Optimal Dimensional Synthesis of Planar Parallel Manipulators with respect to Workspaces

by

Alexander Morrison Hay

Submitted in partial fulfilment of the
requirements for the degree

Philosophiae Doctor (Mechanical Engineering)

in the
Faculty of Engineering, Built Environment and Information Technology,
University of Pretoria, Pretoria

October 2003

Summary

Optimal Dimensional Synthesis of Planar Parallel Manipulators with respect to Workspaces

by

Alexander Morrison Hay

Promoter: **Professor J.A. Snyman**
Department of Mechanical and Aeronautical Engineering

Degree: *Philosophiae Doctor* (Mechanical Engineering)

Parallel manipulators have attracted increasing interest from researchers over the past couple of decades. These manipulators consist of a moving platform, connected to a fixed base by means of a number of separate kinematic chains, placed in parallel. Due to this particular architecture, parallel manipulators possess a number of advantages over traditional serial manipulators. Some of the disadvantages of parallel manipulators, however, are their limited workspaces and nonlinear behavior throughout their workspaces. As a result, development of design methodologies for such manipulators is an important issue in order to ensure performance to their full potential. The methodologies developed in this study are based on the use of numerical optimization techniques.

The development of appropriate design methodologies in this study required three separate issues to be addressed. The first of these was the development, testing and selection of *numerical optimization* algorithms suitable for the solution of the practical optimization problems encountered. Two optimization algorithms, the spherical quadratic steepest descent (SQSD) algorithm[1] for unconstrained problems, and the Dynamic-Q algorithm[2] for constrained problems were developed and tested. These methods compare well with conjugate gradient, and sequential quadratic programming methods respectively, exhibiting robustness and efficiency when applied to a number of test problems.

The second topic addressed is the important issue of the determination of manipulator *workspaces*. The existing chord method for workspace determination is refined, and applied for the first time to the determination of new types of manipulator workspaces for a planar three-degree-of-freedom (3-dof) manipulator. The chord method is also modified for the determination of planar *tendon-driven* parallel manipulator workspaces. A new and efficient method for determining tension distributions in over-constrained tendon-driven manipulators is proposed. The chord method is easily applied to the determination of manipulator workspaces, and determines them accurately and efficiently.

The final issue addressed is that of *dimensional synthesis* of manipulators for prescribed and desired workspaces. Various specific methodologies are investigated and applied to a 2-dof parallel manipulator[3]. The most promising

[1]J.A. Snyman and A.M. Hay, The spherical quadratic steepest descent method for unconstrained minimization with no explicit line searches. *Computers and Mathematics with Applications*, 42:169–178, 2001.

[2]J.A. Snyman and A.M. Hay, The Dynamic-Q optimization method: An alternative to SQP? *Computers and Mathematics with Applications*, 44:1589–1598, 2002.

[3]A.M. Hay and J.A. Snyman, Methodologies for the optimal design of parallel manipulators. Accepted for publication in the *International Journal for Numerical Methods in Engineering*, 2003 (in press).

methodology is then used to optimize a 3-dof planar parallel manipulator[4][5], using various strategies for dealing with the extra angular orientational degree of freedom of the moving platform. An alternative approach is used in optimizing a planar tendon-driven parallel manipulator[6]. The numerical optimization algorithm used in all cases is the Dynamic-Q method, which performs efficiently and robustly in determining optimal designs, even when numerical noise is present in the problem. It is believed that the new methodologies presented provide efficient, practical and easily generalizable numerical alternatives to existing methods for the dimensional synthesis of parallel manipulators.

Keyterms: optimization algorithm, parallel manipulator, optimal design, workspace analysis, mechanism synthesis.

[4]A.M. Hay and J.A. Snyman, The optimal synthesis of parallel manipulators for desired workspaces. In J. Lenarčič and F. Thomas, editors,*Advances in Robot Kinematics*, 337–346, Caldes de Malavella, Spain, June 2002. Kluwer Academic Publishers.

[5]A.M. Hay and J.A. Snyman, The synthesis of parallel manipulators for optimal desired workspaces with respect to the condition number. CD-ROM Proceedings of *ASME 2002 Design Engineering Technical Conferences*, Paper number DETC2002/MECH-34306, Montreal, Canada, October 2002.

[6]A.M. Hay and J.A. Snyman, Analysis and optimization tools for a reconfigurable tendon-driven manipulator. CD-ROM Proceedings of *CIRP 2nd International Conference on Reconfigurable Manufacturing*, Ann Arbor, MI, August 2003.

Samevatting

Optimale Dimensionele Sintese van In-vlak Parallel-manipuleerders met betrekking tot Werkruimtes

deur

Alexander Morrison Hay

Promotor: **Professor J.A. Snyman**

Departement van Meganiese en Lugvaartkundige Ingenieurswese

Graad: *Philosophiae Doctor* (Meganiese Ingenieurswese)

Gedurende die afgelope paar dekades is toenemende belangstelling deur navorsers in parallel-manipuleerders getoon. Hierdie manipuleerders bestaan uit 'n bewegende platform, gekoppel aan 'n vaste basis deur middel van 'n aantal afsonderlike kinematiese kettings, wat in parallel met mekaar geplaas is. As gevolg van hul besondere argitektuur, het parallel-manipuleerders 'n aantal voordele bo tradisionele serie-manipuleerders. Sekere nadele van parallel-manipuleerders is egter, hul beperkte werkruimtes en nie-lineêere gedrag binne die werkruimtes. Gevolglik, is die ontwikkeling van ontwerp-metodologië vir sulke manipuleerders van uiters belang, om te verseker dat hul tot volle potensiaal funksioneer. Die metodologië wat in hierdie studie ontwikkel is, is gebaseer op die gebruik van numeriese optimeringstegnieke.

Die ontwikkeling van gepaste ontwerp-metodologië in hierdie studie, vereis dat drie verskillende sake aangespreek word. Die eerste van hierdie is die ontwikkeling, toetsing en seleksie van numeriese optimerings-algoritmes wat geskik is vir die oplos van die praktiese optimeringsprobleme wat in dié studie voorkom. Twee optimerings-algoritmes, die sferiese kwadratiese steilste daling (SQSD) algoritme[1] vir onbegrensde probleme, en die Dynamic-Q algoritme[2] vir begrensde probleme, is ontwikkel en getoets. Hierdie metodes vergelyk onderskeidelik goed met die toegevoegde gradiënt en agtereenvolgende kwadratiese programmerings (SQP) metodes.

Die tweede belangrike onderwerp wat aangespreek word is die bepaling van manipuleerder werkruimtes. Die bestaande koord-metode vir werkruimtebepaling is verfyn, en vir die eerste keer toegepas in die bepaling van nuwe tipes werkruimtes van 'n in-vlak manipuleerder met 3-vryheidsgrade. Die koord-metode is ook aangepas vir die bepaling van die werkruimtes van 'n in-vlak tendon-aangedrewe parallel-manipuleerder. Die toepassing van die koord-metode lei met gemak tot die doeltreffende en akkurate bepaling van hierdie werkruimtes.

Die finale saak wat bestudeer word is die dimensionele sintese van manipuleerders vir voorgeskrewe en verlangde werkruimtes. Verskeie spesifieke metodologië word ondersoek en toegepas op 'n 2-vryheidsgrade parallel-manipuleerder[3]. Vervolgens is die mees belowende metodologie gebruik in die op-

[1]J.A. Snyman and A.M. Hay, The spherical quadratic steepest descent method for unconstrained minimization with no explicit line searches. *Computers and Mathematics with Applications*, 42:169–178, 2001.

[2]J.A. Snyman and A.M. Hay, The Dynamic-Q optimization method: An alternative to SQP? *Computers and Mathematics with Applications*, 44:1589–1598, 2002.

[3]A.M. Hay and J.A. Snyman, Methodologies for the optimal design of parallel manipulators. Accepted for publication in the *International Journal for Numerical Methods in Engineering*, 2003 (in press).

timering van 'n 3-vryheidsgrade in-vlak parallel-manipuleerder[4,5], waar verskeie strategieë gebruik is om die addisionele hoek-orientasie-vryheidsgraad te hanteer. 'n Alternatiewe benadering is gevolg in die optimering van die tendon-aangedrewe in-vlak parallel-manipuleerder[6]. In al die gevalle is die Dynamic-Q numeriese optimerings-algoritme gebruik. Dié metode is doeltreffend en betroubaar, selfs wanneer numeriese geraas in die probleem teenwoordig is. Die vertroue is dat die nuwe metodologië wat hier aangebied word, doeltreffende, praktiese en maklik veralgemeende numeriese alternatiewe tot bestaande metodes vir dimensionele sintese van parallel-manipuleerders, verteenwoordig.

Sleutelterme: optimerings-algoritme, parallel-manipuleerder, optimale ontwerp, werkruimte-analise, meganisme-sintese

[4] A.M. Hay and J.A. Snyman, The optimal synthesis of parallel manipulators for desired workspaces. In J. Lenarčič and F. Thomas, editors,*Advances in Robot Kinematics*, 337–346, Caldes de Malavella, Spain, June 2002. Kluwer Academic Publishers.

[5] A.M. Hay and J.A. Snyman, The synthesis of parallel manipulators for optimal desired workspaces with respect to the condition number. CD-ROM Proceedings of *ASME 2002 Design Engineering Technical Conferences*, Paper number DETC2002/MECH-34306, Montreal, Canada, October 2002.

[6] A.M. Hay and J.A. Snyman, Analysis and optimization tools for a reconfigurable tendon-driven manipulator. CD-ROM Proceedings of *CIRP 2nd International Conference on Reconfigurable Manufacturing*, Ann Arbor, MI, August 2003.

For the Sun and the Rain...

Acknowledgements

I am indebted to Prof. Jan for his guidance, wisdom and the many things learnt from him, which have not only been restricted to the subjects of optimization and parallel manipulators. It has been a privilege to work with such an excellent mentor and researcher.

I would like to thank my parents for their encouragement and generous financial support, without which I could not have pursued my academic interests. Assistance was also gratefully received from the National Research Foundation and the Mellon Foundation.

Contents

List of Figures

List of Tables

List of Tables

List of Algorithms

Nomenclature

In this work vectors will be denoted in bold font. The following notation will be used:

$\Re^n$	n-dimensional Euclidean (real) space
x, y, z	Cartesian reference frame axes
$\mathbf{q}$	Manipulator generalized coordinates
$\mathbf{u}$	Manipulator output coordinates
$\mathbf{v}$	Manipulator input coordinates
$\mathbf{w}$	Manipulator intermediate coordinates
ϕ_P	Moving platform orientation
$W^C[\phi^{\text{fix}}]$	Constant [angular] orientation workspace
W^M	Maximal workspace
$W^D[\phi^{\min}, \phi^{\max}]$	Dextrous workspace
$W^O[\mathbf{u}^{\text{fix}}]$	Orientation workspace
∂W	Workspace boundary
$\mathbf{x}$	Vector of design variables
f	Objective function
$\mathbf{g}$	Vector of inequality constraints
$\mathbf{h}$	Vector of equality constraints
ε_x	Convergence tolerance on step size
ε_g	Convergence tolerance on gradient value
ε_f	Convergence tolerance on function value

ρ	Move limit for SQSD and Dynamic-Q algorithms
Γ	Finite difference interval
d	Chord length for chord workspace determination method
$\mathbf{d}$	Vector of manipulator design variables
$\mathbf{b}^i$	Point on workspace boundary
$\mathbf{B}^j$	Bifurcation point on workspace boundary
W_p	Prescribed workspace
W_c	Calculated workspace
δW_p	The part of workspace W_p not intersecting W_c
δW_c	The part of workspace W_c not intersecting W_p
$\mathbf{O}'$	Position of local coordinate system $x' - y'$
(β_p, r_p)	Polar coordinate description of prescribed workspace
(β_c, r_c)	Polar coordinate description of calculated workspace
l_i	Length of manipulator actuator (leg) i
$\mathbf{J}$	Manipulator Jacobian matrix
κ	Condition number of the Jacobian matrix
t_i	Tension in tendon i
$\mathbf{S}$	Manipulator structure matrix
$\mathbf{T}$	Transformation matrix from local to global coordinate systems
$\mathbf{f}^{Ci}$	Force transferred to the moving platform by tendon i
$\boldsymbol{\ell}^i$	Displacement vector along tendon i
$\mathbf{f}^P$	External load applied to the moving platform
τ^P	External torque applied to the moving platform

Chapter 1

Introduction: Overview of parallel manipulators and literature review

1.1 Introduction

A parallel manipulator can be defined as

> a closed-loop kinematic chain mechanism whose end-effector is linked to the base by several independent kinematic chains (Merlet [1]).

Parallel manipulators have been increasingly studied and developed over the last couple of decades (Merlet [2], Dasgupta and Mruthyunjaya [3]) from both a theoretical viewpoint as well as for practical applications. Parallel structures are certainly not a new discovery, however advances in computer technology and development of sophisticated control techniques, amongst other factors, have allowed for the more recent practical implementation of

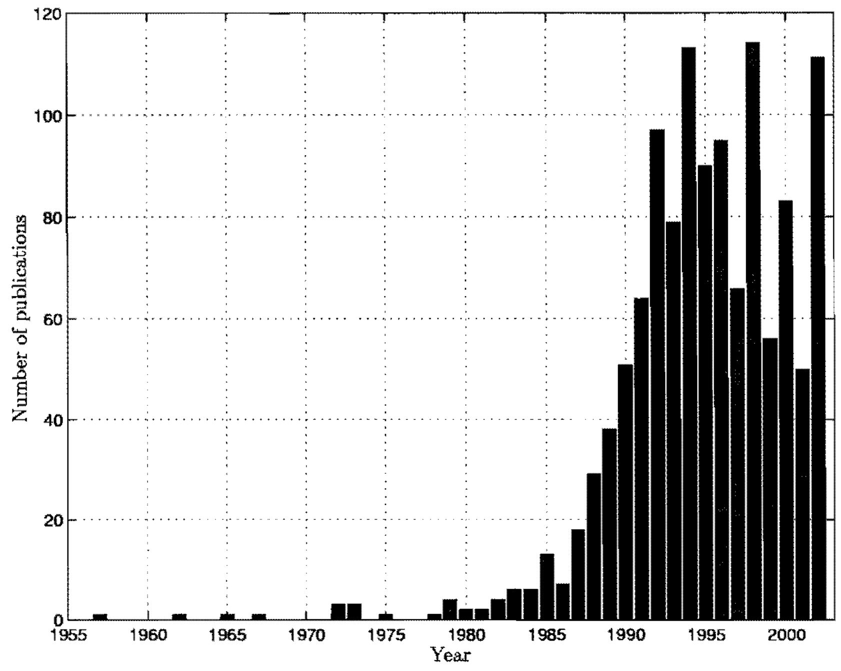

Figure 1.1: Parallel manipulator publications by year 1955-2002

parallel manipulators. This trend is well illustrated by the ever increasing number of publications dedicated to parallel manipulators. Figure 1.1 shows the approximate annual numbers of publications related to parallel manipulators for the past 50 years as reported by Merlet [4].

Interest in parallel manipulators has been stimulated by the advantages offered over traditional serial manipulator architectures. In fact, there exists an interesting duality between parallel and serial architectures, both in terms of analysis and performance, where parallel manipulators have good characteristics in areas where serial manipulators perform poorly, and vice versa. Zamanov and Sotirov [5], Waldron and Hunt [6] and Duffy [7] seek to explain this duality and Fichter and MacDowell [8] discuss some of the practical issues relating to performance of serial and parallel robots.

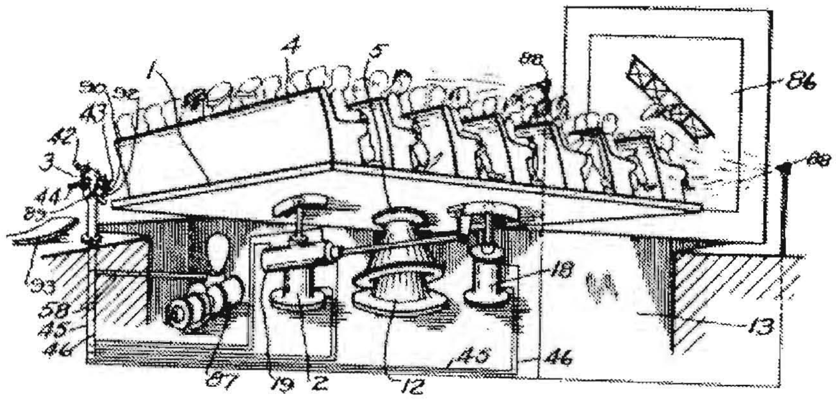

Figure 1.2: The spherical parallel mechanism patented by Gwinnett

The literature review presented in this chapter provides an overview of the development of parallel manipulators, workspace determination, optimal design, characterization of manipulator performance, and optimization methods. This selective overview is used in Section 1.6 to motivate the current study.

1.2 Brief history of parallel manipulator development

Some theoretical problems associated with parallel structures were mentioned by the English architect Sir Christopher Wren as early as the 17th century. Cauchy, Lebesgue, Bricard and Borel all published papers on problems related to parallel mechanisms in the 19th and early 20th century (Merlet [1]).

It appears that the first practical application for a parallel manipulator was proposed by Gwinnett [9] who was granted a patent in 1931 for a motion platform, based on a spherical parallel mechanism (Bonev [10]). As illustrated in Figure 1.2, the motion platform was intended for use in the entertainment industry.

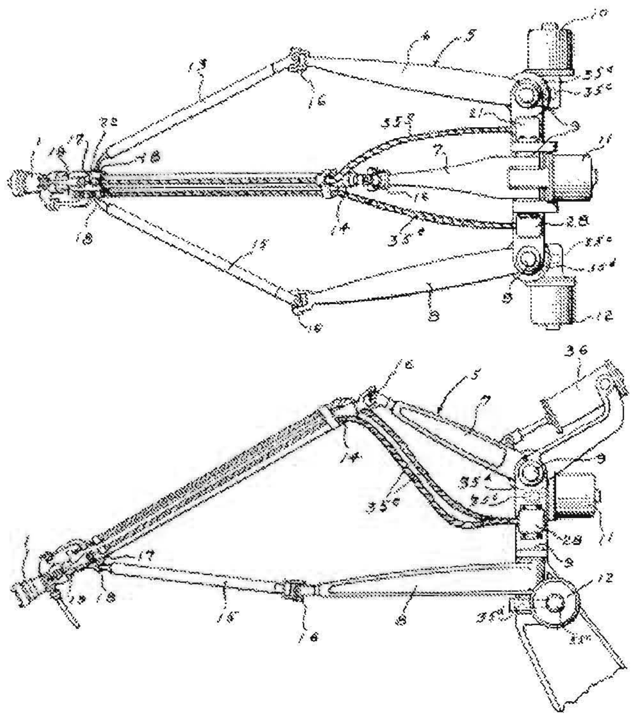

Figure 1.3: Pollard's spatial industrial robot

In 1942 a patent was issued to Pollard [11] for what is now known as the first industrial parallel robot design. The robot, shown in Figure 1.3, was intended for spray painting, but was never built.

In 1965 Stewart [12], published a paper in which he proposed a six-degree-of-freedom (six-dof) parallel platform for use as a flight simulator. This paper attracted so much attention that many researchers began referring to octahedral hexapod parallel manipulators as "Stewart platforms". It is somewhat ironic though that similar ideas to Stewart's had already been independently conceived by two other researchers.

Eric Gough, an employee of the Dunlop Rubber Co., England, had constructed an octahedral hexapod in 1954 (Gough [13], see communications from [12]). This parallel manipulator was used as a universal tyre testing machine as shown in Figure 1.4. Interestingly, Bonev [10] notes that this

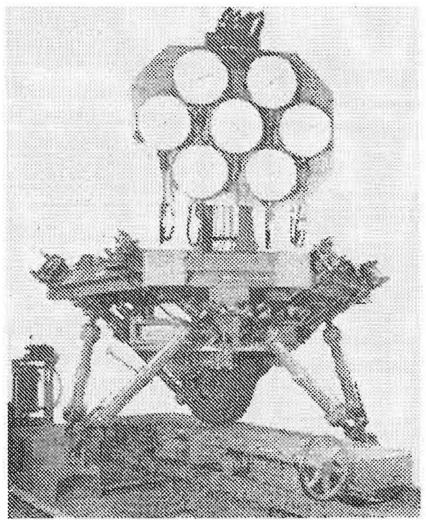

Figure 1.4: The universal tyre-testing machine of Gough

machine continued to operate until 1999.

At the same time an American engineer, Klaus Cappel, was also independently developing an octahedral hexapod manipulator. A patent for an octahedral hexapod to be used as a motion simulator was filed in 1964 and issued in 1967 [14]. The first ever flight simulator based on a octahedral hexapod was made under licence from this patent. Figure 1.5 shows a drawing taken from Cappel's patent.

Since these early days, parallel manipulators have proliferated and found application in many fields. One of the most promising applications is in the manufacturing industry. Prominent early examples of machine tools based on parallel architectures are the Giddings and Lewis Variax and the Ingersoll Octahedral Hexapod which were both first presented at the 1994 International Manufacturing Technology Show (IMTS) in Chicago. More recent and successful applications of parallel manipulator architectures have been the Z^3 machining head developed by DS Technologie Gmbh (DST) which is shown in Figure 1.6 and the Neos Tricept. Other applications include flight simulators, fine positioning devices, overhead cranes (when using cable-driven manipulator architectures) and more recently in medical applications as surgical robots.

1.3 Workspace determination of parallel manipulators

The *workspace* of a manipulator may loosely be defined as

> regions [in output space[1]] which can be reached by a reference point located on the mobile platform [of the manipulator] (Merlet *et al.* [15]).

Based on this definition, the workspace of any manipulator has the same dimension as the number of output degrees of freedom of the manipulator. For example, the workspace of a planar parallel manipulator manipulator, which has three degrees of freedom (translations x and y in the plane and rotation ϕ about the out of plane axis) is thus most fully represented three-dimensionally, with two axes used to represent the x and y positional coordinates of the reference point, and the third axis corresponding to the angular orientation ϕ of the moving platform.

Similarly, the workspace of the Cappell, Gough and Stewart platforms (see the previous section) can only really be fully described in six-dimensional space. Of course, it is difficult for us to conceptualize any space of dimension greater than three. In order to obtain descriptions of manipulator positioning capabilities that can be easily visualized and understood, subsets of the full workspace are defined for which restrictions are placed on some of the output degrees of freedom of the manipulator, most commonly on the orientation

[1]output space – positional plus angular orientational dimensions

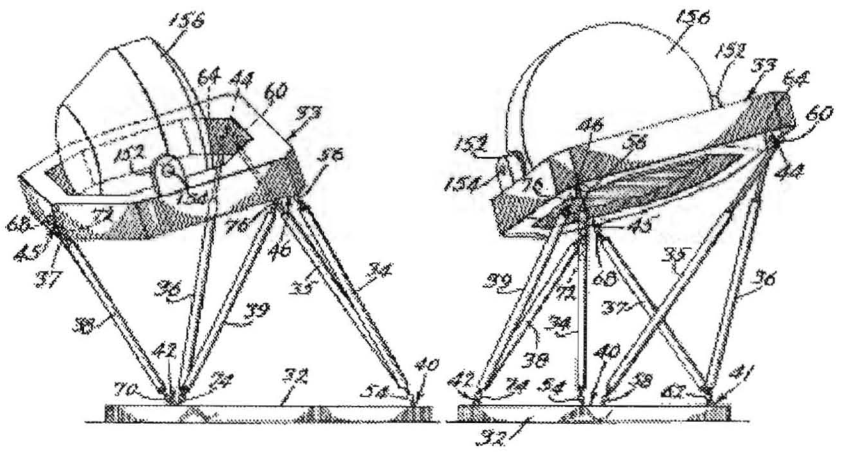

Figure 1.5: Octahedral hexapod motion simulator by Cappel

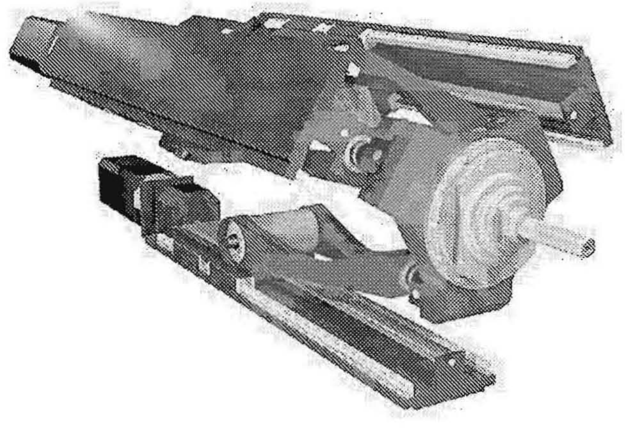

Figure 1.6: The Z^3 machining head by DST

of the moving platform. The various types of workspace commonly used are defined in Section 1.3.1.

