and velocity of all possible architectures of three dof, parallel, planar manipulators from an inverse kinematics viewpoint. Chapter two focuses on developing inverse kinematics position solutions for the seven possible chains. For a given robot architecture with fixed values, all nine joint values may then be determined for a given pose (provided that a solution exists). The Matlab programs simulate results and verify analytical solutions.

Chapter three presents a method to calculate the workspace of all possible robot configurations. Instead of developing analytical solutions for every possible combination, a numerical method is used to test an entire region of the plane for realistic joint values. A testing algorithm plots points within a given mechanism's workspace and does not plot points outside a given robot's workspace. The results of three programs are presented in the chapter and also in the Appendix. These programs calculate any robot's workspace for a given orientation, its reachable workspace for any orientation, and its dexterous workspace for all orientation in a given range.

Chapter four concerns velocity. Jacobian matrices for each chain are developed using the Euler identities. A method of determining a robot's entire inverse Jacobian (M) and Jacobian matrix is also presented. These matrices are then used to simulate resolved-rate control. The results of two programs corresponding to the velocity analysis are presented in the chapter.
An unexpected result arose from Chapter four's velocity analysis. Numerous actuation schemes exist which cause a given manipulator to "automatically" have a singular inverse Jacobian and thus gain an additional degree of freedom.

Throughout this work, computerized analysis has been essential. Simulations have verified analytical solutions, determined workspaces, and brought new insight into how such mechanisms may be actuated. Detailed outputs of the simulation are found in the Appendix.

Future research may draw upon this work to conduct dynamics analysis for all possible configurations. Furthermore, these results should be used for design and control of 3-dof, parallel, planar robots.
References