Determination of these workspaces for parallel manipulators poses a challenging problem (Merlet *et al.* [15]). Various methods proposed by different researchers are discussed in Section 1.3.2.

1.3.1 Classification of workspace types

Constant orientation workspace

A specific *constant [angular] orientation workspace* is defined as (Merlet *et al.* [15])

> the positional region which can be reached by the reference point of the manipulator when the mobile platform has a specific prescribed constant [angular] orientation.

The constant orientation[2] workspace has the same dimension as the number of translational [positional] output degrees of freedom of the parallel manipulator.

For the planar manipulators studied here, the constant orientation workspace is two-dimensional, and will be denoted as $W^C[\phi^{\text{fix}}]$, where ϕ^{fix} is the specific prescribed and fixed angular orientation of the moving platform associated with that particular constant orientation workspace.

Maximal workspace

The definition of the *maximal workspace* adopted here is

[2]Hereafter 'orientation' will be considered synonymous with 'angular orientation'.

> the positional region which can be reached by the reference point of the manipulator with no restrictions on the orientation of the moving platform.

This is in agreement with the definition offered by Merlet *et al.* [15]. Haug *et al.* [16] also refer to maximal workspaces as "accessible output sets". Essentially, the maximal workspace can be thought of as either the projection of the full output workspace onto the positional space of the manipulator, or the *union* of all possible constant orientation workspaces.

For planar manipulators, the maximal workspace, denoted W^M, is two dimensional and for spatial manipulators, three-dimensional.

Dextrous workspace

The *dextrous workspace* of a manipulator is

> the region reachable by the reference point of the manipulator with all orientations in a given set $[\boldsymbol{\phi}^{\min}, \boldsymbol{\phi}^{\max}]$.

This terminology in consistent with that used by Haug *et al.* [17] and Du Plessis and Snyman [18], but differs slightly from that used by Merlet *et al.* [15] who use the term "total orientation workspace" to describe this workspace, and the term "dextrous workspace" for the special case where the moving platform is required to reach *all* possible orientations.

Once again the dextrous workspace is two-dimensional for the planar case and three-dimensional for the spatial case. The dextrous workspace is denoted $W^D[\phi^{\min}, \phi^{\max}]$ here for the planar case, and can also be thought of as the *intersection* of all constant orientation workspaces in the *orientation interval* $[\phi^{\min}, \phi^{\max}]$.

Orientation workspace

Finally a specific *orientation workspace* is defined as

> the set of angular orientations (orientational region) which can be attained by the moving platform for a fixed position of the reference point.

The orientation workspace is difficult to represent for a general spatial manipulator. Merlet [19] notes that simply plotting the standard Euler angles does not lead to intelligible results. He suggests mapping instead the positions which can be reached by a unit vector, fixed to the moving platform, onto a unit sphere. Bonev *et al.* [20] suggest the use of modified Euler tilt-and-torsion angles, which result in a compact and intuitive representation of the orientation workspace. For the planar manipulator, the orientation workspace is one-dimensional and is denoted $W^O[\mathbf{u}^{\text{fix}}]$, where $\mathbf{u}^{\text{fix}}$ is a vector containing the fixed position of the reference point.

1.3.2 Methods for workspace determination

In general, determination of parallel manipulator workspaces poses a more challenging problem than for serial manipulators. This is because of the strong coupling of the positional and orientational capabilities of parallel manipulators. Merlet [1] gives the example of a six-dof serial robot with a concurrent axis wrist. For this manipulator the three-dimensional volume, which the robot can reach, depends only on the motion capability of the first three actuated joints, while the orientational ability depends only on the last three joints. Compare this to a hexapod, where orientational and positional ability are influenced simultaneously by all the actuators. The most prominent and commonly-used methods for workspace determination can be grouped into four categories.

Geometrical methods

Geometrical methods are based on the observation that the workspace boundary must necessarily always be associated with a physical limit on the manipulator input kinematic chains. By separately taking into account the constraints on each input kinematic chain and geometrically determining the region which can be reached by reference point under these conditions, and then determining the intersection of all these regions, the actual workspace of the manipulator can be determined.

The method as applied to parallel manipulators was first introduced by Gosselin and Angeles [21] who used a geometrical approach to determine constant orientation workspaces of a planar 3-$\underline{R}RR$ parallel manipulator. In this notation, the number signifies the number of kinematic chains linking the moving platform to the base, and the set of letters defines the sequence of joints used in each kinematic chain. A revolute joint is denoted by R and a prismatic joint by P. Spatial universal and spherical joints are respectively denoted by U (or sometimes RR) and S. Actuated joints are indicated by underlining.

The geometrical methodology, including the effects of passive joint limits, is applied to 3-$R\underline{P}R$ planar manipulators in Gosselin and Jean [22]. Merlet *et al.* [15] extend the methodology to determining other types of workspaces of planar 3-$R\underline{P}R$ parallel manipulators (see Section 1.3.1).

Geometric methods have also been used to determine constant orientation workspaces of more complex 6-$U\underline{P}S$ spatial manipulators by Gosselin [23]. The effects of passive joint limits and links interference are included in the constant orientation methodology by Merlet [24, 1]. Constant orientation workspaces of other types of six-dof parallel manipulators including the 6-$\underline{P}US$ (Bonev and Ryu [25]) and 6-$\underline{R}US$ (Bonev and Gosselin [26]) parallel manipulators have also been determined by means of the geometrical method.

A hybrid geometrical-numerical method for determining the orientation workspaces of 6-$U\underline{P}S$ parallel manipulators has been proposed by Merlet [19]. This problem is also addressed by Huang *et al.* [27], who propose a closed form solution to the problem.

Bajpai and Roth [28] and Williams and Reinholtz [29] also use geometrical reasoning to determine workspaces of specific classes of manipulators.

Geometrical methods represent the most efficient and accurate methods for workspace determination available, since the workspace boundary is expressed in analytical terms. It is evident however that the various geometric method implementations are specific to distinct classes of manipulators. At present there exist no direct geometric methods for determining maximal and dextrous workspaces of spatial parallel manipulators.

Continuation methods

A broadly applicable method for workspace analysis using continuation methods has been presented by Jo and Haug [30]. In this method, manipulator workspace boundaries are defined as the sets of points for which the Jacobian matrix of the kinematic constraints are row rank deficient. A continuation method is then used to trace the family of one-dimensional trajectories which correspond to the workspace boundary. When determining parallel manipulator workspaces using this methodology, it becomes necessary to use a slack variable formulation to represent the unilateral constraints implied by physical limits to joint motions (Jo and Haug [31]). This is one of the limitations of the continuation method: that the introduction of other constraints limiting the workspace, such as limits on the passive joints, and link interferences, lead to a very large manipulator Jacobian, which in turn may render the procedure difficult to manage (Merlet [1]).

Jo and Haug [31] use the continuation method to determine maximal workspaces of a 3-$R\underline{P}R$ planar manipulator, and constant orientation workspaces of a 6-$S\underline{P}S$ spatial manipulator. Further developments by Haug *et al.* [16] result in the determination of maximal workspaces of 6-$S\underline{P}S$ spatial parallel manipulators. In Haug *et al.* [32] it is shown that the continuation method also provides an effective tool for determining barriers to control within the manipulator workspace.

The determination of dextrous workspaces of 3-$R\underline{P}R$ and 6-$S\underline{P}S$ parallel manipulators (Haug *et al.* [17]) has also been addressed by means of the continuation method. These authors motivate their approach by stating that "numerical methods are required for constructing boundaries of dextrous workspaces [of more complex spatial manipulators]", an assertion that is borne out by the fact that there are at current no analytical methods available for solving this problem.

Continuation methods have also been applied to the problem of determining *operational envelopes*, or the set of points which can be occupied by *all* points on the working body of the manipulator (Haug *et al.* [33], Adkins and Haug [34]). This problem is important in order to avoid interference between the working body and its surroundings. An associated problem is the determination of domains of interference between working bodies and their surroundings (Haug *et al.* [35, 36]).

An overview of continuation methods applied to determination of workspaces, operational envelopes, and domains of interference is given by Haug *et al.* [37].

Discretization methods

Although computationally expensive, discretization methods represent an easy and stable method for workspace determination. A large variety of implementations are found in the literature. One approach (Yang and Lee [38], Sorli and Ceccarelli [39], Cervantes-Sánchez and Rendón-Sánchez [40]) is to vary the manipulator *input* parameters discretely between their limits, and plot the points reached by the reference point. This approach can provide some insight into the effects of design parameters on the manipulator workspace (Ceccarelli and Sorli [41]), but produces results which can be difficult to interpret, and requires the solution of the forward kinematics of the parallel manipulator.

Since the inverse kinematics are easy to solve for a parallel manipulator, seemingly a better approach is to discretize the *output* space of the manipulator, and then determine whether or not each discrete point belongs to the workspace by solving the inverse kinematics and evaluating at that point the various constraints acting on the manipulator. Approaches based on this idea have been used by Fichter [42], Lee and Shah [43], Masory and Wang [44], Arai *et al.* [45], Stamper *et al.* [46] and Wang *et al.* [47] for determining constant orientation and maximal workspaces of various manipulators. Bonev and Ryu [25] propose a discretization method for determining orientation workspaces of 6-$U\underline{P}S$ manipulators.

The main criticism of discretization methods is their exponential increase in computational expense as the required accuracy is increased.

Optimization methods

Of the methods discussed in the previous sections, geometric methods are highly efficient and accurate, but require a specific formulation for each manipulator class and workspace type. At the other extreme discretization methods are computationally intensive resulting in limited accuracy, but can easily be applied to almost any manipulator. Numerical continuation methods lie somewhere between these two approaches, but including all the constraints acting on the manipulator, and the fact that all internal boundary curves are also mapped, can make the method difficult to implement for more complex manipulators.

As an alternative efficient *numerical* approach for workspace determination, optimization methods have been proposed by Snyman *et al.* [48]. The basic philosophy of the optimization approach is to define the workspace boundary in terms of a constrained optimization problem, where the constraints relate to various physical conditions which limit the workspace of the manipulator. A numerical optimization algorithm is used to solve the optimization problem in a number of search directions to obtain a representation of the workspace boundary. There are two specific implementations of the optimization approach, the *ray* method and the *chord* method. These two methods are distinguished from each other by the search geometry used in determining the successive discrete points along the workspace boundary.

The original ray method of Snyman *et al.* [48] determines the points of intersection of a pencil of rays emanating from a fixed radiating point with the workspace boundary. One deficiency of the the ray method is that it cannot be used to map non-convex manipulator workspaces. The modified ray method (Hay and Snyman [49]) addresses this problem by using user interaction together with the original ray method to map sections of the workspace which cannot be mapped automatically. The alternative chord method of Hay and Snyman [50] can be used to determine non-convex manipulator workspaces automatically. From an initial point on the workspace boundary, the chord method uses fixed radius arc searches to determine successive points until closure of the boundary is obtained.

Previous applications of the optimization approach have focussed on the determination of constant orientation (ray methodology - Du Plessis and Snyman [18]) and maximal workspaces (ray method - Snyman *et al.* [48]; modified ray method - Hay and Snyman [49]; chord method - Hay and Snyman [50]) of planar $3\text{-}R\underline{P}R$ and spatial $6\text{-}U\underline{P}S$ manipulators. An efficient indirect method for determining dextrous manipulator workspaces of these same manipulators by calculating the intersection of various constant orientation workspaces has also been proposed by Du Plessis and Snyman [18].

In Section 5.5 of this work a *direct* method for determining dextrous planar manipulator workspaces is given.

In this study, the chord method is used as the method of choice for determining manipulator workspaces when using the optimization approach. A general overview of the chord methodology is given in Appendix C. A detailed presentation and applications of the ray methodology may be found in Snyman *et al.* [48] and Du Plessis [51]. The modified ray and chord methods are further discussed in Hay and Snyman [49, 52] and Hay [53].

An optimization approach similar to the ray method has also been suggested by Wang and Hseih [54].

1.4 Optimal design of parallel manipulators

As already mentioned parallel manipulators possess a number advantages over traditional serial manipulators (Merlet [1]). Parallel manipulators are, however, difficult to design due to their highly nonlinear and often non-intuitive behavior. An effective and systematic way of addressing the problems stated above is through the use of optimization techniques in the design process. Depending on the particular application, certain manipulator performance criteria may be of more importance than others. Such criteria include design so that the manipulator can reach a certain prescribed workspace, design for optimum velocity, force or error transmission ratios between the actuators and the moving platform, stiffness, isotropy, dynamic behavior or dexterity of the manipulator throughout the workspace.

A distinction can be made between the types of problem studied in the current literature in terms of whether or not workspace requirements are included in the optimization. In the next section optimization purely with respect to some performance measure is discussed. The different synthesis problems, which explicitly include requirements on the workspace, are discussed in Section 1.4.2.

1.4.1 Optimization of performance

This first type of problem involves optimizing the *performance* of the manipulator with respect to some performance measure without explicit consideration of the workspace. The various performance measures commonly used in optimizing manipulators are listed below.

Stiffness

Bhattacharya *et al.* [55], investigate the effect of design parameters on the stiffness of a 6-$U\underline{P}S$ manipulator. Since these authors consider only two variables, the optimization is performed by plotting various stiffness measures as functions of the design parameters and then selecting the most appropriate design by inspection of these graphs. A method for synthesizing a manipulator with respect to link and joint stiffnesses so that the end effector has a desired stiffness is suggested by Chakarov [56]. Hayward *et al.* [57] notice the importance of manipulator stiffness in designing a parallel mechanism-based hand controller.

The optimal design of 3-dof spherical manipulator with respect to both stiffness and conditioning is undertaken by Liu *et al.* [58]. Again, since there are only two design parameters, the optimization is done by inspection of plots of these performance measures against the design parameters. Zhang *et al.* [59] use a genetic algorithm to optimize the stiffness of a 5-dof revolute actuated parallel manipulator with passive constraining leg. Simaan and Shoham [60] determine the configurations of a variable geometry 3-$R\underline{P}R$ planar parallel manipulator which yield a desired stiffness of the end-effector.

Static and dynamic behavior

A mechanism is said to be statically balanced when the weights of the links do not produce any torque (or force) at the actuators under static conditions (Gosselin [61]). Such balancing is achieved through the use of counterweights or springs. Static balancing of various spatial parallel mechanisms has been performed analytically by Wang and Gosselin [62, 63], Gosselin and Wang [64] and Gosselin *et al.* [65]. Herder [66] provides a general discussion on statically balanced parallel mechanisms.

Dynamic behavior is mentioned by Merlet [1] as a measure to be used when quantifying manipulator performances. In many practical applications optimization of acceleration and inertial characteristics may be of importance. Weck and Giesler [67] include dynamic properties in their multi-objective optimization of a 2-$R\underline{P}R$ planar machine tool.

Conditioning of the Jacobian matrix and dexterity

The condition number of the Jacobian matrix of the manipulator can be used as a measure of accuracy of control of the manipulator, or manipulator dexterity. Here the term dexterity refers to a measurement of fine end-effector motion in a local sense (Klein and Blaho [68]). When viewed as a measure of accuracy, the condition number can be thought of as a factor amplifying errors in the actuators, and thus affecting the natural precision of the manipulator. The best conditioning is obtained when the Jacobian matrix is orthogonal, and the manipulator is said to be in an isotropic configuration.

Gosselin and Angeles [21, 69] study planar and spherical 3-$\underline{R}RR$ manipulators and determine, amongst other conditions, designs so that these manipulators are isotropic in their home configurations. Gosselin and Lavoie [70] determine designs of spherical 3-dof manipulators so that they have at least one isotropic configuration. Pittens and Podhorodeski [71] and Zanganeh and

Angeles [72] undertake the isotropic design of 6-$U\underline{P}S$ spatial and 3-$\underline{R}RR$ planar manipulators. It is noted by these authors and by Merlet [1] that it is necessary, if the Jacobian contains both rotational and translational terms, to scale the translational terms by a chosen characteristic length, since the Jacobian is not invariant under any choice of dimensional units. This is a criticism of the use of the condition number as a performance measure in these cases (Merlet [73]).

Of course, isotropy of the Jacobian matrix is a *local*, configuration-dependent property of the manipulator. Gosselin and Angeles [74] propose a global conditioning index (GCI), evaluated over the entire workspace of the manipulator, which they used to optimize the *global* conditioning of parallel 3-dof planar and spherical manipulators using the complex method[3]. The GCI is aimed at obtaining better performance of the manipulator throughout its workspace. Stamper *et al.* [46] and Tsai and Joshi [75] use the global condition index to numerically optimize a spatial 3-dof translational parallel manipulator. Kurtz and Hayward [76] and Leguay-Durand and Reboulet [77] use similar principles in optimizing redundantly-actuated spherical mechanisms. In this case since the number of design variables is low the optimization can be performed graphically. Stoughton and Arai [78] argue against averaging the conditioning over the *entire* workspace and propose instead optimizing the average dexterity over a centralized subregion of the workspace of a 6-$U\underline{P}S$ spatial manipulator. The optimization is performed using the numerical BFGS algorithm.

Other

In Lee *et al.* [79, 80], Zhang and Duffy [81] and Lee *et al.* [82] the concept of using the quality index to determine optimal designs and configurations is proposed and developed for various 3-$R\underline{P}R$ and n-$U\underline{P}S$ manipulators. The

[3]The complex method is a constrained version of the simplex method (Box 1965)

quality index is a measure of proximity to a singularity. Carretero *et al.* [83] minimize the parasitic motion of a 3-$\underline{P}RS$ spatial parallel manipulator using a quasi-Newton optimization method.

1.4.2 Workspace synthesis

The second type of problem is concerned with the *workspace* of the manipulator. Essentially this type of problem, concerned with manipulator synthesis, is the inverse of the analysis problem. Analysis is concerned with determining the workspace of the manipulator for a given design and dimensioning. Synthesis seeks to find the dimensions of the manipulator so that is has a required workspace. Since there is not necessarily a unique solution to this problem, additional requirements are sometimes introduced, where required, to ensure a desired performance of the mechanism as well.

Synthesis with respect to workspace only

Gosselin and Guillot [84] use the complex method to optimize a planar 2-$R\underline{P}R$ parallel manipulator so that the workspace of the manipulator is as close as possible to a prescribed workspace. This methodology is extended by Boudreau and Gosselin [85, 86] and is applied, now using a genetic algorithm, to planar 3-$R\underline{P}R$ and 3-$\underline{R}RR$ manipulators, and a spatial 6-$U\underline{P}S$ manipulator.

Murray *et al.* [87] use a quaternion approach which allows them to determine many 3-$R\underline{P}R$ planar manipulator designs, the workspaces of which include a number of prescribed points. In this approach the set of serial chains (forming the links between the base and moving platform) that can reach the desired poses are first determined. The feasible parallel manipulator designs which can also reach these poses are then assembled from these chains. The quaternion approach is extended to other parallel manipulator types, both planar and spatial in Murray and Hanchak [88], and Perez and McCarthy [89].

Another approach proposed by Merlet [90, 91] is used to determine all spatial manipulator geometries, the workspaces of which include prescribed points or line segments. The method presented takes into account leg length constraints, limits on the passive joint angles, and interference between links. The basic approach is to use each constraint separately to restrict the design variable domain. The region where all constraints are satisfied then corresponds to the set of manipulator designs which can reach the desired poses. Merlet [92, 1] later extends the methodology to include constraints on the articular velocities.

The effects of the design parameters on workspaces of various spatial parallel manipulators is also studied by Ji [93].

Multi-objective optimization

The algorithm for workspace synthesis proposed by Merlet [91] has been extended in Merlet [92], resulting in the DEMOCRAT design methodology. Once the design space has been reduced to all the robot designs, which can reach the required workspace as described in the previous section (referred to by Merlet as the "cutting phase"), the "refining phase" then consists of discretizing this reduced design space, and evaluating robot performances with respect to various performance criteria at each resulting node. The advantage of the DEMOCRAT methodology is its ability to determine *all* possible designs which fulfill the designer's requirements. It would appear though, that as the number of design variables increase, the methodology becomes increasingly more difficult to handle.

Kirchner and Neugebauer [94] have combined many performance criteria in optimizing a spatial manipulator with 13 design variables. These authors determine the Pareto-optimal set with respect to these criteria using a genetic algorithm. Similarly Weck and Giesler [67] perform multi-objective optimization of a planar manipulator to be used in machining applications.

Finally, Gallant and Boudreau [95] propose a method which uses a genetic algorithm for synthesizing planar parallel 3-$R\underline{P}R$ manipulators for a desired workspace, including avoidance of singularities and using the global conditioning index.

1.5 Numerical optimization methods

The sustained increase in computing power has led to numerical optimization techniques becoming more and more popular in many fields, including mechanical engineering. Most current optimization algorithms can be broadly classified as either *deterministic* or *stochastic* methods (Chedmail [96]). The methods discussed in this section are in general for nonlinear optimization problems, since these are the sorts of problems which occur in the mechanism synthesis field.

1.5.1 Deterministic methods

Deterministic methods use knowledge of the local topography of the objective (and constraint) function to travel towards the optimum design. Such methods include classical optimization techniques, line search methods, gradient-based methods and methods such as the simplex method and the method of moving asymptotes. Although many such methods exist, the discussion here is limited to methods which have direct relevance to this study.

Line search methods

The method of *steepest descent* is one of the most fundamental procedures for minimizing a differentiable function of several variables. The method, proposed by Cauchy in the middle of the nineteenth century, continues to be the basis of several gradient-based solution procedures (Bazaraa *et al.* [97], p.300). The performance of the steepest descent method is disappointing, however, compared to other first-order (gradient only) line search methods. In spite of using what appears to be the "best" search direction, i.e. that which gives the maximum rate of decrease at the point of application, the method is not really effective in most problems. The method of steepest descent usually works quite well during the early stages of the optimization process, depending on the point of initialization. However, as a stationary point is approached, the method often behaves poorly, taking small and nearly orthogonal steps. Steepest descent methods are discussed more fully in Section 2.2.

Amongst the methods that use only gradient information and perform successive line searches, the most popular method is probably the *conjugate gradient* method of Fletcher and Reeves [98]. This method generates mutually conjugate directions by taking, at each successive point, a suitable convex combination of the current gradient and the direction used at the previous iteration, as search direction. A slight variation of the Fletcher-Reeves method is the method of Polak and Ribière [99], which is argued to be preferable for non-quadratic functions (Bazaraa *et al.* [97], p.357). Gradient only methods, such as the Fletcher-Reeves method, remain of great importance because they become indispensable when the problem size (number of variables) becomes very large.

Second order methods, using Hessian information and based on Newton's method, have also been proposed. For large numbers of variables the full evaluation of the Hessian matrix, required by Newton's method at each step, becomes time-consuming. In order to avoid this difficulty, *quasi-Newton* methods, which approximate the Hessian matrix by means of an update formula after each step, have been proposed. Two implementations of such quasi-Newton methods are the Davidon-Fletcher-Powell (DFP), and the more recent Broyden-Fletcher-Goldfarb-Shanno (BFGS) methods. These methods exhibit fast convergence. When the number of variables exceed approximately 100, however, attempts to update the Hessian become impractical because of the size of the matrix (Bazaraa *et al.* [97], p.328).

Lagrangian-based methods

In 1760 Lagrange developed the *classical* method for solving equality constrained optimization problems by the introduction of Lagrange multipliers and solution of the resulting unconstrained optimization problem (Snyman [100]). This method can be extended to inequality constrained problems by the use of auxiliary variables. Karush (1939) and Kuhn and Tucker (1951) derived the necessary Karush-Kuhn-Tucker (KKT) conditions, expressed in terms of the Lagrangian, that must be satisfied at the solution of an inequality constrained problem. Analytical determination of the stationary point of the Lagrangian is not always practical, or possible. In order to address this problem, *augmented Lagrange multiplier* methods combine the classical Lagrangian method with a penalty function approach in solving the constrained problem. Here successive approximations to the Lagrange multipliers are used in order to obtain the solution via an iterative procedure.

Sequential quadratic programming (SQP) methods are based on the application of Newton's method to determine the optimum from the KKT conditions of the constrained problem. The method relies on the solution of a quadratic programming problem at each step, in order to determine the next approximate solution and associated Lagrange multipliers. A complete discussion of SQP methods is given in Section 3.2.

The dynamic trajectory method

An alternative optimization algorithm, based on modelling the dynamic trajectory followed by a particle of unit mass in a conservative force field has been proposed by Snyman [101, 102, 103]. This method, called the dynamic trajectory, or Leap-Frog (LfopC) optimization algorithm, has a number of properties which make it suitable for implementation in solving practical engineering optimization problems. A detailed discussion of this method appears in Appendix B.

1.5.2 Stochastic methods

In contrast to deterministic methods, stochastic methods only use gradient information indirectly, and use instead *random* processes for finding new points in the design space. Such methods usually rely on modelling natural phenomena as the basis of the algorithm. Examples of this type of method are the genetic, simulated annealing and particle swarm algorithms.

Genetic algorithms

Genetic algorithms were first introduced by Holland (1965). Their use has subsequently been encouraged by Goldberg [104] and Michalewicz [105]. These algorithms mimic the process of evolution found in nature. From an initial, random population, where each individual is characterized by a specific design vector, subsequent generations are created by “inheriting” features, or parts of the design vector, from the previous generation. The best individuals in each generation are given a better chance of passing on their features to subsequent generations, thus driving the entire population towards an optimum design. Various strategies, such as introducing random perturbations or “mutations” into the design vector of certain individuals, are also employed. Genetic algorithms have gained tremendous popularity due to their high ease of programming, and ability to take into account discrete and continuous design variables (Chedmail [96]). These algorithms are however computationally very demanding, and often require the experimental selection of many optimization parameters in order to obtain good performance for a given problem.

Simulated annealing

Optimization by simulated annealing was proposed by Kirkpatrick *et al.* [106] who credit Metropolis *et al.* [107] for the basic idea. The problem studied by Metropolis and his colleagues was to determine the equilibrium state of a material, composed of a number of particles, by simulating the thermal motion of these particles at a given temperature. In order to use this simulation as a component in an optimization technique, the temperature is used as a control parameter, and under systematic reduction of this, the algorithm asymptotically and statistically converges to the global optimum of the system being optimized. The difficulty associated with such algorithms is that the efficiency of the algorithm, and accuracy of results, are affected by the choice of parameters, such as the rate of decreasing the control temperature. As with genetic algorithms, some initial experimentation is necessary to determine the best settings for a given problem. Many function evaluations are also required in comparison to deterministic methods.

1.6 Motivation for the study

1.6.1 Optimization of parallel manipulators

Some of the advantages offered by parallel manipulators, when *properly designed*, include an excellent load to weight ratio, high stiffness and positioning accuracy and good dynamic behavior (Merlet [2, 1], Fichter and MacDowell [8]). These characteristics are, to a large extent, the result of the load on the platform being distributed more or less equally among the actuators, as opposed to the serial case where the full load is carried by each actuator. In addition, depending on the exact parallel manipulator design, the stress in the actuators is mostly tension or compression, which means that the manipulator can be made very rigid, especially when using linear actuators. In contrast, for a serial manipulator the load is often carried in a cantilever fashion, and the mechanism must be designed to carry the resulting bending loads, often resulting in bulky links. Another factor influencing the accuracy of parallel manipulators is that the positioning accuracy of the end-effector is only slightly affected by errors in the actuators. Errors tend to average in the parallel case, whereas they are cumulative for a serial robot. All of these factors, and the availability of new control and component technologies, have resulted in the increasing popularity of parallel manipulators.

There are, however, also some disadvantages associated with parallel manipulators, which have inhibited their application in some cases. Most serious of these is that the particular architecture of parallel manipulators leads to smaller manipulator workspaces than their serial counterparts. This is due to the additional constraints imposed by the closed kinematic chains of such mechanisms. Parallel manipulators can also be difficult to design (Gosselin *et al.* [108]), since the relationships between design parameters and the workspace, and behavior of the manipulator throughout the workspace, are not intuitive by any means. In addition, parallel manipulator performances are highly dependent on their dimensions. Merlet [73] gives the example that changing the radius of a Gough-Stewart platform by 10% results in a 700% change in the minimal stiffness of the robot. For all of these reasons, Merlet [1] argues that customization of parallel manipulators for each application is absolutely necessary in order to ensure that all performance requirements can be met by the manipulator.

1.6.2 The need for new methodologies

In their recent review paper Dasgupta and Mruthyunjaya [3] survey 214 relevant publications, and at the end of the paper state that, amongst others, the following open problems exist:

1. A detailed and easy-to-use description of the workspace.
2. Workspace synthesis for the Stewart platform.
3. Optimum kinematic synthesis of the Stewart platform for well-conditioned workspace.

These notions are supported even more recently by Merlet, who devotes a keynote address to "the need for a systematic methodology for the evaluation and optimal design of parallel robots" [109]. In a later paper the same author [110] states that "none of [the existing dimensional synthesis methods] are appropriate for parallel robots, which usually have a large number of design parameters".

Of the synthesis methods discussed in Section 1.4.2, genetic algorithm approaches are capable of synthesizing manipulators with large numbers of design variables. These methods are however disadvantaged by their reliance on weighting the contributions of individual performance measures when performing the multi-objective optimization, and the *high computational expense* of these optimization algorithms. Alternative approaches, while efficient, are limited by their need to derive a specific formulation for various manipulator types. In addition, increasing the number of manipulator variables leads to dramatically increasing complexity of the methods. It is thus felt that there is a need for a design methodology, based on *efficient* numerical optimization techniques, which is generally applicable to a variety of manipulator architectures.

1.6.3 Objectives of this study

In an effort to address the points highlighted in the previous section, the following issues are addressed in this study:

1. The development of efficient numerical optimization algorithms capable of handling engineering problems.
2. The development and refinement of numerical methods for workspace determination of various parallel manipulators.
3. The development of alternative numerical methods for manipulator dimensional synthesis, and the investigation of the applicability of the new optimization algorithms, mentioned in 1, to such problems.

This work is split into two parts:

PART I: OPTIMIZATION ALGORITHMS is devoted to the development of new optimization algorithms (item 1 above). Two separate numerical optimization algorithms are presented. The spherical quadratic steepest descent (SQSD) algorithm, presented in **Chapter 2** is intended for unconstrained problems. In **Chapter 3** an optimization algorithm for constrained problems, called the Dynamic-Q algorithm, is presented.

PART II: MANIPULATOR OPTIMIZATION is devoted to workspace determination and development of new methods for manipulator optimization (items 2 and 3 above). In **Chapter 4** various optimization algorithms are applied to the problem of synthesizing a 2-$R\underline{P}R$ planar parallel manipulator. Various forms of the optimization problem statement are developed and evaluated. Building on these results, **Chapter 5** contains application of the methodology to a planar 3-$R\underline{P}R$ manipulator, together with some new developments for dextrous workspace determination of such manipulators. A different class of manipulator, tendon-driven parallel manipulators, are studied in **Chapter 6**. New analysis methods for this class of manipulator are introduced, and optimization of such manipulators is performed. Finally in **Chapter 7** conclusions are drawn from the work performed, and recommendations for future research are made.

Part I

OPTIMIZATION ALGORITHMS

Chapter 2

The spherical quadratic steepest descent algorithm

2.1 Introduction

In this chapter an extremely simple gradient only algorithm is proposed that, in terms of storage requirement (only 3 n-vectors need be stored) and computational efficiency, may be considered as an alternative to the conjugate gradient methods. The method effectively applies the steepest descent (SD) method to successive simple spherical quadratic approximations of the objective function in such a way that no explicit line searches are performed in solving the minimization problem. It is shown that the method is convergent when applied to general positive-definite quadratic functions. The method is tested by its application to some standard and other test problems. On the evidence presented the new method, called the SQSD algorithm, appears to be reliable and stable, and very competitive compared to the well established conjugate gradient methods. In particular, it does very well when applied to extremely ill-conditioned problems.

2.2 The classical steepest descent method

Consider the following unconstrained optimization problem:

$$\min f(\mathbf{x}), \ \ \mathbf{x} \in \Re^n \tag{2.1}$$

where f is a scalar objective function defined on $\Re^n$, the n-dimensional real Euclidean space, and $\mathbf{x}$ is a vector of n real components $x_1, x_2, \ldots, x_n$. It is assumed that f is differentiable so that the gradient vector $\boldsymbol{\nabla} f(\mathbf{x})$ exists everywhere in $\Re^n$. The solution is denoted by $\mathbf{x}^*$.

The steepest descent (SD) algorithm for solving problem (2.1) may then be stated as follows:

Algorithm 2.1 SD algorithm

Initialization: Specify convergence tolerances ε_g and ε_x, select starting point $\mathbf{x}^0$. Set $k := 1$ and go to main procedure.

Main procedure:

1. If $\left\|\boldsymbol{\nabla} f(\mathbf{x}^{k-1})\right\| < \varepsilon_g$, then set $\mathbf{x}^* \cong \mathbf{x}^c = \mathbf{x}^{k-1}$ and stop; otherwise set $\mathbf{u}^k := -\boldsymbol{\nabla} f(\mathbf{x}^{k-1})$.

2. Let λ_k be such that $f(\mathbf{x}^{k-1} + \lambda_k \mathbf{u}^k) = \min_\lambda f(\mathbf{x}^{k-1} + \lambda \mathbf{u}^k)$ subject to $\lambda \geq 0$ {line search step}.

3. Set $\mathbf{x}^k := \mathbf{x}^{k-1} + \lambda_k \mathbf{u}^k$; if $\left\|\mathbf{x}^k - \mathbf{x}^{k-1}\right\| < \varepsilon_x$, then $\mathbf{x}^* \cong \mathbf{x}^c = \mathbf{x}^k$ and stop; otherwise set $k := k + 1$ and go to Step 1.

It can be shown that if the steepest descent method is applied to a general positive-definite quadratic function of the form[1] $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\mathsf{T}\mathbf{A}\mathbf{x} + \mathbf{b}^\mathsf{T}\mathbf{x} + c$, then the sequence $\left\{f(\mathbf{x}^k)\right\} \rightarrow f(\mathbf{x}^*)$. Depending, however, on the starting point $\mathbf{x}^0$ and the condition number of $\mathbf{A}$ associated with the quadratic form, the rate of convergence may become extremely slow.

[1]A superscript T means transpose.

It is proposed here that for general functions $f(\mathbf{x})$, better overall performance of the steepest descent method may be obtained by applying it successively to a sequence of very simple quadratic approximations of $f(\mathbf{x})$. The proposed modification, named here the spherical quadratic steepest descent (SQSD) method, remains a first order method since only gradient information is used with no attempt being made to construct the Hessian of the function. The storage requirements therefore remain minimal, making it ideally suitable for problems with a large number of variables. Another significant characteristic is that the method requires no explicit line searches.

2.3 The spherical quadratic steepest descent method

In the SQSD approach, given an initial approximate solution $\mathbf{x}^0$, a sequence of spherically quadratic optimization subproblems $P[k], k = 0, 1, 2, \ldots$ is solved, generating a sequence of approximate solutions $\mathbf{x}^{k+1}$. More specifically, at each point $\mathbf{x}^k$ the constructed approximate subproblem is $P[k]$:

$$\min_{\mathbf{x}} \tilde{f}_k(\mathbf{x}) \tag{2.2}$$

where the approximate objective function $\tilde{f}_k(\mathbf{x})$ is given by

$$\tilde{f}_k(\mathbf{x}) = f(\mathbf{x}^k) + \boldsymbol{\nabla}^T f(\mathbf{x}^k)(\mathbf{x} - \mathbf{x}^k) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^k)\mathbf{C}_k^T(\mathbf{x} - \mathbf{x}^k) \tag{2.3}$$

and $\mathbf{C}_k = \text{diag}(c_k, c_k, \ldots, c_k) = c_k\mathbf{I}$. The solution to this problem will be denoted by $\mathbf{x}^{*k}$, and for the construction of the next subproblem $P[k+1]$, $\mathbf{x}^{k+1} := \mathbf{x}^{*k}$.

For the first subproblem the curvature c_0 is set to $c_0 := \|\boldsymbol{\nabla} f(\mathbf{x}^0)\| / \rho$, where $\rho > 0$ is some arbitrarily specified step limit. Thereafter, for $k \geq 1$, c_k is chosen such that $\tilde{f}(\mathbf{x}^k)$ interpolates $f(\mathbf{x})$ at both $\mathbf{x}^k$ and $\mathbf{x}^{k-1}$. The latter conditions imply that for $k = 1, 2, \ldots$

$$c_k := \frac{2\left[f(\mathbf{x}^{k-1}) - f(\mathbf{x}^k) - \nabla^T f(\mathbf{x}^k)(\mathbf{x}^{k-1} - \mathbf{x}^k)\right]}{\|\mathbf{x}^{k-1} - \mathbf{x}^k\|^2} \tag{2.4}$$

Clearly the identical curvature entries along the diagonal of the Hessian, mean that the level surfaces of the quadratic approximation $\tilde{f}_k(\mathbf{x})$, are indeed concentric hyper-spheres. The approximate subproblems $P[k]$ are therefore aptly referred to as spherical quadratic approximations.

It is now proposed that for a large class of problems the sequence $\mathbf{x}^0, \mathbf{x}^1, \ldots$ will tend to the solution of the original problem (2.1), i.e.

$$\lim_{k \to \infty} \mathbf{x} = \mathbf{x}^* \tag{2.5}$$

For subproblems $P[k]$ that are convex, i.e. $c_k > 0$, the solution occurs where $\nabla \tilde{f}_k(\mathbf{x}) = \mathbf{0}$, that is where

$$\nabla f(\mathbf{x}^k) + c_k \mathbf{I}(\mathbf{x} - \mathbf{x}^k) = \mathbf{0} \tag{2.6}$$

The solution to the subproblem, $\mathbf{x}^{*k}$ is therefore given by

$$\mathbf{x}^{*k} = \mathbf{x}^k - \frac{\nabla f(\mathbf{x}^k)}{c_k} \tag{2.7}$$

Clearly the solution to the spherical quadratic subproblem lies along a line through $\mathbf{x}^k$ in the direction of steepest descent. The SQSD method may formally be stated in the form given in Algorithm 2.2.

Step size control is introduced in Algorithm 2.2 through the specification of a step limit ρ and the test for $\|\mathbf{x}^k - \mathbf{x}^{k-1}\| > \rho$ in Step 2 of the main procedure. Note that the choice of c_0 ensures that for $P[0]$ the solution $\mathbf{x}^1$ lies at a distance ρ from $\mathbf{x}^0$ in the direction of steepest descent. Also the test in Step 3 that $c_k < 0$, and setting $c_k := 10^{-60}$ where this condition is true ensures that the approximate objective function is always positive-definite.

Algorithm 2.2 SQSD algorithm

Initialization: Specify convergence tolerances ε_g and ε_x, step limit $\rho > 0$ and select starting point $\mathbf{x}^0$. Set $c_0 := \|\nabla f(\mathbf{x}^0)\| / \rho$. Set $k := 1$ and go to main procedure.

Main procedure:

1. If $\|\nabla f(\mathbf{x}^{k-1})\| < \varepsilon_g$, then $\mathbf{x}^* \cong \mathbf{x}^c = \mathbf{x}^{k-1}$ and stop; otherwise set
$$\mathbf{x}^k := \mathbf{x}^{k-1} - \frac{\nabla f(\mathbf{x}^{k-1})}{c_{k-1}}.$$

2. If $\|\mathbf{x}^k - \mathbf{x}^{k-1}\| > \rho$, then set
$$\mathbf{x}^k := \mathbf{x}^k - \rho\frac{\nabla f(\mathbf{x}^{k-1})}{\|\nabla f(\mathbf{x}^{k-1})\|};$$
if $\|\mathbf{x}^k - \mathbf{x}^{k-1}\| < \varepsilon_x$, then $\mathbf{x}^* \cong \mathbf{x}^c = \mathbf{x}^k$ and stop.

3. Set
$$c_k := \frac{2\left[f(\mathbf{x}^{k-1}) - f(\mathbf{x}^k) - \nabla^T f(\mathbf{x}^k)(\mathbf{x}^{k-1} - \mathbf{x}^k)\right]}{\|\mathbf{x}^{k-1} - \mathbf{x}^k\|^2};$$
if $c_k < 0$ set $c_k := 10^{-60}$.

4. Set $k := k + 1$ and go to Step 1 for next iteration.

2.4 Convergence of the SQSD method

An analysis of the convergence rate of the SQSD method, when applied to a general positive-definite quadratic function, affords insight into the convergence behavior of the method when applied to more general functions. This is so because for a large class of continuously differentiable functions, the behavior close to local minima is quadratic. For quadratic functions the following theorem may be proved.

THEOREM. *The SQSD algorithm (without step size control) is convergent when applied to the general quadratic function of the form* $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathsf{T}}\mathbf{x}$, *where* $\mathbf{A}$ *is a* $n \times n$ *positive-definite matrix and* $\mathbf{b} \in \Re^n$.

PROOF. Begin by considering the bivariate quadratic function, $f(\mathbf{x}) = x_1^2 + \gamma x_2^2$, $\gamma \geq 1$ and with $\mathbf{x}^0 = [\alpha, \beta]^{\mathsf{T}}$. Assume $c_0 > 0$ given, and for convenience in what follows set $c_0 = 1/\delta, \delta > 0$. Also employ the notation $f_k = f(\mathbf{x}^k)$.

Application of the first step of the SQSD algorithm yields

$$\mathbf{x}^1 = \mathbf{x}^0 - \frac{\boldsymbol{\nabla} f_0}{c_0} = [\alpha(1-2\delta), \beta(1-2\gamma\delta)]^{\mathsf{T}} \tag{2.8}$$

and it follows that

$$\left\|\mathbf{x}^1 - \mathbf{x}^0\right\|^2 = 4\delta^2(\alpha^2 + \gamma^2\beta^2) \tag{2.9}$$

and

$$\boldsymbol{\nabla} f_1 = [2\alpha(1-2\delta), 2\gamma\beta(1-2\gamma\delta)]^{\mathsf{T}} \tag{2.10}$$

For the next iteration the curvature is given by

$$c_1 = \frac{2[f_0 - f_1 - \boldsymbol{\nabla}^{\mathsf{T}} f_1(\mathbf{x}^0 - \mathbf{x}^1)]}{\|\mathbf{x}^0 - \mathbf{x}^1\|^2} \tag{2.11}$$

Utilizing the information contained in (2.8)-(2.10), the various entries in expression (2.11) are known, and after substitution c_1 simplifies to

$$c_1 = \frac{2(\alpha^2 + \gamma^3\beta^2)}{\alpha^2 + \gamma^2\beta^2} \tag{2.12}$$

In the next iteration, Step 1 gives

$$\mathbf{x}^2 = \mathbf{x}^1 - \frac{\boldsymbol{\nabla} f_1}{c_1} \tag{2.13}$$

And after the necessary substitutions for $\mathbf{x}^1$, $\boldsymbol{\nabla} f_1$ and c_1, given by (2.8), (2.10) and (2.12) respectively, (2.13) reduces to

$$\mathbf{x}^2 = [\alpha(1-2\delta)\mu_1, \beta(1-2\gamma\delta)\omega_1]^T \tag{2.14}$$

where

$$\mu_1 = 1 - \frac{1+\gamma^2\beta^2/\alpha^2}{1+\gamma^3\beta^2/\alpha^2} \tag{2.15}$$

and

$$\omega_1 = 1 - \frac{\gamma+\gamma^3\beta^2/\alpha^2}{1+\gamma^3\beta^2/\alpha^2} \tag{2.16}$$

Clearly if $\gamma = 1$, then $\mu_1 = 0$ and $\omega_1 = 0$. Thus by (2.14) $\mathbf{x}^2 = \mathbf{0}$ and convergence to the solution is achieved within the second iteration.

Now for $\gamma > 1$, and for any choice of α and β, it follows from (2.15) that

$$0 \leq \mu_1 \leq 1 \tag{2.17}$$

which implies from (2.14) that for the first component of $\mathbf{x}^2$:

$$\left|x_1^{(2)}\right| = |\alpha(1-2\delta)\mu_1| < |\alpha(1-2\delta)| = \left|x_1^{(1)}\right| \tag{2.18}$$

or introducing α notation (with $\alpha_0 = \alpha$), that

$$|\alpha_2| = |\mu_1\alpha_1| < |\alpha_1| \tag{2.19}$$

{Note: because $c_0 = 1/\delta > 0$ is chosen arbitrarily, it cannot be said that $|\alpha_1| < |\alpha_0|$. However α_1 is finite.}

The above argument, culminating in result (2.19), is for the two iterations $\mathbf{x}^0 \rightarrow \mathbf{x}^1 \rightarrow \mathbf{x}^2$. Repeating the argument for the sequence of overlapping pairs of iterations $\mathbf{x}^1 \rightarrow \mathbf{x}^2 \rightarrow \mathbf{x}^3$; $\mathbf{x}^2 \rightarrow \mathbf{x}^3 \rightarrow \mathbf{x}^4; \ldots$, it follows similarly that $|\alpha_3| = |\mu_2\alpha_2| < |\alpha_2|$; $|\alpha_4| = |\mu_3\alpha_3| < |\alpha_3|; \ldots$, since $0 \leq \mu_2 \leq 1$; $0 \leq$

$\mu_3 \leq 1; \ldots$, where the μs are given by (corresponding to equation (2.15) for μ_1):

$$\mu_1 = 1 - \frac{1 + \gamma^2 \beta_{j-1}^2 / \alpha_{j-1}^2}{1 + \gamma^3 \beta_{j-1}^2 / \alpha_{j-1}^2} \tag{2.20}$$

Thus in general

$$0 \leq \mu_j \leq 1 \tag{2.21}$$

and

$$|\alpha_{j+1}| = |\mu_j \alpha_j| < |\alpha_j| \tag{2.22}$$

For large positive integer m it follows that

$$|\alpha_m| = |\mu_{m-1}\alpha_{m-1}| = |\mu_{m-1}\mu_{m-2}\alpha_{m-2}| = |\mu_{m-1}\mu_{m-2}\cdots\mu_1\alpha_1| \tag{2.23}$$

and clearly for $\gamma > 0$, because of (2.21)

$$\lim_{m\to\infty} |\alpha_m| = 0 \tag{2.24}$$

Now for the second component of $\mathbf{x}^2$ in (2.14), the expression for ω_1, given by (2.16), may be simplified to

$$\omega_1 = \frac{1-\gamma}{1 + \gamma^3 \beta^2 / \alpha^2} \tag{2.25}$$

Also for the second component:

$$x_2^{(2)} = \beta(1 - 2\gamma\delta)\omega_1 = \omega_1 x_2^{(1)} \tag{2.26}$$

or introducing β notation

$$\beta_2 = \omega_1 \beta_1 \tag{2.27}$$

The above argument is for $\mathbf{x}^0 \to \mathbf{x}^1 \to \mathbf{x}^2$ and again, repeating it for the sequence of overlapping pairs of iterations, it follows more generally for $j = 1, 2, \ldots$, that

$$\beta_{j+1} = \omega_j \beta_j \tag{2.28}$$

where ω_j is given by

$$\omega_j = \frac{1-\gamma}{1+\gamma^3\beta_{j-1}^2/\alpha_{j-1}^2} \tag{2.29}$$

Since by (2.24), $|\alpha_m| \to 0$, it follows that if $|\beta_m| \to 0$ as $m \to \infty$, the theorem is proved for the bivariate case. Make the assumption that $|\beta_m|$ does not tend to zero, then there exists a finite positive number ε such that

$$|\beta_j| \geq \varepsilon \tag{2.30}$$

for all j. This allows the following argument:

$$|\omega_j| = \left|\frac{1-\gamma}{1+\gamma^3\beta_{j-1}^2/\alpha_{j-1}^2}\right| \leq \left|\frac{1-\gamma}{1+\gamma^3\varepsilon^2/\alpha_{j-1}^2}\right| = \left|\frac{(1-\gamma)\alpha_{j-1}^2}{\alpha_{j-1}^2+\gamma^3\varepsilon^2}\right| \tag{2.31}$$

Clearly since by (2.24) $|\alpha_m| \to 0$ as $m \to \infty$, (2.31) implies that also $|\omega_m| \to 0$. This result taken together with (2.28) means that $|\beta_m| \to 0$ which contradicts the assumption above. With this result the theorem is proved for the bivariate case.

Although the algebra becomes more complicated, the above argument can clearly be extended to prove convergence for the multivariate case, where

$$f(\mathbf{x}) = \sum_{i=1}^{n} \gamma_i x_i^2, \ \gamma_1 = 1 < \gamma_2 < \gamma_3 < \ldots < \gamma_n \tag{2.32}$$

Finally since the general quadratic function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^{\mathsf{T}}\mathbf{A}\mathbf{x} + \mathbf{b}^{\mathsf{T}}\mathbf{x}, \ \mathbf{A} \text{ positive} - \text{definite} \tag{2.33}$$

may be transformed to the form (2.32), convergence of the SQSD method is also ensured in the general case.

2.5 Numerical results and conclusion

The SQSD method is now demonstrated by its application to some test problems. For comparison purposes the results are also given for the standard SD method and both the Fletcher-Reeves (FR) and Polak-Ribiere (PR) conjugate gradient methods. The latter two methods are implemented using the CG+ FORTRAN conjugate gradient program of Gilbert and Nocedal [111]. The CG+ implementation uses the line search routine of Moré and Thuente [112]. The function and gradient values are evaluated together in a single subroutine. The SD method is applied using CG+ with the search direction modified to the steepest descent direction. The FORTRAN programs were run on a 266 MHz Pentium 2 computer using double precision computations.

The standard (references [113, 114, 115, 116]) and other test problems used are listed in Appendix A and the results are given in Tables 2.1 and 2.2. The convergence tolerances applied throughout are $\varepsilon_g = 10^{-5}$ and $\varepsilon_x = 10^{-8}$, except for the extended homogenous quadratic function with $n = 50000$ (Problem 12) and the extremely ill-conditioned Manevich functions (Problems 14). For these problems the extreme tolerances $\varepsilon_g \cong 0 (= 10^{-75})$ and $\varepsilon_x = 10^{-12}$, are prescribed in an effort to ensure very high accuracy in the approximation $\mathbf{x}^c$ to $\mathbf{x}^*$. For each method the number of function-cum-gradient-vector evaluations (N^{fg}) are given. For the SQSD method the number of iterations is the same as N^{fg}. For the other methods the number of iterations (N^{it}) required for convergence, and which corresponds to the number of line searches executed, are also listed separately. In addition the relative error (E^r) in optimum function value, defined by

$$E^r = \left| \frac{f(\mathbf{x}^*) - f(\mathbf{x}^c)}{1 + |f(\mathbf{x}^*)|} \right| \tag{2.34}$$

where $\mathbf{x}^c$ is the approximation to $\mathbf{x}^*$ at convergence, is also listed. For the Manevich problems, with $n \geq 40$, for which the other (SD, FR and PR) algorithms fail to converge after the indicated number of steps, the infinite norm of the error in the solution vector (I^∞), defined by $\|\mathbf{x}^* - \mathbf{x}^c\|_\infty$ is also tabulated. These entries, given instead of the relative error in function value (E^r), are made in italics.

Inspection of the results shows that the SQSD algorithm is consistently com-

Prob. #	n	SQSD			Steepest Descent		
		ρ	N^{fg}	E^r	N^{fg}	N^{it}	E^r/I^∞
1	3	1	12	3.E-14	41	20	6.E-12
2	2	1	31	1.E-14	266	131	9.E-11
3	2	1	33	3.E-08	2316	1157	4.E-08
4	2	0.3	97	1.E-15	$>$ 20000		3.E-09
5(a)	3	1	11	1.E-12	60	29	6.E-08
5(b)	3	1	17	1.E-12	49	23	6.E-08
6	4	1	119	9.E-09	$>$ 20000		2.E-06
7	3	1	37	1.E-12	156	77	3.E-11
8	2	10	39	1.E-22	12050*	6023*	26*
9	2	0.3	113	5.E-14	6065	3027	2.E-10
10	2	1	43	1.E-12	1309	652	1.E-10
11	4	2	267	2.E-11	16701	8348	4.E-11
12	20	1.E+04	58	1.E-11	276	137	1.E-11
	200	1.E+04	146	4.E-12	2717	1357	1.E-11
	2000	1.E+04	456	2.E-10	$>$ 20000		2.E-08
	20000	1.E+04	1318	6.E-09	$>$ 10000		8.E+01
	50000	1.E+10	4073	3.E-16	$>$ 10000		5.E+02
13	10	0.3	788	2.E-10	$>$ 20000		4.E-07
	100	1	2580	1.E-12	$>$ 20000		3.E+01
	300	1.73	6618	1.E-10	$>$ 20000		2.E+02
	600	2.45	13347	1.E-11	$>$ 20000		5.E+02
	1000	3.16	20717	2.E-10	$>$ 30000		9.E+02
14	20	1	3651	2.E-27	$>$ 20000		9.E-01
		10	3301	9.E-30			
	40	1	13302	5.E-27	$>$ 30000		*1.E+00*
		10	15109	2.E-33			
	60	1	19016	7.E-39	$>$ 30000		*1.E+00*
		10	16023	6.E-39			
	100	1	39690	1.E-49	$>$ 50000		*1.E+00*
		10	38929	3.E-53			
	200	1	73517	5.E-81	$>$ 100000		*1.E+00*
		10	76621	4.E-81			

* Convergence to a local minimum with $f(\mathbf{x}^c) = 48.9$.

Table 2.1: Performance of the SQSD and SD optimization algorithms when applied to the test problems listed in Appendix A

Prob. #	n	Fletcher-Reeves			Polak-Ribiere		
		N^{fg}	N^{it}	E^r/I^∞	N^{fg}	N^{it}	E^r/I^∞
1	3	7	3	0$	7	3	0$
2	2	30	11	2.E-11	22	8	2.E-12
3	2	45	18	2.E-08	36	14	6.E-11
4	2	180	78	1.E-11	66	18	1.E-14
5(a)	3	18	7	6.E-08	18	8	6.E-08
5(b)	3	65	31	6.E-08	26	11	6.E-08
6	4	1573	783	8.E-10	166	68	3.E-09
7	3	132	62	4.E-12	57	26	1.E-12
8	2	72*	27*	26*	24*	11*	26*
9	2	56	18	5.E-11	50	17	1.E-15
10	2	127	60	6.E-12	30	11	1.E-11
11	4	193	91	1.E-12	99	39	9.E-14
12	20	42	20	9.E-32	42	20	4.E-31
	200	163	80	5.E-13	163	80	5.E-13
	2000	530	263	2.E-13	530	263	2.E-13
	20000	1652	825	4.E-13	1652	825	4.E-13
	50000	3225	1161	1.E-20	3225	1611	1.E-20
13	10	> 20000		2.E-02	548	263	4.E-12
	100	> 20000		8.E+01	1571	776	2.E-12
	300	> 20000		3.E+02	3253	1605	2.E-12
	600	> 20000		6.E+02	5550	2765	2.E-12
	1000	> 30000		1.E+03	8735	4358	2.E-12
14	20	187	75	8.E-24	1088	507	2.E-22
	40	> 30000		*1.E+00*	> 30000		*1.E+00*
	60	> 30000		*1.E+00*	> 30000		*1.E+00*
	100	> 50000		*1.E+00*	> 50000		*1.E+00*
	200	> 100000		*1.E+00*	> 100000		*1.E+00*

* Convergence to a local minimum with $f(\mathbf{x}^c) = 48.9$; \$ Solution to machine accuracy.

Table 2.2: Performance of the FR and PR algorithms when applied to the test problems listed in Appendix A

petitive with the other three methods and performs notably well for large problems. Of all the methods the SQSD method appears to be the most reliable one in solving each of the posed problems. As expected, because line searches are eliminated and consecutive search directions are no longer forced to be orthogonal, the new method completely overshadows the standard SD method. What is much more gratifying, however, is the performance of the SQSD method relative to the well-established and well-researched conjugate gradient algorithms. Overall the new method appears to be very competitive with respect to computational efficiency and, on the evidence presented, remarkably stable.

In the implementation of the SQSD method to highly non-quadratic and non-convex functions, some care must however be taken in ensuring that the chosen step limit parameter ρ, is not too large. A too large value may result in excessive oscillations occurring before convergence. Therefore a relatively small value, $\rho = 0.3$, was used for the Rosenbrock problem with $n = 2$ (Problem 4). For the extended Rosenbrock functions of larger dimensionality (Problems 13), correspondingly larger step limit values ($\rho = \sqrt{n}/10$) were used with success.

For quadratic functions, as is evident from the convergence analysis of Section 2.4, no step limit is required for convergence. This is borne out in practice by the results for the extended homogenous quadratic functions (Problems 12), where the very large value $\rho = 10^4$ was used throughout, with the even more extreme value of $\rho = 10^{10}$ for $n = 50000$. The specification of a step limit in the quadratic case also appears to have little effect on the convergence rate, as can be seen from the results for the ill-conditioned Manevich functions (Problems 14), that are given for both $\rho = 1$ and $\rho = 10$. Here convergence is obtained to at least 11 significant figures accuracy ($\|\mathbf{x}^* - \mathbf{x}^c\|_\infty < 10^{-11}$) for each of the variables, despite the occurrence of extreme condition numbers, such as 10^{60} for the Manevich problem with $n = 200$.

The successful application of the new method to the ill-conditioned Manevich problems, and the analysis of the convergence behavior for quadratic functions, indicate that the SQSD algorithm represents a powerful approach to solving quadratic problems with large numbers of variables. In particular, the SQSD method can be seen as an *unconditionally convergent*, *stable* and *economic* alternative iterative method for solving large systems of linear equations, ill-conditioned or not, through the minimization of the sum of the squares of the residuals of the equations.

Chapter 3

The Dynamic-Q optimization algorithm

3.1 Introduction

An efficient *constrained* optimization method is presented in this chapter. The method, called the Dynamic-Q method, consists of applying the dynamic trajectory optimization algorithm (see Appendix B) to successive quadratic approximations of the actual optimization problem. This method may be considered as an extension of the unconstrained SQSD method, presented in Chapter 2, to one capable of handling general constrained optimization problems.

Due to its efficiency with respect to the number of function evaluations required for convergence, the Dynamic-Q method is primarily intended for optimization problems where function evaluations are expensive. Such problems occur frequently in engineering applications where time consuming numerical simulations may be used for function evaluations. Amongst others, these numerical analyses may take the form of a computational fluid dynamics (CFD) simulation, a structural analysis by means of the finite element method (FEM) or a dynamic simulation of a multibody system. Because these simulations are usually expensive to perform, and because the relevant functions may not be known analytically, standard classical optimization methods are normally not suited to these types of problems. Also, as will be shown, the storage requirements of the Dynamic-Q method are minimal. No Hessian information is required. The method is therefore particularly suitable for problems where the number of variables n is large.

In the next section sequential quadratic programming (SQP) methods are briefly discussed to allow for comparison with the proposed method. Next, the Dynamic-Q methodology is presented. Finally the performance of the method is tested and compared to that of an SQP method.

3.2 Sequential quadratic programming methods

Sequential quadratic programming (SQP) methods have been developed over the past thirty years, and are generally considered to be some of the most efficient algorithms available today. Based on Lagrangian methods, it can be shown that the solution $\mathbf{x}^*$ of the nonlinear equality constrained optimization problem

$$\begin{gathered}\min_{\mathbf{x}} f(\mathbf{x});\ \ \mathbf{x} = [x_1, x_2, \ldots, x_n]^T \in \Re^n \\ \text{subject to } \mathbf{h}(\mathbf{x}) = \mathbf{0}\end{gathered} \tag{3.1}$$

where $f(\mathbf{x})$ and $\mathbf{h}(\mathbf{x})$ are respectively a scalar and a vector function of $\mathbf{x}$, can be obtained by solving, at successive approximations $\mathbf{x}^i$ to $\mathbf{x}^*$, a sequence of corresponding quadratic programming (QP) subproblems ($QP[i], i = 0, 1, 2, \ldots$)

containing linearized constraints of the following form:

$$\begin{aligned} &\min_{\mathbf{s}} f(\mathbf{x}^i) + \boldsymbol{\nabla}^{\mathsf{T}} f(\mathbf{x}^i)\mathbf{s} + \frac{1}{2}\mathbf{s}^{\mathsf{T}}\mathbf{W}^i\mathbf{s} \\ &\text{subject to } \boldsymbol{\nabla}^{\mathsf{T}}\mathbf{h}(\mathbf{x}^i)\mathbf{s} + \mathbf{h}(\mathbf{x}^i) = \mathbf{0} \end{aligned} \tag{3.2}$$

where $\mathbf{W}^i = \boldsymbol{\nabla}^2 f(\mathbf{x}^i) + \boldsymbol{\lambda}^{i\mathsf{T}}\boldsymbol{\nabla}^2\mathbf{h}(\mathbf{x}^i)$, with $\boldsymbol{\lambda}^i$ denoting the associated vector of Lagrange multipliers. The solution to subproblem $QP[i]$ is denoted by $\mathbf{s}^i$ and the point at which the next subproblem $QP[i+1]$ is constructed is $\mathbf{x}^{i+1} = \mathbf{x}^i + \mathbf{s}^i$. If successful, the SQP method yields a sequence $\mathbf{x}^0, \mathbf{x}^1, \mathbf{x}^2, \ldots$ that converges to $\mathbf{x}^*$. The particular QP subproblem given here is one of a number of possible forms that may be chosen.

Based on the above argument, a simple SQP algorithm is as follows (Papalambros and Wilde [117]).

Algorithm 3.1 Simple SQP algorithm

Initialization: Select initial point $\mathbf{x}^0$ and initial Lagrange multipliers $\boldsymbol{\lambda}^0$. Set $i := 1$.

Main procedure:

1. Solve the quadratic programming problem $QP[i]$ corresponding to (3.2) to determine $\mathbf{s}^i$ and $\boldsymbol{\lambda}^{i+1}$.

2. Set $\mathbf{x}^{i+1} := \mathbf{x}^i + \mathbf{s}^i$.

3. If termination criteria are satisfied, set $\mathbf{x}^* = \mathbf{x}^{i+1}$ and stop; else set $i := i + 1$ and go to Step 1.

Numerous authors have proposed modifications and variations to the above basic algorithm. There are four areas in which the differences are most prominent. The first of these is the way in which *inequality constraints* are also included in the algorithm. For optimization problems containing inequality constraints an active set strategy may be used. This strategy can be implemented in one of two ways, either on the original problem or by including all of the inequality constraints in the QP subproblem, and applying an active set strategy to the subproblem. The second point of difference lies in the way the QP subproblem is solved. Almost any method for nonlinear programming, such as the augmented Lagrangian method or the dual method, may be specially adapted to the solution of the QP subproblem. A third way in which SQP algorithms differ from each other is in the computation of second derivatives of the problem. In the above simple SQP algorithm it is necessary to evaluate the second derivatives of the objective function and the constraints in the computation of $\mathbf{W}^i$, which will usually be a computationally intensive process. In any event, the storage of Hessian information is required which implies the availability of $O(n^2)$ storage locations, and the determination and manipulation of the elements of the $n \times n$ Hessian matrix. Some authors have avoided the latter difficulties by applying quasi-Newton updating formulae to approximate the second derivatives. Powell [118] , for example, has proposed using the BFGS formula to approximate these second derivatives. A fourth point of difference lies in dealing with the feasibility or infeasibility of the constructed subproblems. If the QP subproblem (3.2) is constructed at a point far from the solution $\mathbf{x}^*$ of the constrained optimization problem (3.1), then the subproblem may have an unbounded or infeasible solution. For this reason many modern SQP algorithms rather use $\mathbf{s}^i$ as a search direction. Then the point $\mathbf{x}^{i+1}$ at which the next subproblem is constructed is set at $\mathbf{x}^{i+1} := \mathbf{x}^i + \alpha_i \mathbf{s}^i$ with the step size α_i determined by performing a line search on an appropriate merit function in the direction $\mathbf{s}^i$.

3.3 The Dynamic-Q method

Consider the general nonlinear optimization problem:

$$\begin{aligned} &\min_{\mathbf{x}} f(\mathbf{x});\ \mathbf{x} = [x_1, x_2, \ldots, x_n]^{\mathsf{T}} \in \Re^n \\ &\text{subject to} \\ &g_j(\mathbf{x}) = \mathbf{0};\ j = 1, 2, \ldots, p \\ &h_k(\mathbf{x}) = \mathbf{0};\ k = 1, 2, \ldots, q \end{aligned} \tag{3.3}$$

where $f(\mathbf{x})$, $g_j(\mathbf{x})$ and $h_k(\mathbf{x})$ are scalar functions of $\mathbf{x}$.

In the Dynamic-Q approach, successive subproblems $P[i]$, $i = 0, 1, 2, \ldots$ are generated, at successive approximations $\mathbf{x}^i$ to the solution $\mathbf{x}^*$, by constructing *spherically quadratic* approximations $\tilde{f}(\mathbf{x})$, $\tilde{g}_j(\mathbf{x})$ and $\tilde{h}_k(\mathbf{x})$ to $f(\mathbf{x})$, $g_j(\mathbf{x})$ and $h_k(\mathbf{x})$. These approximation functions, evaluated at a point $\mathbf{x}^i$, are given by

$$\begin{aligned} \tilde{f}(\mathbf{x}) &= f(\mathbf{x}^i) + \nabla^{\mathsf{T}} f(\mathbf{x}^i)(\mathbf{x} - \mathbf{x}^i) + \frac{1}{2}(\mathbf{x} - \mathbf{x}^i)^{\mathsf{T}}\mathbf{A}(\mathbf{x} - \mathbf{x}^i) \\ \tilde{g}_j(\mathbf{x}) &= g_j(\mathbf{x}^i) + \nabla^{\mathsf{T}} g_j(\mathbf{x}^i)(\mathbf{x} - \mathbf{x}^i) \\ &\quad + \frac{1}{2}(\mathbf{x} - \mathbf{x}^i)^{\mathsf{T}}\mathbf{B}_j(\mathbf{x} - \mathbf{x}^i),\ j = 1, \ldots, p \\ \tilde{h}_k(\mathbf{x}) &= h_k(\mathbf{x}^i) + \nabla^{\mathsf{T}} h_k(\mathbf{x}^i)(\mathbf{x} - \mathbf{x}^i) \\ &\quad + \frac{1}{2}(\mathbf{x} - \mathbf{x}^i)^{\mathsf{T}}\mathbf{C}_k(\mathbf{x} - \mathbf{x}^i),\ k = 1, \ldots, q \end{aligned} \tag{3.4}$$

with the Hessian matrices $\mathbf{A}$, $\mathbf{B}_j$ and $\mathbf{C}_k$ taking on the simple forms

$$\begin{aligned} \mathbf{A} &= \operatorname{diag}(a, a, \ldots, a) = a\mathbf{I} \\ \mathbf{B}_j &= b_j\mathbf{I} \\ \mathbf{C}_k &= c_k\mathbf{I} \end{aligned} \tag{3.5}$$

Clearly the identical entries along the diagonal of the Hessian matrices indicate that the approximate subproblems $P[i]$ are indeed spherically quadratic.

For the first subproblem ($i = 0$) a linear approximation is formed by setting the curvatures a, b_j and c_k to zero. Thereafter a, b_j and c_k are chosen so that the approximating functions (3.4) interpolate their corresponding actual functions at both $\mathbf{x}^i$ and $\mathbf{x}^{i-1}$. These conditions imply that for $i = 1, 2, 3, \ldots$

$$\begin{aligned} a &= \frac{2\left[f(\mathbf{x}^{i-1}) - f(\mathbf{x}^i) - \boldsymbol{\nabla}^{\mathrm{T}} f(\mathbf{x}^i)(\mathbf{x}^{i-1} - \mathbf{x}^i)\right]}{\|\mathbf{x}^{i-1} - \mathbf{x}^i\|^2} \\ b_j &= \frac{2\left[g_j(\mathbf{x}^{i-1}) - g_j(\mathbf{x}^i) - \boldsymbol{\nabla}^{\mathrm{T}} g_j(\mathbf{x}^i)(\mathbf{x}^{i-1} - \mathbf{x}^i)\right]}{\|\mathbf{x}^{i-1} - \mathbf{x}^i\|^2}, \; j = 1, \ldots, p \quad (3.6) \\ c_k &= \frac{2\left[h_k(\mathbf{x}^{i-1}) - h_k(\mathbf{x}^i) - \boldsymbol{\nabla}^{\mathrm{T}} h_k(\mathbf{x}^i)(\mathbf{x}^{i-1} - \mathbf{x}^i)\right]}{\|\mathbf{x}^{i-1} - \mathbf{x}^i\|^2}, \; k = 1, \ldots, q \end{aligned}$$

If the gradient vectors $\boldsymbol{\nabla}^{\mathrm{T}} f$, $\boldsymbol{\nabla}^{\mathrm{T}} g_j$ and $\boldsymbol{\nabla}^{\mathrm{T}} h_k$ are not known analytically, they may be approximated from functional data by means of first-order forward finite differences.

The particular choice of spherically quadratic approximations in the Dynamic-Q algorithm has implications on the computational and storage requirements of the method. Since the second derivatives of the objective function and constraints are approximated using function and gradient data, the $O(n^2)$ calculations and storage locations, which would usually be required for these second derivatives, are not needed. The computational and storage resources for the Dynamic-Q method are thus reduced to $O(n)$. At most, $4+p+q+r+s$ n-vectors need be stored (where p, q, r and s are respectively the number of inequality and equality constraints and the number of lower and upper limits of the variables). These savings become significant when the number of variables becomes large. For this reason it is expected that the Dynamic-Q method is well suited, for example, to engineering problems such as structural optimization problems where a large number of variables are present.

In many optimization problems, additional simple side constraints of the form $\hat{k}_i \leq x_i \leq \check{k}_i$ occur. Constants $\hat{k}_i$ and $\check{k}_i$ respectively represent lower and upper bounds for variable x_i. Since these constraints are of a simple form (having zero curvature), they need not be approximated in the Dynamic-Q method and are instead explicitly treated as special linear inequality constraints. Constraints corresponding to lower and upper limits are respectively of the form

$$\begin{aligned}\hat{g}_l(\mathbf{x}) &= \hat{k}_{vl} - x_{vl} \leq 0,\ l = 1,2,\dots,r \leq n \\ \check{g}_m(\mathbf{x}) &= x_{wm} - \check{k}_{wm} \leq 0,\ m = 1,2,\dots,s \leq n\end{aligned} \tag{3.7}$$

where $vl \in \hat{I} = (v1, v2, \dots, vr)$ the set of r subscripts corresponding to the set of variables for which respective lower bounds $\hat{k}_{vl}$ are prescribed, and $wm \in \check{I} = (w1, w2, \dots, ws)$ the set of s subscripts corresponding to the set of variables for which respective upper bounds $\check{k}_{wm}$ are prescribed. The subscripts vl and wm are used since there will, in general, not be n lower and upper limits, i.e. usually $r \neq n$ and $s \neq n$.

In order to obtain convergence to the solution in a controlled and stable manner, move limits are placed on the variables. For each approximate subproblem $P[i]$ this move limit takes the form of an additional single inequality constraint

$$g_\rho(\mathbf{x}) = \left\|\mathbf{x} - \mathbf{x}^{i-1}\right\|^2 - \rho^2 \leq 0 \tag{3.8}$$

where ρ is an appropriately chosen step limit and $\mathbf{x}^{i-1}$ is the solution to the previous subproblem.

The approximate subproblem, constructed at $\mathbf{x}^i$, to the optimization problem (3.4) (plus simple side constraints (3.7) and move limit (3.8)), thus becomes $P[i]$:

$$\begin{gathered}\min_{\mathbf{x}} \tilde{f}(\mathbf{x}),\ \mathbf{x} = [x_1, x_2, ..., x_n]^{\mathrm{T}} \in \Re^n \\ \text{subject to} \\ \tilde{g}_j(\mathbf{x}) \leq 0,\ j = 1,2,\dots,p \\ \tilde{h}_k(\mathbf{x}) = 0,\ k = 1,2,\dots,q \\ \hat{g}_l(\mathbf{x}) \leq 0,\ l = 1,2,\dots,r \\ \check{g}_m(\mathbf{x}) \leq 0,\ m = 1,2,\dots,s \\ g_\rho(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}^{i-1}\|^2 - \rho^2 \leq 0\end{gathered} \tag{3.9}$$

with solution $\mathbf{x}^{*i}$. The Dynamic-Q algorithm is given by Algorithm 3.2. In the Dynamic-Q method the subproblems generated are solved using the dynamic trajectory, or "leap-frog" (LfopC) method of Snyman [101, 102] for unconstrained optimization applied to penalty function formulations (Snyman *et al.* [119], Snyman [103]) of the constrained problem. A brief description of the LfopC algorithm is given in Appendix B.

Algorithm 3.2 Dynamic-Q algorithm

```
Initialization: Select starting point $\mathbf{x}^0$ and move limit $\rho$. Set $i := 0$.
Main procedure:
    1. Evaluate $f(\mathbf{x}^i)$, $g_j(\mathbf{x}^i)$ and $h_k(\mathbf{x}^i)$ as well as $\nabla f(\mathbf{x}^i)$, $\nabla g_j(\mathbf{x}^i)$ and
       $\nabla h_k(\mathbf{x}^i)$. If termination criteria are satisfied set $\mathbf{x}^* = \mathbf{x}^i$ and stop.
    2. Construct a local approximation $P[i]$ to the optimization problem at
       $\mathbf{x}^i$ using expressions (3.4) to (3.6).
    3. Solve the approximated subproblem $P[i]$ (given by (3.9)) using the
       constrained optimizer LfopC with $\mathbf{x}^0 := \mathbf{x}^i$ (see Appendix B) to give
       $\mathbf{x}^{*i}$.
    4. Set $i := i + 1$, $\mathbf{x}^i := \mathbf{x}^{*(i-1)}$ and return to Step 2.
```

The LfopC algorithm possesses a number of outstanding characteristics, which makes it highly suitable for implementation in the Dynamic-Q methodology. The algorithm requires only gradient information and no explicit line searches or function evaluations are performed. These properties, together with the influence of the fundamental physical principles underlying the method, ensure that the algorithm is extremely robust. This has been proven over many years of testing (Snyman [103]). A further desirable characteristic related to its robustness, and the main reason for its application in solving the subproblems in the Dynamic-Q algorithm, is that if there is no feasible solution to the problem, the LfopC algorithm will still find the best possible compromised solution without breaking down. The Dynamic-Q algorithm thus usually converges to a solution from an infeasible remote point without the need to use line searches between subproblems, as is the case with SQP. The LfopC algorithm used by Dynamic-Q is identical to that presented in Snyman [103] except for a minor change to Lfop which is advisable should the subproblems become effectively unconstrained.

3.4 Numerical results and conclusion

The Dynamic-Q method requires very few parameter settings by the user. Other than convergence criteria and specification of a maximum number of iterations, the only parameter required is the step limit ρ. The algorithm is not very sensitive to the choice of this parameter, however, ρ should be chosen of the same order of magnitude as the diameter of the region of interest. For the problems listed in Table 3.1 a step limit of $\rho = 1$ was used except for problems 72 and 106 where step limits and $\rho = 100$ were used respectively.

Given specified positive tolerances ε_x, ε_f and ε_c, then at step i termination of the algorithm occurs if the normalized step size

$$\frac{\|\mathbf{x}^i - \mathbf{x}^{i-1}\|}{1 + \|\mathbf{x}^i\|} < \varepsilon_x \tag{3.10}$$

or if the normalized change in function value

$$\frac{|f^i - f_{\text{best}}|}{1 + |f_{\text{best}}|} < \varepsilon_f \tag{3.11}$$

where f_{best} is the lowest previous feasible function value and the current $\mathbf{x}^i$ is feasible. The point $\mathbf{x}^i$ is considered feasible if the absolute value of the violation of each constraint is less than ε_c. This particular function termination criterion is used since the Dynamic-Q algorithm may at times exhibit oscillatory behavior near the solution.

In Table 3.1, for the same starting points, the performance of the Dynamic-Q method on some standard test problems is compared to results obtained for Powell's SQP method as reported by Hock and Schittkowski [120]. The

Prob. #	n	f_{act}	SQP			Dynamic-Q		
			N^{fg}	f^*	E^r	N^{fg}	f^*	E^r
2	2	5.04E-02	16~	2.84E+01	2.70E+01	7*	4.94E+00	<1.00E-08
10	2	-1.00E+00	12	-1.00E+00	5.00E-08	13	-1.00E+00	<1.00E-08
12	2	-3.00E+01	12	-3.00E+01	<1.00E-08	9	-3.00E+01	<1.00E-08
13	2	1.00E+00	45	1.00E+00	5.00E-08	50$	9.59E-01	2.07E-02
14	2	1.39E+00	6	1.39E+00	8.07E-09	5	1.39E+00	7.86E-07
15	2	3.07E+02	5	3.07E+02	<1.00E-08	15*	3.60E+02	5.55E-07
16	2	2.50E-01	6*	2.31E+01	<1.00E-08	5*	2.31E+01	<1.00E-08
17	2	1.00E+00	12	1.00E+00	<1.00E-08	16	1.00E+00	<1.00E-08
20	2	3.82E+01	20	3.82E+01	4.83E-09	4*	4.02E+01	<1.00E-08
22	2	1.00E+00	9	1.00E+00	<1.00E-08	3	1.00E+00	<1.00E-08
23	2	2.00E+00	7	2.00E+00	<1.00E-08	5	2.00E+00	<1.00E-08
24	2	-1.00E+00	5	-1.00E+00	<1.00E-08	4	-1.00E+00	1.00E-08
26	3	0.00E+00	19	4.05E-08	4.05E-08	27	1.79E-07	1.79E-07
27	3	4.00E-02	25	4.00E-02	1.73E-08	28	4.00E-02	9.62E-10
28	3	0.00E+00	5	2.98E-21	2.98E-21	12	7.56E-10	7.56E-10
29	3	-2.26E+01	13	-2.26E+01	8.59E-11	11	-2.26E+01	8.59E-11
30	3	1.00E+00	14	1.00E+00	<1.00E-08	5	1.00E+00	<1.00E-08
31	3	6.00E+00	10	6.00E+00	<1.00E-08	10	6.00E+00	1.43E-08
32	3	1.00E+00	3	1.00E+00	<1.00E-08	4	1.00E+00	<1.00E-08
33	3	-4.59E+00	5*	-4.00E+00	<1.00E-08	3*	-4.00E+00	<1.00E-08
36	3	-3.30E+03	4	-3.30E+03	<1.00E-08	15	-3.30E+03	<1.00E-08
45	5	1.00E+00	8	1.00E+00	<1.00E-08	7	1.00E+00	1.00E-08
52	5	5.33E+00	8	5.33E+00	5.62E-09	12	5.33E+00	1.02E-08
55	6	6.33E+00	1~	6.00E+00	4.54E-02	2*	6.66E+00	1.30E-09
56	7	-3.46E+00	11	-3.46E+00	<1.00E-08	20	-3.46E+00	6.73E-08
60	3	3.26E-02	9	3.26E-02	3.17E-08	11	3.26E-02	1.21E-09
61	3	-1.44E+02	10	-1.44E+02	1.52E-08	10	-1.44E+02	1.52E-08
63	3	9.62E+02	9	9.62E+02	2.18E-09	6	9.62E+02	2.18E-09
65	3	9.54E-01	11~	2.80E+00	9.47E-01	9	9.54E-01	2.90E-08
71	4	1.70E+01	5	1.70E+01	1.67E-08	6	1.70E+01	1.67E-08
72	4	7.28E+02	35	7.28E+02	1.37E-08	30	7.28E+02	1.37E-08
76	4	-4.68E+00	6	-4.68E+00	3.34E-09	8	-4.68E+00	3.34E-09
78	5	-2.92E+00	9	-2.92E+00	2.55E-09	6	-2.92E+00	2.55E-09
80	5	5.39E-02	7	5.39E-02	7.59E-10	6	5.39E-02	7.59E-10
81	5	5.39E-02	8	5.39E-02	1.71E-09	12	5.39E-02	1.90E-10
100	7	6.80E+02	20	6.80E+02	<1.00E-08	16	6.80E+02	1.46E-10
104	8	3.95E+00	19	3.95E+00	8.00E-09	42	3.95E+00	5.26E-08
106	8	7.05E+03	44	7.05E+03	1.18E-05	79	7.05E+03	1.18E-05
108	9	-8.66E-01	9*	-6.97E-01	1.32E-02	26	-8.66E-01	3.32E-09
118	15	6.65E+02	~	~	~	38	6.65E+02	3.00E-08
Svan	21	2.80E+02	150	2.80E+02	9.96E-05	93	2.80E+02	1.59E-06

* Converges to a local minimum - listed E^r relative to function value at local minimum;
~ Fails; $ Terminates on maximum number of steps

Table 3.1: Performance of the Dynamic-Q and SQP optimization algorithms

problem numbers given correspond to the problem numbers in Hock and Schittkowski's book. For each problem, the actual function value f_{act} is given, as well as, for each method, the calculated function value f^* at convergence, the relative function error

$$E^r = \frac{|f_{\text{act}} - f^*|}{1 + |f_{\text{act}}|} \tag{3.12}$$

and the number of function-gradient evaluations (N^{fg}) required for convergence. In some cases it was not possible to calculate the relative function error due to rounding off of the solutions reported by Hock and Schittkowski. In these cases the calculated solutions were correct to at least eight significant figures. For the Dynamic-Q algorithm, convergence tolerances of $\varepsilon_f = 10^{-8}$ on the function value, $\varepsilon_x = 10^{-5}$ on the step size and $\varepsilon_c = 10^{-6}$ for constraint feasibility, were used. These were chosen to allow for comparison with the reported SQP results.

The result for the 12-corner polytope problem of Svanberg [121] is also given. For this problem the results given in the SQP columns are for Svanberg's Method of Moving Asymptotes (MMA). The recorded number of function evaluations for this method is approximate since the results given correspond to 50 outer iterations of the MMA, each requiring about 3 function evaluations.

A robust and efficient method for nonlinear optimization, with minimal storage requirements compared to those of the SQP method, has been proposed and tested. The particular methodology proposed is made possible by the special properties of the LfopC optimization algorithm (Snyman [103]), which is used to solve the quadratic subproblems. Comparison of the results for Dynamic-Q with the results for the SQP method show that equally accurate results are obtained with comparable number of function evaluations.

Part II

MANIPULATOR OPTIMIZATION

Chapter 4

Choice of a suitable dimensional synthesis methodology

4.1 Introduction

The main objective of this chapter is to find a fundamentally sound and robust *numerical* methodology for synthesizing parallel manipulators for a desired workspace. Various formulations for manipulator dimensional synthesis are proposed and investigated numerically by application to a two-degree-of-freedom (dof) parallel manipulator. Specifically, the various synthesis strategies result in one unconstrained optimization formulation and two constrained optimization formulations. In the next section, the 2-$R\underline{P}R$ planar parallel manipulator studied in this chapter is described. Thereafter four candidate numerical optimization algorithms, LfopC, EtopC, Dynamic-Q and SQSD, are briefly discussed. These four optimization algorithms are applied to the the unconstrained O synthesis (definition follows below) manipulator design formulation in order to assess their relative merits and potential for application to parallel manipulator dimensional synthesis problems in general. The goal of the O synthesis formulation is to determine a manipulator design so that its workspace is as close as possible to some prescribed workspace. The most suitable optimization algorithm identified by means of the O synthesis problem is then applied to the proposed (constrained) E and P synthesis formulations. The E synthesis formulation seeks to determine the manipulator design so that workspace of the optimal manipulator fully contains a prescribed workspace in the most efficient manner. The second, more practical, P synthesis formulation is aimed at determining a manipulator design so that a prescribed workspace is fully enclosed, but that the workspace is also well-conditioned with respect to some performance index. The final methodology arrived upon is applied to the more complicated case of a 3-dof planar parallel manipulator in the next chapter.

4.2 Coordinates and kinematic constraints

In general terms, when describing the kinematics of a mechanism, the following descriptions and definitions can be used (Haug *et al.* [16]). *Generalized coordinates* $\mathbf{q} = [q_1, q_2, \ldots, q_{nq}]^T \in \Re^{nq}$ are used to characterize the position and orientation of each body in the mechanism. It the vicinity of an *assembled configuration* of the mechanism, these generalized coordinates satisfy m independent holonomic kinematic constraint equations of the form

$$\mathbf{\Phi}(\mathbf{q}) = \mathbf{0} \tag{4.1}$$

where $\mathbf{\Phi} : \Re^{nq} \rightarrow \Re^{m}$ is a smooth function.

Mechanisms are usually designed to produce a desired functionality. Specifying the values of a selected subset of the generalized coordinates, called the *input coordinates*, defines the motion of the mechanism. These input coordinate values are controlled by external influences with the intent of prescribing the motion of the mechanism. The vector of input coordinates is denoted by $\mathbf{v} = [v_1, v_2, \ldots, v_{nv}]^{\mathsf{T}} \in \Re^{nv}$.

In order to characterize the functionality of the mechanism some measure of output, which is controlled by the mechanism inputs, must be monitored. *Output coordinates* are the subset of the mechanism's generalized coordinates that define the useful functionality of the mechanism. Output coordinates are distinct from input coordinates and are denoted by $\mathbf{u} = [u_1, u_2, \ldots, u_{nu}]^{\mathsf{T}} \in \Re^{nu}$. A choice of input and output coordinates for a mechanism defines a mechanical system with an intended function. This mechanism is then called a manipulator.

Generalized coordinates of a mechanism that are neither input coordinates nor output coordinates are called *intermediate coordinates*, denoted by $\mathbf{w} = [w_1, w_2, \ldots, w_{nw}]^{\mathsf{T}} \in \Re^{nw}$, where $nw = nq - nv - nu$.

4.3 The planar two-degree-of-freedom parallel manipulator

The mechanism used to investigate the various methodologies proposed in this chapter is a planar 2-$R\underline{P}R$ parallel manipulator similar to that studied by Gosselin and Guillot [84]. As shown in Figure 4.1, the manipulator consists of two linear actuators, with variable lengths l_1 and l_2, connected to the ground by means of revolute joints A and B, and to each other by a revolute joint P with global coordinates (x_P, y_P). Point A has coordinates (x_A, y_A) and point B coordinates (x_B, y_B). It assumed here that $y_B = y_A$. Point P will be used to describe the motion of the mechanism and is called the *working point* of the manipulator. Here for the sample 2-dof manipulator, with no limits on the actuator lengths, point P may arbitrarily be positioned in the $x - y$ plane by controlling the lengths of legs 1 and 2. It is evident that this manipulator thus has two degrees of freedom. The *input* coordinates, which are used to control the manipulator, are the leg lengths $\mathbf{v} = [l_1, l_2]^{\mathsf{T}} \in \Re^2$. *Output* coordinates,

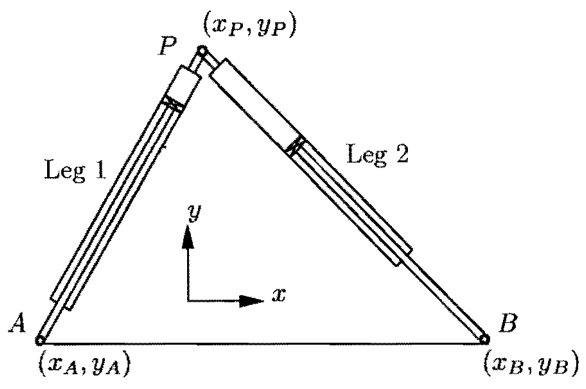

Figure 4.1: The 2-dof parallel manipulator

describing the functionality of the mechanism, are $\mathbf{u} = [x_P, y_P]^\mathrm{T} \in \Re^2$. There are no intermediate coordinates. Thus for this 2-dof example $nu = 2, nv = 2$ and $nw = 0$. The generalized coordinates of the manipulator are partitioned as follows

$$\mathbf{q} = [\mathbf{u}^\mathrm{T}, \mathbf{v}^\mathrm{T}]^\mathrm{T} \in \Re^4 \tag{4.2}$$

For this two-degree-of-freedom manipulator there are two kinematic constraint equations (i.e. $m = 2$), and (4.1) can be rewritten in terms of the partitioning given by (4.2)

$$\mathbf{\Phi}(\mathbf{q}) = \mathbf{\Phi}(\mathbf{u}, \mathbf{v}) = \mathbf{0} \tag{4.3}$$

The motion of the manipulator is restricted when the actuator legs have limits associated with them of the form

$$l_i^{\min} \leq l_i \leq l_i^{\max}, i = 1, 2 \tag{4.4}$$

or more generally

$$\mathbf{v}^{\min} \leq \mathbf{v} \leq \mathbf{v}^{\max} \tag{4.5}$$

where $\mathbf{v}^{\min} = [l_1^{\min}, l_2^{\min}]^\mathrm{T}$ and $\mathbf{v}^{\max} = [l_1^{\max}, l_2^{\max}]^\mathrm{T}$.

These constraints, together with the geometry of the manipulator, determine the size and shape of the workspace of the manipulator. Since the manipulator has no orientational ability, the types of workspace discussed in Section 1.3.1 have no meaning here, and the workspace W is simply the set of points that can be reached by the working point P of the manipulator. The workspace can thus be defined as

$$W = \{\mathbf{u} \in \Re^{nu} : \mathbf{\Phi}(\mathbf{u}, \mathbf{v}) = \mathbf{0}, \text{ with } \mathbf{v} \text{ satisfying (4.5)}\} \tag{4.6}$$

The boundary ∂W of the workspace may then be defined as

$$\begin{aligned} \partial W = \{&\mathbf{u} \in \Re^{nu} : \mathbf{u} \in W \text{ and } \exists \text{ an } \mathbf{s} \in \Re^{nu} \text{ such that for} \\ &\mathbf{u}' = \mathbf{u} + \lambda\mathbf{s}, \lambda \in \Re \text{ arbitrarily small and either} \\ &\text{positive or negative, no } \mathbf{v} \text{ exists that satisfies} \\ &\mathbf{\Phi}(\mathbf{u}', \mathbf{v}) = \mathbf{0} \text{ as well as inequalities (4.5)}\} \end{aligned} \tag{4.7}$$

The particular workspace determination method used here is the chord optimization method (Snyman and Hay [122]) although the ray optimization method (Snyman *et al.* [48]) would be equally applicable. The chord method is fully described in Appendix C and yields on application discrete points $\mathbf{b}^i \in \mathbf{u}, i = 1, 2, \ldots, n_b$ along the boundary of the workspace at constant chord lengths d, as shown in Figure 4.2. Included in this set of points are the bifurcation points $\mathbf{B}^j \in \mathbf{u}, j = 1, 2, \ldots, N_B$. It is arbitrarily assumed that all points are ordered counterclockwise.

4.4 Candidate optimization algorithms

In this preliminary study four different optimization algorithms, all developed at the University of Pretoria, were compared in order to assess their suitability for solving the manipulator optimization problems to be addressed. The four candidate algorithms were the unconstrained spherical quadratic steepest descent optimization algorithm, and three constrained optimization algorithms: LfopC, Dynamic-Q and EtopC.

Of these algorithms, the SQSD method, developed in Chapter 2, has proven to work particularly well on ill-conditioned problems with large numbers of variables, however the fact that this is an unconstrained algorithm limits its suitability for manipulator optimization problems. The LfopC algorithm ((Snyman [103], see Appendix B)) possesses a number of outstanding characteristics making it a good potential candidate for implementation. The algorithm requires only gradient information, and no explicit line searches or function evaluations are performed. These properties, together with the influence of the fundamental physical principles underlying the method, ensure that the algorithm is extremely robust. This has been proven over many years of testing (Snyman [103]). A further desirable characteristic related to its robustness, is that if there is no feasible solution to the problem, the LfopC algorithm will still find the best possible compromised solution without breaking down. The one disadvantage of the LfopC method is that, although it moves quickly to the vicinity of the solution, it may converge relatively slowly towards the exact optimum if high accuracy is required. In an attempt to reduce the number of gradient evaluations required, while still retaining the robust properties of LfopC, the Dynamic-Q method (Chapter 3) may prove to be useful.

In order to obtain a gradient-only method with fast convergence in the vicinity of the solution, the conjugate gradient method has been adapted to require only gradient information (Snyman [114]). In the resultant algorithm EtopC, the use of only gradient evaluations is made possible by exploiting a step size selection procedure based on an Euler-trapezium integration scheme. Either the Fletcher-Reeves or Polak-Ribiere directions can be used. In this study, the Polak-Ribiere version of EtopC has been used.

4.5 Best overall fit to a prescribed workspace

The goal of the first formulation presented in this section is stated as

> **O synthesis:** *Determine a manipulator design that results in a workspace W_c which most closely approximates a prescribed workspace W_p.*[1]

4.5.1 Optimization formulation

As indicated in Figure 4.2, the prescribed and calculated workspace boundaries are defined using polar coordinates (β, r), centered on a local coordinate system $x' - y'$ at $\mathbf{O}'$. The angle between the x'-axis of the local coordinate system and the ray to a point $\mathbf{b}_p$ on the prescribed boundary is denoted β_p. The distance from $\mathbf{O}'$ to $\mathbf{b}_p$ is r_p. The boundary of the workspace W_c, associated with specific manipulator design $\mathbf{d}$, may be defined in a similar manner. Here the boundary point $\mathbf{b}_c^i$, generated by the chord method (Snyman and Hay [122], see Appendix C), corresponds to angle β_c^i and ray length r_c^i. For convenience, a point on the prescribed boundary at the angle β_c^i that corresponds to the computed workspace boundary point $\mathbf{b}_c^i$ is denoted $\mathbf{b}_p^i$ and $r_p(\beta_c^i) = r_p^i$.

The part of workspace W_p not intersecting W_c is denoted δW_p, and the part of workspace W_c not intersecting W_p is denoted δW_c (respectively indicated by the light and dark shaded areas in Figure 4.2). It is assumed that the workspace W_c, dependent on the design vector $\mathbf{d}$, will most closely approximate the prescribed workspace W_p when $\mathbf{d}$ is chosen such that the sum of the non-intersecting areas δW_p and δW_c is minimized (Gosselin and Guillot [84]). The optimum solution $\mathbf{d}^*$ is obtained by solving the unconstrained optimization problem :

$$\min_{\mathbf{d}} f(\mathbf{d}) = w_p \delta W_p + w_c \delta W_c \tag{4.8}$$

where the respective weights w_p and w_c in the objective function satisfy the

[1]Subscripts p and c respectively denote quantities associated with the *prescribed* workspace, and workspace *calculated* for a specific design

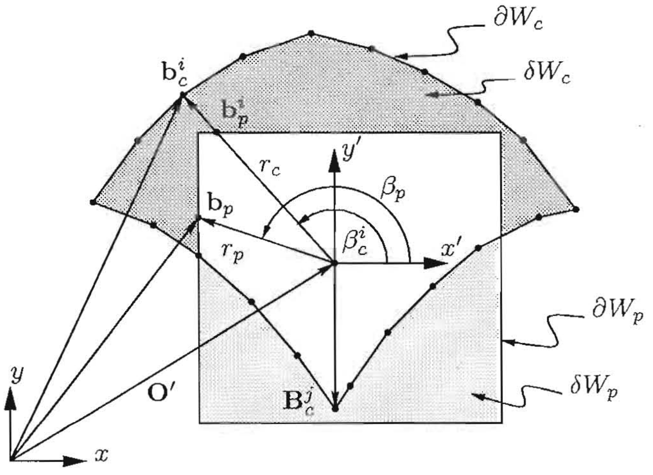

Figure 4.2: Prescribed and calculated workspaces

condition $w_p + w_c = 1$, and are chosen here to be equal, and thus each take on the value 0.5.

The calculation of approximations to the areas δW_p and δW_c is performed using a numerical scheme. Considering two consecutive calculated workspace boundary points $\mathbf{b}_c^i$ and $\mathbf{b}_c^{i+1}$, the incremental contributions dW_p and dW_c to areas δW_p and δW_c may be calculated by use of the following expression, relating the area A of a triangle to the coordinates $\mathbf{L} = [L_x, L_y]^\top$, $\mathbf{M} = [M_x, M_y]^\top$ and $\mathbf{N} = [N_x, N_y]^\top$ of its vertices:

$$A(\mathbf{L}, \mathbf{M}, \mathbf{N}) = \frac{1}{2} |M_x N_y + N_x L_y + L_x M_y - M_x L_y - L_x N_y - N_x M_y| \quad (4.9)$$

In applying formula (4.9), a distinction must be made between four possibilities that may arise regarding the relative position of the prescribed workspace boundary to any two adjacent calculated workspace boundary points. With

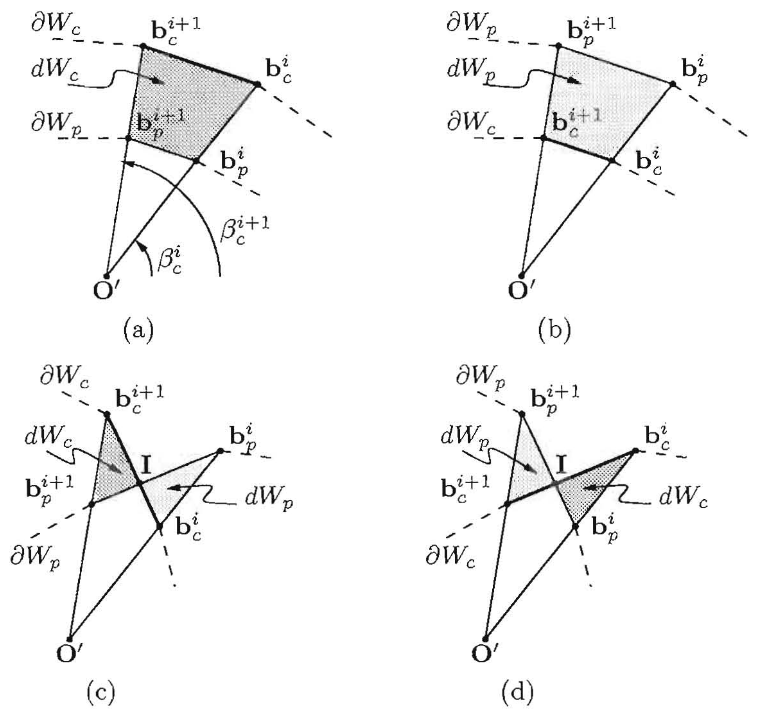

Figure 4.3: Various cases for numerical calculation of workspace areas

reference to Figure 4.3 these four possibilities are:

$$\begin{aligned}
&\text{a.}\quad \text{If } r_c^i > r_p^i \text{ and } r_c^{i+1} > r_p^{i+1} \text{ then}\\
&\qquad dW_c = A(\mathbf{b}_c^i, \mathbf{b}_c^{i+1}, \mathbf{O}') - A(\mathbf{b}_p^i, \mathbf{b}_p^{i+1}, \mathbf{O}_p')\\
&\text{b.}\quad \text{If } r_c^i < r_p^i \text{ and } r_c^{i+1} < r_p^{i+1} \text{ then}\\
&\qquad dW_p = A(\mathbf{b}_p^i, \mathbf{b}_p^{i+1}, \mathbf{O}') - A(\mathbf{b}_c^i, \mathbf{b}_c^{i+1}, \mathbf{O}')\\
&\text{c.}\quad \text{If } r_c^i < r_p^i \text{ and } r_c^{i+1} > r_p^{i+1} \text{ then}\\
&\qquad dW_c = A(\mathbf{b}_c^{i+1}, \mathbf{b}_p^{i+1}, \mathbf{I})\\
&\qquad dW_p = A(\mathbf{b}_c^i, \mathbf{b}_p^i, \mathbf{I})\\
&\text{d.}\quad \text{If } r_c^i > r_p^i \text{ and } r_c^{i+1} < r_p^{i+1} \text{ then}\\
&\qquad dW_c = A(\mathbf{b}_p^i, \mathbf{b}_c^i, \mathbf{I})\\
&\qquad dW_p = A(\mathbf{b}_p^{i+1}, \mathbf{b}_c^{i+1}, \mathbf{I})
\end{aligned} \tag{4.10}$$

where point $\mathbf{I}$ is calculated as required.

Areas δW_p and δW_c can now be approximated by summing the incremental contributions dW_p and dW_c for sequential pairs of boundary points $(\mathbf{b}_c^i, \mathbf{b}_c^{i+1}), i = 1, \ldots, n_{bc} - 2$ where n_{bc} is the number of boundary points generated by the chord method. The closing boundary section is calculated by considering points $(\mathbf{b}_c^{n_{bc}-1}, \mathbf{b}_c^1)$. Note that the final boundary point $\mathbf{b}^{n_{bc}}$ fulfills no part in the calculation of the workspace area since it lies between points $\mathbf{b}_c^1$ and $\mathbf{b}_c^2$. Using this methodology, the objective function in optimization problem (4.8) can be computed for any design vector $\mathbf{d}$. Hence, solving minimization problem (4.8) by means of a suitable optimization algorithm results in a manipulator design $\mathbf{d}^*$ which is a solution to the O synthesis problem.

4.5.2 Numerical results

The four optimization algorithms, listed in Section 4.4, are compared and evaluated in this section to determine which one most efficiently and robustly solves optimization problems of the form (4.8). The particular prescribed workspace chosen is that already considered by Gosselin and Guilliot [84] and shown in Figure 4.4. The choice of this prescribed workspace provides some means of validation of the current results.

The boundary of the prescribed workspace is defined in polar coordinates relative to a local coordinate system centered at $\mathbf{O}' = [0, 3]^{\mathrm{T}}$:

$$r_p(\beta_p) = \alpha + \sqrt{D^2 \cos^2(\beta_p) + R^2 - D^2} \tag{4.11}$$

where $R = 2.86\dot{7}$ and $D = 1.68\dot{7}$ and the expression for α for various intervals of β_p is given in Table 4.1.

Each of the four candidate optimization algorithms is applied to problem (4.8) using the prescribed workspace defined by (4.11), and run from four different starting designs $\mathbf{d}^0$, given in Table 4.2. The workspaces corresponding to these starting designs are shown, together with the prescribed workspace, in Figure 4.4. A chord length of $d = 0.5$ was used for calculating the

$[\beta_{p\,\min}, \beta_{p\,\max})$	α
$[-\pi/4, \pi/4)$	$-D\cos(\beta_p)$
$[\pi/4, 3\pi/4)$	$-D\sin(\beta_p)$
$[3\pi/4, 5\pi/4)$	$+D\cos(\beta_p)$
$[5\pi/4, 7\pi/4)$	$+D\sin(\beta_p)$

Table 4.1: Parameters specifying the prescribed workspace for O synthesis

	x_A	y_A	x_B	$l_1^{\min}$	$l_2^{\min}$
SP1	-2	-0.075	2	3	3
SP2	-4	-1	4	5	5
SP3	-1	-1	4	3	5
SP4	-5	-1	1	5	3

Table 4.2: O synthesis starting designs

workspace boundary, usually resulting in $n_{bc} \approx 22$ at the solution.

For the 2-dof manipulator there are in total eight parameters which have an effect on the shape and position of the workspace. These parameters are

$$\mathbf{d} = [x_A, y_A, x_B, y_B, l_1^{\min}, l_1^{\max}, l_2^{\min}, l_2^{\max}]^T \tag{4.12}$$

It is assumed for this illustrative example, in which arbitrary units of length are used, that $y_B = y_A$ and $l_i^{\max} = 1.5 l_i^{\min}, i = 1, 2$. The design vector now becomes

$$\mathbf{d} = [x_A, y_A, x_B, l_1^{\min}, l_2^{\min}]^T \tag{4.13}$$

For each starting design and algorithm, Tables 4.3 to 4.6 give the number of gradient vector evaluations (N^g) required for convergence, as well as final objective function values $f(\mathbf{d}^*)$ and solution vectors $\mathbf{d}^*$. For all algorithms, and where applicable, the termination tolerances on the function value, step size and gradient norm were respectively set to $\varepsilon_f = 10^{-6}$, $\varepsilon_x = 10^{-4}$ and

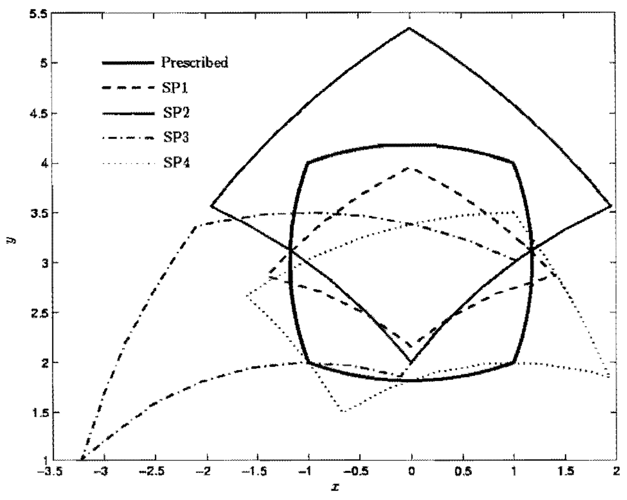

Figure 4.4: Prescribed workspace and workspaces corresponding to starting designs

$\varepsilon_g = 10^{-3}$. Where applicable, move limits were set to $\rho = 0.2$. Figures 4.5 to 4.8 give the convergence histories corresponding to Tables 4.3 to 4.6.

The number of function evaluations (N^f) in the case of the Complex method used by Gosselin and Guillot [84] is not reported. For the Pretoria algorithms (see Section 4.4), because forward finite differences are used for computing the components of the gradient vector, $N^f = (n+1)N^g$, where $n = 5$ is the number of design variables. The interval used in calculating the forward finite differences is $\Gamma = 10^{-5}$ for all the variables.

Algorithm	N^g	f^*	$\mathbf{d}^*$
Dynamic-Q	97	0.596617	$[-3.835, -0.6972, 3.382, 4.277, 4.059]^T$
SQSD	124	0.595935	$[-3.767, -0.6440, 3.552, 4.214, 4.119]^T$
LfopC	152	0.611484	$[-3.962, -0.7344, 3.298, 4.361, 4.058]^T$
EtopC	379	0.611510	$[-3.939, -0.7344, 3.325, 4.348, 4.072]^T$
Complex [84]	-	0.654	$[-4.25, -0.81, 3.02, 4.57, 3.92]^T$

Table 4.3: O synthesis solutions obtained from SP1

Algorithm	N^g	f^*	$\mathbf{d}^*$
Dynamic-Q	32	0.618382	$[-3.873, -0.9893, 3.611, 4.482, 4.324]^\top$
SQSD	90	0.612084	$[-3.890, -0.7318, 3.410, 4.318, 4.114]^\top$
LfopC	161	0.617327	$[-3.947, -0.9825, 3.581, 4.519, 4.309]^\top$
EtopC	444	0.617914	$[-3.941, -0.9638, 3.642, 4.508, 4.326]^\top$

Table 4.4: O synthesis solutions obtained from SP2

Algorithm	N^g	f^*	$\mathbf{d}^*$
Dynamic-Q	80	0.640436	$[-2.928, -0.6551, 4.189, 3.810, 4.502]^\top$
SQSD	71	0.627223	$[-2.818, -0.5753, 4.367, 3.711, 4.540]^\top$
LfopC	186	0.625862	$[-2.896, -0.5905, 4.241, 3.756, 4.472]^\top$
EtopC	228	0.647717	$[-3.005, -0.9524, 3.980, 4.016, 4.501]^\top$

Table 4.5: O synthesis solutions obtained from SP3

Algorithm	N^g	f^*	$\mathbf{d}^*$
Dynamic-Q	86	0.633575	$[-4.075, -0.5207, 2.786, 4.343, 3.679]^\top$
SQSD	189	0.632303	$[-4.005, -0.5264, 2.839, 4.305, 3.707]^\top$
LfopC	116	0.631643	$[-3.878, -0.5418, 2.916, 4.242, 3.753]^\top$
EtopC	509	0.595853	$[-3.828, -0.6684, 3.455, 4.259, 4.081]^\top$

Table 4.6: O synthesis solutions obtained from SP4

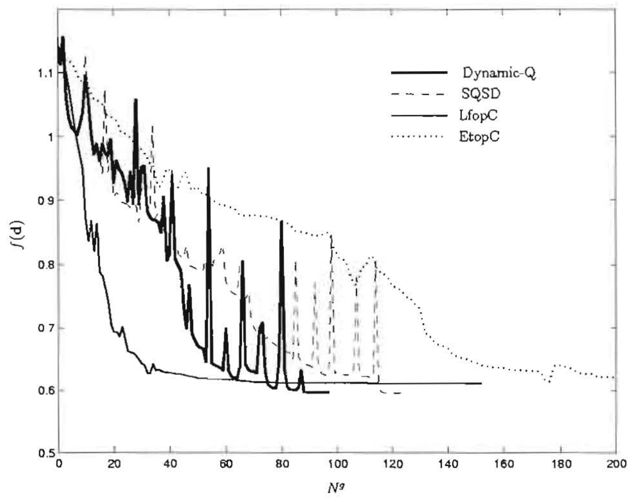

Figure 4.5: O synthesis convergence histories from SP1

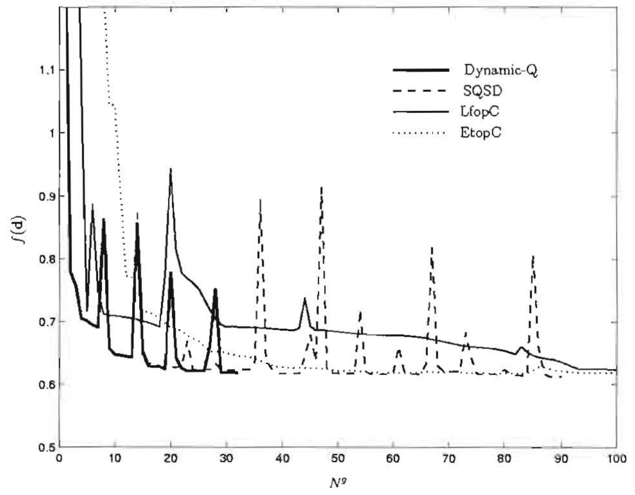

Figure 4.6: O synthesis convergence histories from SP2

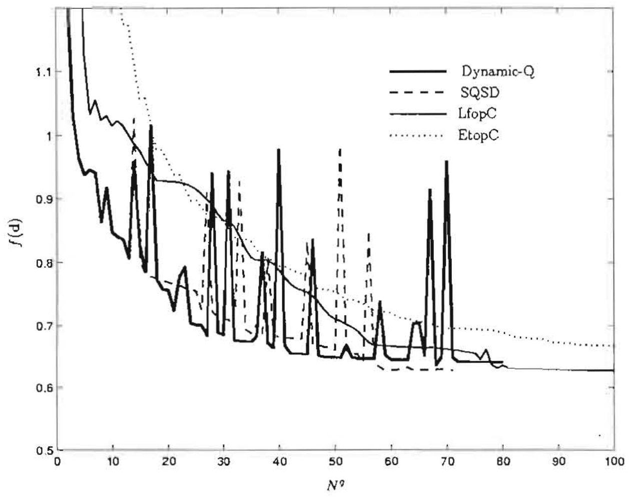

Figure 4.7: O synthesis convergence histories from SP3

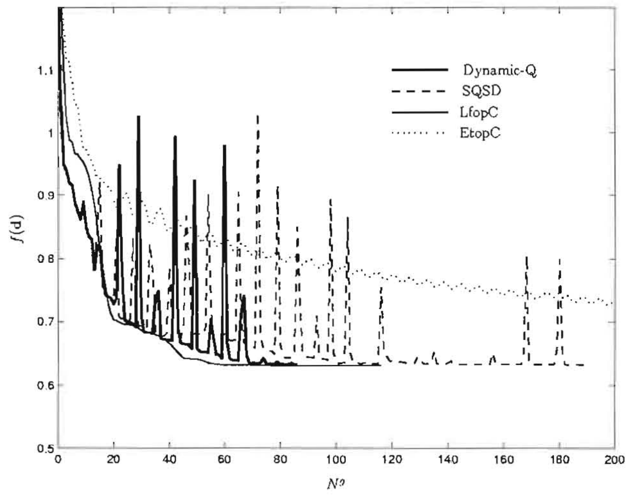

Figure 4.8: O synthesis convergence histories from SP4

In all cases except from SP3, Dynamic-Q is the fastest of the four methods. For SP3, SQSD is marginally faster. As expected, LfopC moves quickly to the vicinity of the solution but takes relatively longer to converge to the accuracy specified. EtopC is the slowest of the four methods but does, however, yield the overall lowest function value of $f^* = 0.595853$ for SP4.

A point of interest relating to the accuracy of the solutions obtained for the O synthesis problem is that the objective function is relatively insensitive to large changes in the design vector. Consider for example the highest (EtopC for SP3) and lowest (EtopC for SP4) solutions obtained. The distance (ℓ_2 norm) between the two solution vectors is 1.127 whereas the difference in function value is only 0.052. This result is typical, and further inspection of the results would suggest a design space which contains a flat-bedded, steep sided solution valley where a large number of different and widely separated design vectors give objective function values close to the global optimum value. Indeed, there does not seem to be a sharp global optimum, or for that matter, sharp local optima since of the 16 runs none converged to identical solutions. A further factor contributing to this is the small amount of numerical noise present in the optimization due to the numerical approximation of areas ∂W_c and ∂W_p.

From a practical point of view it is important to note that all the solutions obtained are almost equally valid in giving designs that closely match the prescribed workspace. This means that any of the algorithms (with indeed any starting point) may in practice be used to satisfactorily solve the design problem. This is further illustrated by Figure 4.9 which depicts the best and worst computed workspaces relative to the prescribed workspace. For comparative purposes the Complex Method solution of Gosselin and Guillot [84] is also shown.

The only remaining consideration is that of computational expense. Although the Dynamic-Q algorithm does not find the lowest solution for each starting point, the solutions found are very good, and its superior perfor-

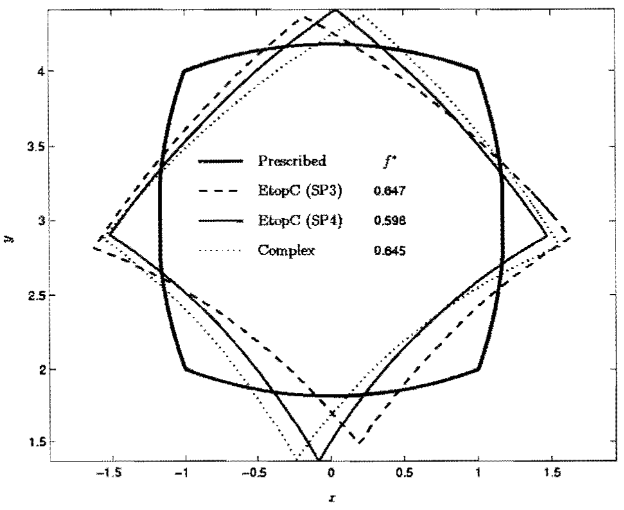

Figure 4.9: O synthesis workspaces corresponding to optimal designs

mance with respect to computational efficiency offsets any other slight disadvantage. Although based on the same approximation principles as Dynamic-Q, SQSD appears to be less efficient, requiring overall many more gradient evaluations than Dynamic-Q. The difference in performance between the two methods can be attributed to the fact that, in SQSD, the constructed approximations are solved analytically whereas in Dynamic-Q, the solutions to the subproblems are approximated iteratively by the LfopC algorithm. This introduces a stochastic element to the solution procedure. It appears that this stochastic strategy is more efficient for the O synthesis problem. A further reason for the rejection of SQSD as the general algorithm of choice for the manipulator optimization problem, is that it is purely a method for unconstrained problems. This makes its use in constrained extensions to the manipulator optimization problem impossible.

All the algorithms tested are capable of solving the manipulator optimization problem given sufficient computing power. The function evaluations, each corresponding to a workspace computation, are however relatively expensive, and will become more so as manipulator dimensionality and complexity increase. Efficiency is thus of primary importance. From the results obtained for the simple manipulator studied here it is expected that, for the more challenging manipulator design problems to be tackled later, Dynamic-Q will also be able to provide the efficiency required, with a minimal compromise on the accuracy of the solution.

4.6 Efficient inclusion of a prescribed workspace

The synthesis problem studied in this section is

> **E synthesis:** *determine the manipulator design resulting in a workspace which completely includes the prescribed workspace in the most efficient manner.*

4.6.1 Optimization formulation

One possible way of solving the E synthesis problem is by adjusting the weights w_c and w_p in (4.8). As w_c tends towards 0, δW_c takes on less importance in the optimization problem and the manipulator tends towards a design with a workspace that fully includes the prescribed workspace. The problem with this approach is that when $w_c = 0$, there are clearly an infinite number of solutions yielding the global optimum value $f^* = 0$, with as many of them corresponding to designs with unnecessarily large workspaces. This problem may be overcome by considering the following alternative *exact fit* constrained optimization problem:

$$\begin{aligned} &\min_{\mathbf{d}} f(\mathbf{d}) = \delta W_c \\ &\text{subject to the equality constraint} \\ &h(\mathbf{d}) = \delta W_p = 0 \end{aligned} \qquad (4.14)$$

	N^g	f^*	h^*	$\mathbf{d}^*$
SP1	253	2.88511	$0.7E-2$	$[-4.152, -2.461, 4.060, 5.491, 5.441]^T$
SP2	573	2.85018	$0.3E-2$	$[-4.570, -2.203, 4.545, 5.542, 5.527]^T$
SP3	323	3.03526	$0.3E-2$	$[-4.023, -2.518, 4.235, 5.468, 5.563]^T$
SP4	402	3.15618	$0.1E-2$	$[-4.269, -2.567, 3.986, 5.618, 5.481]^T$

Table 4.7: E synthesis solutions

A solution to this problem corresponds to a manipulator design $\mathbf{d}^*$ with a corresponding workspace that fully encloses the prescribed workspace in an optimal manner, as required by the E synthesis statement.

4.6.2 Numerical results

The constrained optimization problem (4.14) was solved using the Dynamic-Q algorithm, with the same prescribed workspace (4.11) and starting points (Table 4.2 and Figure 4.4) as for the unconstrained problem. Termination parameters used for Dynamic-Q were $\varepsilon_f = 10^{-6}$ and $\varepsilon_x = 10^{-4}$. The results obtained for the different runs are summarized in Table 4.7 which gives the number of gradient evaluations (N^g), final objective f^* and equality constraint h^* function values, as well as the final design vector. Figure 4.10 shows the workspaces corresponding to the solutions obtained. For the results presented here, a chord length of $d = 0.1$ was used with number of boundary points $n_{bc} \approx 124$ at the solution. The finite difference interval for determining function gradients was $\Gamma = 10^{-3}$. Figure 4.11 gives the convergence histories from the four starting designs.

From Table 4.7 and Figure 4.10 it can be seen that the final workspaces obtained are grouped more closely together than for the unconstrained problem. This is to be expected with the introduction of the equality constraint. Of interest as well is that, although the resultant optimum designs and cor-

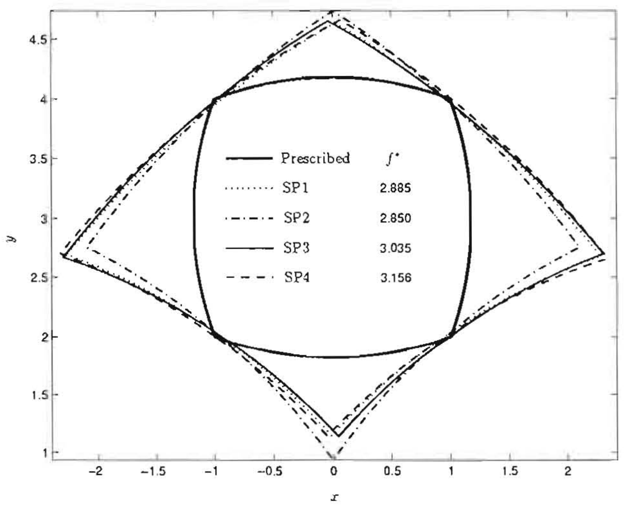

Figure 4.10: E synthesis workspaces corresponding to optimal designs

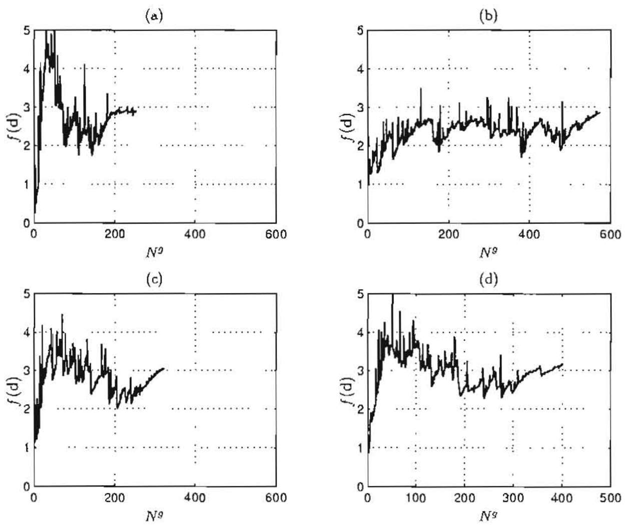

Figure 4.11: E synthesis convergence histories

responding workspaces for SP1 and SP2 are quite different from each other, their function values differ by less than 1%. This once more reinforces the previous conclusion that the objective functions considered here have relatively large flat regions near the optimum solution.

It is evident from inspection of Table 4.7 that the equality constraint h is not accurately satisfied at the converged solutions. This inaccuracy, and relatively slow and erratic convergence (see Figure 4.11) compared to that previously experienced (in Chapter 3), can be attributed to three possible factors. The first of these factors is the discrete nature of the workspace calculation, which may cause numerical noise in the problem. Indeed, as the chord length used in the workspace calculation is reduced, the equality constraint is more accurately satisfied, but at increased computational cost. The second factor influencing the poor convergence is the topography of the optimization problem near the solution. In particular, the choice that $l_i^{\max} = 1.5 l_i^{\min}, i = 1, 2$ seems to result in a flat-bedded, steep-sided valley near the solution, which in turn inhibits sharp convergence of the algorithm to the solution. A third possible reason for the erratic convergence behavior may be the obvious discontinuous nature of the gradient vector of $h(\mathbf{d})$ as design points at which exact fit is achieved are approached from design regions where $\delta W_p > 0$. This discontinuous nature cannot be modelled by the continuous quadratic approximations to $h(\mathbf{d})$ used by the Dynamic-Q algorithm.

4.7 Synthesis with respect to a performance measure

The final methodology proposed in this chapter is:

P synthesis: *determine a manipulator design such that a prescribed workspace is fully enclosed by the manipulator workspace,* *and that the behavior is optimal, with respect to some performance measure, within the workspace.*

4.7.1 Optimization formulation

Further investigation of the numerical ill-conditioning reported in Section 4.6.2, reveals that the dominant factor in introducing the poor and erratic convergence, is probably the the imposition of the leg-ratio equality constraint. It is concluded that this constraint is undesirable and in most design situations probably unnecessary. On the other hand it appears that the discrete manner in which the functions are computed does not seriously affect the conditioning of the problem. It is evident that for any prescribed workspace there are an infinite number of manipulator designs, even when restricted by some leg length ratio, which will result in a workspace that fully includes the prescribed one. Thus far, the criterion for choosing the most suitable design for the constrained cases was that the non-intersecting part of the calculated workspace area should be a minimum. As mentioned in the introduction, there are however a number of other factors, such as manipulator stiffness or conditioning of the workspace, which may be of even greater importance in parallel manipulator design. A possibly better approach to practical design may be, depending on the application, the maximization of, for example, the overall manipulator stiffness, subject to the constraint that the workspace of the optimal manipulator should include the prescribed workspace. In this respect the motivation for method presented here is similar to the Quaternion and Democrat methodologies discussed in Section 1.4.2. These methods determine the *set* of manipulator designs, the workspaces of which include prescribed points, or line segments. The most suitable manipulator with respect to some performance criterion or criteria can then be selected from this set. In this section a similarly motivated approach is developed and applied to the 2-dof planar parallel manipulator, although in this case the optimum manipulator is determined in one step by simultaneously considering performance and workspace.

Thus the constrained optimization formulation, proposed here for the solution of the P synthesis problem, is aimed at optimizing manipulators with respect to some suitable performance index κ, which is to be reduced in an overall sense for the optimum performance of the manipulator, i.e.:

$$\begin{aligned} &\min_{\mathbf{d}} \left\{ \max_{\mathbf{u}\in W_p} \kappa(\mathbf{d},\mathbf{u}) \right\} \\ &\text{subject to the inequality constraint} \\ &g(\mathbf{d}) \leq 0 \end{aligned} \tag{4.15}$$

where for a design $\mathbf{d}$, κ may be measured at any point $\mathbf{u}$ within the prescribed workspace W_p, and the inequality constraint function $g(\mathbf{d})$ is defined slightly differently to the equality constraint function $h(\mathbf{d})$ in (4.14). The displacement vector between the prescribed workspace boundary and calculated workspace boundary, measured along a ray emanating from $\mathbf{O}'$ at angle β_c^i is denoted by $r^i\mathbf{e}^i$, where $\mathbf{e}^i$ is a unit outward vector at angle β_c^i. If $r^{\min} = \min_i \{|r^i|, i = 1,2,\ldots,n_{bc}\}$ set $r = r^{\min}$. The constraint function is now defined as follows:

$$g(\mathbf{d}) = \begin{cases} \delta W_p \text{ if } \delta W_p > 0 \\ -r^2 \text{ if } \delta W_p = 0 \end{cases} \tag{4.16}$$

This modification is made in order to improve the topography of the inequality constraint function by effectively avoiding the severe discontinuity in the gradient of $g(\mathbf{d})$, at the point of exact fit, that was previously present in the gradient of $h(\mathbf{d})$. It is believed that the severe discontinuities inherent in $h(\mathbf{d})$ adversely affected the performance of the Dynamic-Q algorithm (see last paragraph in Section 4.6.2).

The solution to optimization problem (4.15) seeks to improve the *single worst point* with respect to some chosen performance measure κ, within the *prescribed* workspace, W_p. This philosophy differs from that proposed by Gosselin and Angeles [74] and used by numerous other researchers where the *average* performance index over the *entire* workspace is optimized. Since it is assumed that the manipulator's movements will be limited to the prescribed workspace, it is only necessary to ensure good performance qualities within this workspace and thus it is better here to optimize the single worst value, instead of the average.

4.7.2 The condition number of the manipulator

The specific performance measure chosen here is the condition number of the Jacobian matrix of the manipulator, although any of the other criteria mentioned in Section 1.4.1 could also be used. The accuracy of control of the manipulator is dependent on the condition number (Gosselin and Angeles [74]). Since this quantity tends to infinity as the manipulator approaches a singular position, minimizing the condition number also ensures that the manipulator remains far away from such singular positions. For a general parallel manipulator the inverse kinematics are easy to solve.

From (4.3), an inverse transformation relating the input and output velocities can be determined:

$$\mathbf{J_u}\dot{\mathbf{u}} = -\mathbf{J_v}\dot{\mathbf{v}} \tag{4.17}$$

where $\mathbf{J_u}$ and $\mathbf{J_v}$ are the respective constraint Jacobian matrices, containing the partial derivatives of the m kinematic constraints (4.3) with respect to the variables $\mathbf{u}$ and $\mathbf{v}$. Equation (4.17) can be rewritten as

$$\mathbf{J}\dot{\mathbf{u}} = \dot{\mathbf{v}} \tag{4.18}$$

where $\mathbf{J} = -\mathbf{J_v}^{-1}\mathbf{J_u}$.

In general, the condition number κ of an $n \times n$ Jacobian $\mathbf{J}$ is defined as

$$\kappa = \|\mathbf{J}\|\|\mathbf{J}^{-1}\| \tag{4.19}$$

where $\|\cdot\|$ denotes any norm of its matrix argument. The norm adopted here is the same as that used by Gosselin and Angeles [74], namely

$$\|\mathbf{J}\| = \sqrt{\text{trace}(\mathbf{J}\mathbf{W}\mathbf{J}^{\mathrm{T}})} \tag{4.20}$$

where $\mathbf{W}$ is defined as n^{-1} multiplied by the $n \times n$ identity matrix. The lower the condition number, the better the behavior of the manipulator, with the lowest possible value of κ being unity. The value of κ^{-1}, the *inverse* of the condition number, thus lies between 0 and 1, and is preferably used in the objective function as it is bounded and better conditioned than the condition number itself. For the 2-dof parallel manipulator studied here $n = m = 2$ (see Section 4.3).

An optimization problem equivalent to (4.15) above is therefore:

$$\begin{aligned} &\max_{\mathbf{d}} \left\{ \min_{\mathbf{u}\in W_p} \kappa^{-1}(\mathbf{d},\mathbf{u}) \right\} \\ &\text{subject to the inequality constraint} \\ &g(\mathbf{d}) \leq 0 \end{aligned} \tag{4.21}$$

where $g(\mathbf{d})$ is defined as in (4.16).

One point which arises concerns the nested part of optimization problem (4.21) and the question of how to determine the smallest value of κ^{-1} over the set $\mathbf{u} \in W_p$. Since we only require the single lowest value of the inverse condition number, an efficient method for determining this value, based on the necessary condition for an internal maximum or minimum of the condition number is proposed and used here.

The Jacobian $\mathbf{J}$ of the 2-dof manipulator ($nv = nu = 2$, $nw = 0$) shown in Figure 4.1 is given by

$$\mathbf{J} = \begin{bmatrix} (u_1 - x_A)/v_1 & (u_2 - y_A)/v_1 \\ (u_1 - x_B)/v_2 & (u_2 - y_B)/v_2 \end{bmatrix} \tag{4.22}$$

Using (4.19) and (4.20), and assuming $y_A = y_B$ the inverse condition number

κ^{-1} of the manipulator can be determined (see Appendix D):

$$\kappa^{-1}(\mathbf{u}) = \det(\mathbf{J}) = \frac{(u_2 - y_A)(x_B - x_A)}{v_1(\mathbf{u})v_2(\mathbf{u})} \tag{4.23}$$

The necessary condition for an internal maximum or minimum of the function $\kappa^{-1}(\mathbf{u})$ is that $\nabla\kappa^{-1}(\mathbf{u}) = \mathbf{0}$. It can be shown that the only set of solutions corresponding to this condition is

$$\mathbf{U} = \left\{\mathbf{u} \in \Re^{nu} \middle| \, [u_1 - (x_A + x_B)/2]^2 + [u_2 - y_A]^2 = [(x_A - x_B)/2]^2\right\} \tag{4.24}$$

which is the set of points on a circle of radius $(x_A - x_B)/2$ centered at $[(x_A+x_B)/2, y_A]^{\mathrm{T}}$. It can be shown that for these points $\kappa^{-1}(\mathbf{u}) = 1$, which is known to be the maximum possible value of κ^{-1}. The set $\mathbf{U}$ thus corresponds to a "maximum ridge" of the inverse condition number κ^{-1}. Now assume that there exists a point $\mathbf{u}' \in \Re^{nu}$ such that $\kappa^{-1}(\mathbf{u}')$ is a minimum. Since κ^{-1} is continuous for $y_P > y_A$ it is then necessary that $\nabla\kappa^{-1}(\mathbf{u}') = \mathbf{0}$. It has however been shown that the only solutions corresponding to the necessary conditions in the upper plane are elements of $\mathbf{U}$ and they are maxima. Thus the minimum value of κ^{-1} must lie on the boundary ∂W_p of the prescribed workspace W_p. A complete version of this proof is given in Appendix D.

The maximum value of the condition number can thus be approximated by calculating κ at points $\mathbf{b}_p^i$, $i = 1, \ldots, n_{bc}$ and then determining the overall maximum of the values at these candidate points. This of course also gives the corresponding overall minimum value of κ^{-1} required in (4.21).

4.7.3 Numerical results

Optimization problem (4.21) is solved for the 2-dof parallel manipulator with three different prescribed workspaces denoted P1-P3. Here it is assumed that the actuator sizes are fixed with $l_i^{\min} = 4.0, i = 1, 2$ and $l_i^{\max} = 7.0, i = 1, 2$. There are thus three design variables and the design vector is $\mathbf{d} = [x_A, y_A, x_B]^{\mathrm{T}}$. P1 is the workspace studied in the previous sections and is

$[\beta_{p\,\min}, \beta_{p\,\max})$	dx	dy	R	$\pm$
$[0, \pi/2)$	0.0000	0.0000	2.0000	$+$
$[\pi/2, \pi)$	-1.5000	0.0000	2.5000	$+$
$[\pi, 1.444\pi)$	2.7500	2.2500	2.8504	$-$
$[1.444\pi, 1.555\pi)$	-0.07005	0.86647	0.29206	$-$
$[1.555\pi, 2\pi)$	-2.3125	2.500	2.5195	$-$

Table 4.8: Parameters specifying prescribed workspace P3

described by (4.11). P2 is an ellipse centered at $\mathbf{O}' = [0, 3]^\mathrm{T}$ with x and y half axis lengths $a = 1.75$ and $b = 1.00$. P3 is a non-convex, non-symmetrical workspace centered at $\mathbf{O}' = [0, 3]^\mathrm{T}$ and defined in polar coordinates by

$$\begin{aligned} r_p(\beta_p) &= -b \pm \sqrt{b^2 - dx^2 - dy^2 + R^2} \\ &\text{where } b = dx \cos \beta_p + dy \sin \beta_p \end{aligned} \tag{4.25}$$

and where the parameters dx, dy, R and the sign before the square root are, for various angular intervals $[\beta_{p\,\min}, \beta_{p\,\max})$, as given in Table 4.8. The boundary thus consists of five smooth arcs.

The three prescribed workspaces, as well as the workspace for the starting design $\mathbf{d}^0 = [-4, -0.1, 4]^\mathrm{T}$ are shown in Figure 4.12(a). Contours of the reciprocal of the condition number κ^{-1} corresponding to the starting design are also plotted. Optimization problem (4.21) was implemented using the Dynamic-Q algorithm with move limit $\rho = 0.2$ and chord length of $d = 0.1$ for calculating the workspace. Termination criteria $\varepsilon_f = 10^{-4}$ and $\varepsilon_x = 10^{-3}$ were used. For each prescribed workspace P1-P3, Table 4.9 summarizes the number of gradient evaluations N^g required to reach the optimum solution f^* from the starting function value $f(\mathbf{d}^0)$, as well as the value of the inequality constraint function g^* at the solution and the solution vector $\mathbf{d}^*$. Workspaces corresponding to the solutions along with plots of the reciprocal of the condition number are given in Figures 4.12(b) and 4.13. Convergence histories corresponding to Table 4.9 are given in Figure 4.14.

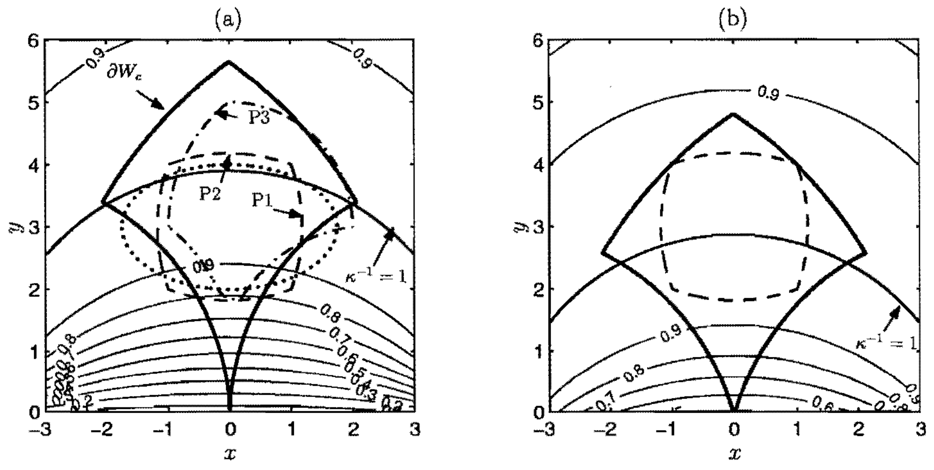

Figure 4.12: P synthesis (a) prescribed workspaces P1-P3, manipulator workspace and κ^{-1} contours corresponding to the starting design and (b) prescribed workspace P1 and corresponding optimal manipulator workspace and κ^{-1} contours

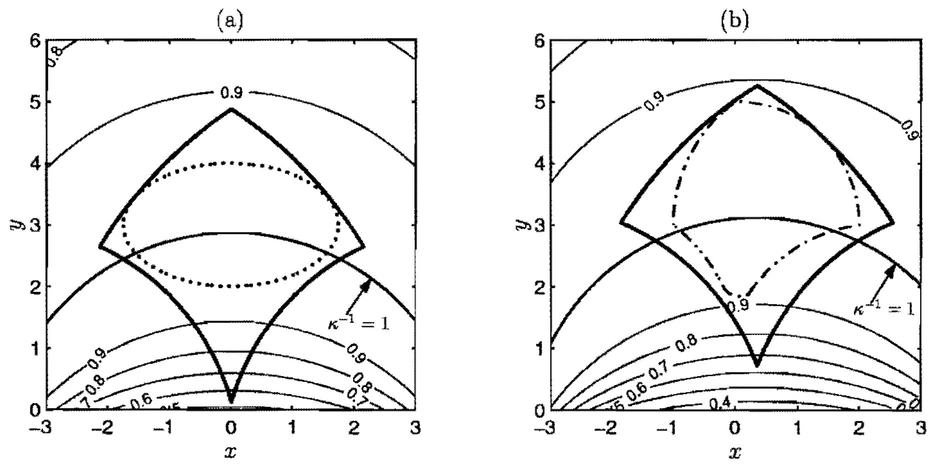

Figure 4.13: P synthesis manipulator workspace and κ^{-1} contours corresponding to the optimal design for prescribed workspaces (a) P2 and (b) P3

	N^g	$f(\mathbf{d}_0)$	f^*	g^*	$\mathbf{d}^*$
P1	10	0.780	0.952	$-0.5E-5$	$[-3.892, -1.016, 3.882]^T$
P2	12	0.823	0.968	$-0.2E-5$	$[-3.830, -0.9740, 3.859]^T$
P3	10	0.786	0.921	$-0.5E-4$	$[-3.414, -0.6427, 4.107]^T$

Table 4.9: P synthesis solutions

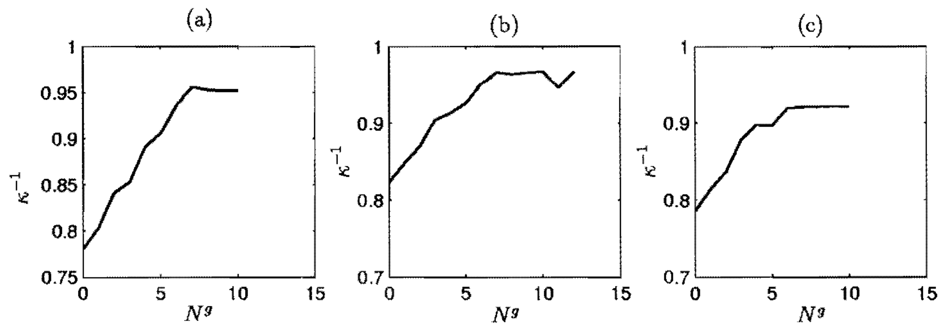

Figure 4.14: P synthesis convergence histories for (a) P1, (b) P2, and (c) P3

The results obtained are extremely encouraging, accurate and optimal solutions having been obtained with minimal computational effort. In each case the algorithm not only determines manipulator dimensions so that the prescribed workspace can be reached by the manipulator, but also places the calculated workspace so that the condition number is as low as possible throughout the prescribed workspace.

4.8 Conclusion

Various computational schemes for synthesizing planar parallel manipulator designs have been proposed, successfully implemented and evaluated. In broad terms, two different criteria are applied in measuring the correspondence. The application of the first criterion is equivalent to seeking a good overall approximation of the prescribed workspace and results in an unconstrained optimization problem. The second criterion requires that the prescribed workspace be fully contained in the optimal manipulator workspace and results in a constrained optimization problem. The unconstrained formulation solving the O synthesis problem has been used to select the most suitable algorithm for this class of problems from four different algorithms developed at the University of Pretoria. The Dynamic-Q method exhibits higher efficiency than the other methods under consideration. In the search for an efficient, stable, well-conditioned and practically relevant method two different formulations of the constrained optimization problem, namely E and P synthesis formulations, were proposed and numerically evaluated. Although the Dynamic-Q algorithm was successfully applied to the constrained optimization problems, its performance, with respect to accuracy and convergence rate, appeared to be seriously affected by the apparent poor conditioning of the E synthesis formulation. This was initially ascribed to both the inherent discrete manner in which the objective and constraint functions are computed and to the topographical ill-conditioning introduced by the imposition of the leg ratio equality constraint. A third likely reason for the erratic convergence behavior is thought to be the discontinuous nature of the prescribed equality constraint. Nevertheless, from an engineering design point of view, good solutions are obtained. Finally, the P synthesis formulation, which attempts to obtain a well-conditioned manipulator workspace which fully contains the prescribed workspace, was proposed. In this formulation the severe discontinuity previously present in the constraint function, which adversely influenced the performance of Dynamic-Q, was effectively removed. This final methodology produces convincing results giving a stable and efficient method for designing 2-dof planar parallel manipulators. Although the search for a fundamentally sound and robust numerical methodology for synthesizing parallel manipulators was restricted to the 2-dof planar case, it nevertheless led to a successful methodology that appears to be general. It seems that that the final P synthesis methodology is possibly the most practically relevant one.

Chapter 5

The three-degree-of-freedom planar parallel manipulator

5.1 Introduction

As an extension to the work presented in the previous chapter, the constrained optimization formulations presented here are aimed at determining 3-dof planar parallel manipulator designs so that a prescribed workspace is fully enclosed and well-conditioned with respect to some performance index. Depending on the particular application, certain manipulator performance criteria may be of more importance than others (see Section 1.4.1). The performance measure used here is the condition number of the manipulator Jacobian matrix, although a number of other performance measures, or a combination of such measures, could also have been used. The optimization method used in performing the optimization is the Dynamic-Q method.

In the next two sections the 3-$R\underline{P}R$ planar parallel manipulator, and the kinematics and determination of the condition number for this manipulator, are presented. The remainder of this chapter then separately deals with the topics of workspace determination and dimensional synthesis of 3-dof parallel manipulators. In Sections 5.4 and 5.5 the chord workspace determination methodology is extended to the determination of constant orientation and dextrous workspaces of planar 3-$R\underline{P}R$ manipulators.

The P synthesis methodology developed in Chapter 4 is then applied to the 3-dof planar manipulator. Three forms of the dimensional synthesis problem are proposed and implemented. These forms differ from each other in the way that the orientational ability of the 3-dof platform is accounted for. Respectively, the dimensional synthesis is performed for a *single constant orientation* workspace (SO synthesis), *multiple constant orientation* workspaces (MO synthesis), and for a *dextrous* workspace (D synthesis). These methodologies and are discussed in Sections 5.6 to 5.8.

5.2 The three-degree-of-freedom parallel manipulator

The manipulator considered in this chapter is the 3-dof planar parallel mechanism shown in Figure 5.1. The manipulator consists of a platform of length $2r$ connected to a base by three linear actuators, which control the three output degrees of freedom of the platform. The actuators have leg lengths l_1, l_2 and l_3 and are joined to the base and platform by means of revolute joints identified by the letters $A-E$. It will be assumed that $y_C = y_D = y_E$. The coordinates of point P, the mid-point of the platform, are (x_P, y_P) and the orientation of the platform is ϕ_P. With reference to the definitions given in Section 4.2, the actuator leg lengths are the input variables, i.e. $\mathbf{v} = [l_1, l_2, l_3]^T \in \Re^3$. The global coordinates of the working point P form the output coordinates, i.e. $\mathbf{u} = [x_P, y_P]^T \in \Re^2$. In contrast with the 2-dof manipulator considered in the previous chapter, the 3-dof manipulator may, in addition to positioning P in the $x - y$ plane, be orientated at an angle ϕ_P by controlling the three

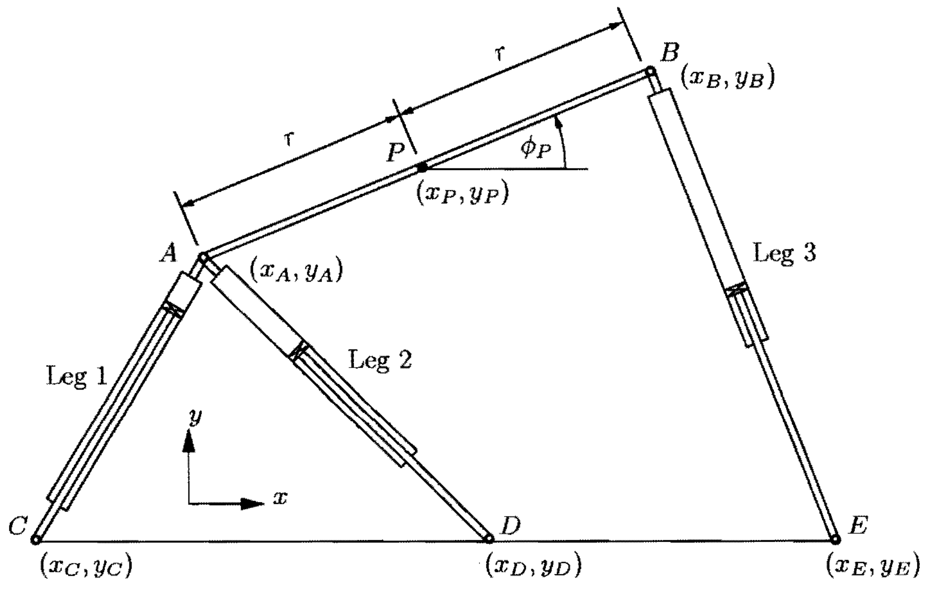

Figure 5.1: The 3-dof parallel manipulator

leg lengths. It is evident that this manipulator thus has three degrees of freedom. The rotation angle of the platform is considered as an intermediate coordinate $w = \phi_P$. For the 3-dof manipulator $nu = 2, nv = 3$ and $nw = 1$. The generalized coordinates for this platform are therefore given by

$$\mathbf{q} = [\mathbf{u}^\top, \mathbf{v}^\top, w]^\top = [x_P, y_P, l_1, l_2, l_3, \phi_P]^\top \in \Re^6 \tag{5.1}$$

In the vicinity of an assembled configuration the input, output and intermediate coordinates satisfy the m independent kinematic constraint equations of the form

$$\mathbf{\Phi}(\mathbf{q}) = \mathbf{\Phi}(\mathbf{u}, \mathbf{v}, w) = \mathbf{0} \tag{5.2}$$

For the 3-dof planar parallel manipulator, $m = 3$.

In general, factors imposed by the physical construction of the planar parallel manipulator, which limit the workspace, may be related to the input variables or a combination of input, output and intermediate variables. An example of former type for the planar parallel manipulator are leg length limits, and of the latter, limits on the angular displacement of the revolute joints connecting the legs to the ground and to the platform. These limiting factors are described by means of inequality constraints and may respectively take the general forms

$$\mathbf{v}^{\min} \leq \mathbf{v} \leq \mathbf{v}^{\max} \tag{5.3}$$

$$\mathbf{g}^{\min} \leq \mathbf{g}(\mathbf{u}, \mathbf{v}, w) \leq \mathbf{g}^{\max} \tag{5.4}$$

Limits on the platform orientation (intermediate coordinate) take one of two forms given by

$$w^{\min} \leq w \leq w^{\max} \tag{5.5}$$

$$\text{or } w = w^{\text{fix}} \tag{5.6}$$

where w^{fix} is a prescribed fixed scalar quantity.

The above definitions are necessary in order to facilitate the mathematical description of kinematics and workspaces types of the 3-dof planar parallel manipulator.

5.3 The kinematics and condition number of the manipulator

In general, the parallel manipulator inverse kinematics are easy to solve. For the manipulator under consideration, the three leg lengths are given by

$$\begin{aligned} l_1^2 &= (x_P - r\cos\phi_P - x_C)^2 + (y_P - r\sin\phi_P - y_C)^2 \\ l_2^2 &= (x_P - r\cos\phi_P - x_D)^2 + (y_P - r\sin\phi_P - y_D)^2 \\ l_3^2 &= (x_P + r\cos\phi_P - x_E)^2 + (y_P + r\sin\phi_P - y_E)^2 \end{aligned} \tag{5.7}$$

Writing in the standard form of the kinematic constraint equations (5.2) and using the coordinates definitions from the previous section, (5.7) become

$$\begin{aligned}\mathbf{\Phi}(\mathbf{u},\mathbf{v},w) &= \begin{bmatrix} v_1^2-(u_1-r\cos w-x_C)^2-(u_2-r\sin w-y_C)^2 \\ v_2^2-(u_1-r\cos w-x_D)^2-(u_2-r\sin w-y_D)^2 \\ v_3^2-(u_1+r\cos w-x_E)^2-(u_2+r\sin w-y_E)^2 \end{bmatrix} \\ &= \mathbf{0}\end{aligned} \tag{5.8}$$

The explicit expressions for $\mathbf{v}$ in terms of $\mathbf{u}$ and w, $\mathbf{v}=\mathbf{v}(\mathbf{u},w)$, may be determined from (5.7), allowing constraints (5.3) to be written as follows:

$$\mathbf{v}^{\min} \leq \mathbf{v}(\mathbf{u},w) \leq \mathbf{v}^{\max} \tag{5.9}$$

where $\mathbf{v}^{\min}=[l_1^{\min},l_2^{\min},l_3^{\min}]^{\mathrm{T}}$ and $\mathbf{v}^{\min}=[l_1^{\max},l_2^{\max},l_3^{\max}]^{\mathrm{T}}$.

As in Chapter 4, the specific performance used here to characterize the performance of the 3-dof planar parallel manipulator is the inverse of the condition number of the Jacobian matrix of the manipulator. The accuracy of control of the manipulator is dependent on the condition number, denoted here by κ. Since κ tends to infinity as the manipulator approaches a singular position, maximizing the inverse condition number, κ^{-1}, also ensures that the manipulator remains far away from singular positions. From (5.2), an inverse transformation relating the input, output and intermediate velocities can be determined:

$$\mathbf{J}_{\boldsymbol{\theta}}\dot{\boldsymbol{\theta}} = -\mathbf{J}_{\mathbf{v}}\dot{\mathbf{v}} \tag{5.10}$$

where $\boldsymbol{\theta}=[\mathbf{u}^{\mathrm{T}},w]^{\mathrm{T}}$, and $\mathbf{J}_{\boldsymbol{\theta}}$ and $\mathbf{J}_{\mathbf{v}}$ are the respective constraint Jacobian matrices containing the partial derivatives of the m kinematic constraints (5.2) with respect to the variables $\boldsymbol{\theta}$ and $\mathbf{v}$. Equation (5.10) can be rewritten as

$$\mathbf{J}\dot{\boldsymbol{\theta}} = \dot{\mathbf{v}} \tag{5.11}$$

where $\mathbf{J}=-\mathbf{J}_{\mathbf{v}}^{-1}\mathbf{J}_{\boldsymbol{\theta}}$. Recall that for the parallel manipulator studied here $m=n=nv=nu+nw$.

One point now arises due to the platform's orientational ability. In contrast to the the 2-dof case, the Jacobian of the manipulator contains entries related to both positional and rotational abilities of the platform. The condition number will thus inherently contain a mix of these terms. It is important to normalize the positional terms of the Jacobian matrix so that positional and rotational abilities are equally represented by the condition number. Pittens and Podhorodeski [71] and Stoughton and Arai [78] note this occurrence and suggest that the best approach is to normalize the positional terms of the Jacobian with respect to the platform radius r, a suggestion which is adopted here.

In explicit terms, differentiation, with respect to time, of the kinematic constraints (5.2), and writing in the form (5.10) yields

$$\begin{bmatrix} x_{AC} & y_{AC} & rx_{AC}\sin w - ry_{AC}\cos w \\ x_{AD} & y_{AD} & rx_{AD}\sin w - ry_{AD}\cos w \\ x_{BE} & y_{BE} & -rx_{BE}\sin w + ry_{BE}\cos w \end{bmatrix} \begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \dot{w} \end{bmatrix} = \begin{bmatrix} v_1 & 0 & 0 \\ 0 & v_2 & 0 \\ 0 & 0 & v_3 \end{bmatrix} \begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \\ \dot{v}_3 \end{bmatrix} \tag{5.12}$$

where the notation $x_{AB} = x_A - x_B$ is used, and $x_A = u_1 - r\cos w$, $y_A = u_2 - r\sin w$, $x_B = u_1 + r\cos w$ and $y_B = u_2 + r\sin w$.

The Jacobian $\mathbf{J}$ of the 3-dof planar manipulator, as defined by (5.11), is thus given by

$$\mathbf{J} = \begin{bmatrix} x_{AC}/rv_1 & y_{AC}/rv_1 & (rx_{AC}\sin w - ry_{AC}\cos w)/v_1 \\ x_{AD}/rv_2 & y_{AD}/rv_2 & (rx_{AD}\sin w - ry_{AD}\cos w)/v_2 \\ x_{BE}/rv_3 & y_{BE}/rv_3 & (-rx_{BE}\sin w + ry_{BE}\cos w)/v_3 \end{bmatrix} \tag{5.13}$$

Note the normalization of the positional terms in the first two columns by the platform radius r. The condition number κ of this 3×3 Jacobian may be determined using equations (4.19) and (4.20) of Section 4.7.2.

5.4 Constant orientation workspace determination

5.4.1 Workspace definition

The constant orientation workspace associated with a fixed value $w = w^{\text{fix}}$ of the intermediate variable, in the form of (5.6), is denoted $W^C[w^{\text{fix}}]$. In agreement with the definition given in Section 1.3.1, the constant orientation workspace of the 3-dof manipulator can be defined mathematically as

$$\begin{aligned} W^C[w^{\text{fix}}] = \{\mathbf{u} \in \Re^{nu} : \mathbf{\Phi}(\mathbf{u},\mathbf{v},w) = \mathbf{0},\ \text{with } \mathbf{v} \text{ satisfying (5.3)},\\ \mathbf{g}(\mathbf{u},\mathbf{v},w) \text{ satisfying (5.4) and } w \text{ satisfying (5.6)}\} \end{aligned} \tag{5.14}$$

Intuitively the boundary $\partial W^C[w^{\text{fix}}]$ of the constant orientation workspace may be defined as

$$\begin{aligned} \partial W^C[w^{\text{fix}}] = \Big\{\mathbf{u} \in \Re^{nu} : \mathbf{u} \in W^C[w^{\text{fix}}] \text{ and } \exists \text{ an } \mathbf{s} \in \Re^{nu} \text{ such that for}\\ \mathbf{u}' = \mathbf{u} + \lambda\mathbf{s},\ \lambda \in \Re \text{ arbitrarily small and either}\\ \text{positive or negative, no } \mathbf{v} \text{ exists that satisfies}\\ \mathbf{\Phi}(\mathbf{u}',\mathbf{v},w) = \mathbf{0};\ \text{as well as inequalities (5.3) and (5.4),}\\ \text{and equality constraint (5.6)}\Big\} \end{aligned} \tag{5.15}$$

5.4.2 Mapping the constant orientation workspace boundary

The boundary of the constant orientation workspace may be mapped numerically by means of the chord method. The basic methodology remains the same as described in Appendix C, however the precise optimization problems used to determine points on the workspace boundary, differ for two reasons from those given by (C.9) and (C.12).

The first reason for the difference is that it has been noted that the optimization problems, used to determine successive points on the workspace boundary, can be solved much more efficiently by reducing the number of optimization variables. In the original form of both the ray and chord methods for maximal workspace determination, the output and intermediate coordinates of the manipulator were the optimization variables. For planar parallel manipulators, the resulting optimization problems thus contained *three* variables, and *one* equality constraint dictating the direction in which the next boundary point was determined. It is possible, however, to enforce the equality constraint explicitly and analytically in the optimization problem, resulting in a reduction by one of the number of required optimization variables. The resulting increase in efficiency is a result both of this, and the elimination of the numerical equality constraint and associated equality gradient function evaluations.

The second reason for the different form of the optimization problems, of course, is that the platform orientation (the intermediate coordinate) is now fixed for the constant orientation workspace. This again reduces the number of optimization variables by one, since this requirement can also be explicitly and analytically enforced in the optimization problem.

For these reasons, the precise forms of optimization problems (C.9) and (C.12) for constant orientation workspace determination are as follows. Given a radiating point $\mathbf{u}^0$ inside the constant orientation workspace $W^C[w^{\text{fix}}]$, and a search direction specified by a unit vector $\mathbf{s}^1 \in \Re^2$, the output coordinates $\mathbf{u}$ in terms of a scalar r is given by

$$\mathbf{u}(r) = \mathbf{u}^0 + r\mathbf{s}^1 \qquad (5.16)$$

An initial point $\mathbf{b}^1 = \mathbf{u}(r^*)$ on the constant orientation workspace boundary

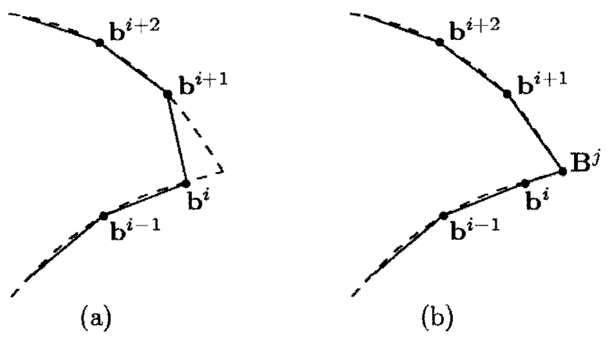

Figure 5.2: The importance of mapping bifurcation points

the two active constraints. This can be easily accomplished by solving the following least squares optimization problem:

$$\min_{\mathbf{u}} \|\mathbf{p}(\mathbf{u}, w^{\text{fix}}) - \mathbf{p}^{\text{ext}}\|^2 \tag{5.20}$$

where $\mathbf{p}$ contains the active set of constraints from (5.3) or (5.4) and $\mathbf{p}^{\text{ext}}$ the corresponding upper or lower limits. Active constraints, and the presence of bifurcation points can be determined by continuously monitoring the values of constraints (5.3) or (5.4) while tracing the workspace boundary. Figure 5.2 illustrates the importance of mapping bifurcation points. Successive points determined along the workspace boundary ∂W^C using the chord method are $\mathbf{b}^{i-1}, \ldots, \mathbf{b}^{i+2}$. It is evident on comparison of Figure 5.2 (a) and (b) that the inclusion of the bifurcation point $\mathbf{B}^j$ results in a much more accurate representation of the workspace boundary.

5.5 Dextrous workspace determination

This section presents a new numerical multi-level optimization methodology for determining dextrous workspaces of planar parallel manipulators. The methodology is based on the chord method discussed in Appendix C, which was extended and refined in the previous section for determining constant orientation workspaces. It should be noted that the method proposed here differs from the optimization method for determining dextrous workspaces proposed by Du Plessis and Snyman [18]).

5.5.1 Workspace definitions

Dextrous workspace

The dextrous requirement for the manipulator at a point $\mathbf{u}$ is that all orientations in the range

$$\phi^{\min} \leq w \leq \phi^{\max} \tag{5.21}$$

can be attained by the manipulator (see Section 1.3.1). The dextrous workspace $W^D[\phi^{\min}, \phi^{\max}]$ of the planar manipulator is thus defined as:

$$\begin{aligned} W^D = \{&\mathbf{u} \in \Re^{nu} : \mathbf{\Phi}(\mathbf{u}, \mathbf{v}, w) = \mathbf{0},\ \text{with } \mathbf{v} \text{ satisfying (5.3)} \\ &\text{and } \mathbf{g}(\mathbf{u}, \mathbf{v}, w) \text{ satisfying (5.4) for all } w \in [\phi^{\min}, \phi^{\max}]\} \end{aligned} \tag{5.22}$$

The boundary ∂W^D of the dextrous workspace can thus be defined as:

$$\begin{aligned} \partial W^D = \Big\{&\mathbf{u} \in \Re^{nu} : \mathbf{u} \in W^D \text{ and } \exists \text{ an } \mathbf{s} \in R^{nu} \text{ such that for} \\ &\mathbf{u}' = \mathbf{u} + \lambda\mathbf{s},\ \lambda \in \Re \text{ arbitrarily small and either positive} \\ &\text{or negative, no } \mathbf{v} \text{ exists that satisfies } \mathbf{\Phi}(\mathbf{u}', \mathbf{v}, w) = \mathbf{0}; \\ &\text{as well as inequalities (5.3) and (5.4) for all } w \in [\phi^{\min}, \phi^{\max}]\Big\} \end{aligned} \tag{5.23}$$

In order to calculate the dextrous workspace, it is necessary to be able to calculate the manipulator orientation workspace, for any given position $\mathbf{u}$, as well.

Orientation workspace

The orientation workspace $W^O[\mathbf{u}^{\text{fix}}]$ of a manipulator, for a fixed position $\mathbf{u}^{\text{fix}}$ of the working point, is the set of orientations that can be attained by the manipulator end-effector (see Section 1.3.1). For a planar manipulator, since only one rotation is possible (about the z-axis perpendicular to the plane), the orientation workspace is one-dimensional and can easily be specified by the maximum $w^{\max}$ and minimum $w^{\min}$ orientations attainable by the manipulator end-effector. For a given $\mathbf{u}^{\text{fix}}$, the orientation workspace can thus be described mathematically as

$$\begin{aligned} W^O[\mathbf{u}^{\text{fix}}] = \{ & w \in \Re : \boldsymbol{\Phi}(\mathbf{u}^{\text{fix}}, \mathbf{v}, w) = \mathbf{0} \text{ with } \mathbf{v} \text{ satisfying} \\ & (5.3) \text{ and } \mathbf{g}(\mathbf{u}^{\text{fix}}, \mathbf{v}, w) \text{ satisfying } (5.4)\} \end{aligned} \tag{5.24}$$

The boundary ∂W^O of the orientation workspace for a planar manipulator is thus

$$\begin{aligned} \partial W^O[\mathbf{u}^{\text{fix}}] = \{ & w \in O \text{ and } \exists \text{ a } \lambda \in \Re \text{ such that for } w' = w + \lambda \\ & \text{with } \lambda \text{ arbitrarily small and either positive or} \\ & \text{negative, no } \mathbf{v} \text{ exists that satisfies } \boldsymbol{\Phi}(\mathbf{u}^{\text{fix}}, \mathbf{v}, w') = \mathbf{0} \\ & \text{as well as conditions (5.3) and (5.4)}\} \end{aligned} \tag{5.25}$$

In practical terms, for $\mathbf{u}^{\text{fix}}$, the values of $w^{\min}$ and $w^{\max}$ may easily be obtained numerically by solution of the following optimization problem.

$$\begin{gathered} \max_{w} \, (w - w^{\text{ave}})^2 \\ \text{subject to } \mathbf{v}^{\min} \leq \mathbf{v}(\mathbf{u}^{\text{fix}}, w) \leq \mathbf{v}^{\max}, \\ \mathbf{g}^{\min} \leq \mathbf{g}(\mathbf{u}^{\text{fix}}, \mathbf{v}, w) \leq \mathbf{g}^{\max} \end{gathered} \tag{5.26}$$

where w^{ave} is a suitably chosen value of the manipulator orientation that lies inside the orientation workspace. By choosing a suitable starting point for optimization problem (5.26) the values of $w^{\min}$ and $w^{\max}$, corresponding to the two extreme local minima, can be determined.

5.5.2 Mapping the dextrous workspace boundary

Finding an initial point on the workspace boundary

Finding an initial point on the dextrous workspace boundary requires the sequential solution of three problems:

1. Finding the assembled point $\mathbf{u}^a$ of the manipulator with input coordinates at their average value.
2. Finding the point $\mathbf{u}^d$ where the manipulator has its greatest dextrous ability.
3. Determining an initial point $\mathbf{b}^1$ on the dextrous workspace boundary.

These three steps are realized through the implementation of three different constrained optimization problems, which result in a reliable and automatic determination of the initial boundary point $\mathbf{b}^1$.

Step 1 essentially involves the solution of the forward kinematics of the manipulator, i.e. solve for $\mathbf{u}$ and w in (5.2) with $\mathbf{v}$ prescribed as

$$\bar{\mathbf{v}} = (\mathbf{v}^{\min} + \mathbf{v}^{\max})/2 \tag{5.27}$$

In practice this can be done by solving the least squares optimization problem

$$\min_{\mathbf{u},w} \|\mathbf{v}(\mathbf{u}, w) - \bar{\mathbf{v}}\|^2 \tag{5.28}$$

where $\mathbf{v}(\mathbf{u}, w)$ denotes the inverse kinematic solution of (5.2) for any given $\mathbf{u}$ and w. The solution of this problem yields the correct value for $\mathbf{u}^a$.

The point of greatest dexterity of the manipulator $\mathbf{u}^d$ (Step 2) can similarly be determined by means of the unconstrained optimization problem:

$$\max_{\mathbf{u}} \left(w^{\max}(\mathbf{u}) - w^{\min}(\mathbf{u})\right)^2 \tag{5.29}$$

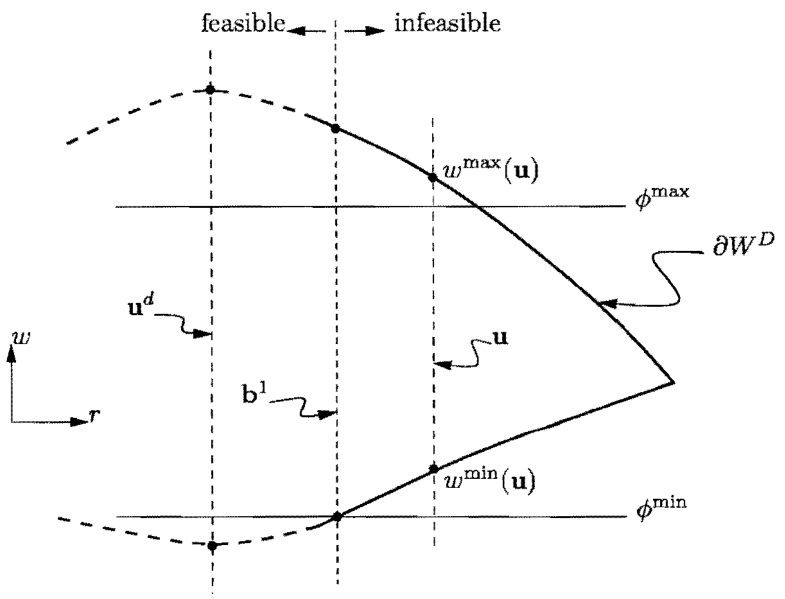

Figure 5.3: Finding an initial point on the dextrous workspace boundary

where the values of $w^{\min}(\mathbf{u})$ and $w^{\max}(\mathbf{u})$ for a fixed $\mathbf{u}$, are determined by solution of optimization problem (5.26), i.e. by determining the orientation workspace of the manipulator at point $\mathbf{u}$.

Consistent with the definition of ∂W^D in (5.23), an initial point $\mathbf{b}^1$ on the dextrous workspace boundary in an arbitrary direction from $\mathbf{u}^d$, designated by a unit vector $\mathbf{s}^1 \in \Re^{nu}$, is determined by solving the following constrained optimization problem (corresponding to optimization problem (C.9) in Appendix C):

$$\begin{aligned} &\max_r r^2 \\ \text{such that } c_1 &= \phi^{\max} - w^{\max}(\mathbf{u}(r)) \leq 0; \\ c_2 &= w^{\min}(\mathbf{u}(r)) - \phi^{\min} \leq 0 \end{aligned} \tag{5.30}$$

where $\mathbf{u}(r) = \mathbf{u}^d + r\mathbf{s}^1$. The solution of this problem is schematically illustrated in Figure 5.3. Once more the the values of $w^{\min}(\mathbf{u})$ and $w^{\max}(\mathbf{u})$ for a point $\mathbf{u}$ in (5.30), are determined by solution of optimization problem (5.26).

Mapping the workspace boundary

Once an initial point on the workspace boundary has been found, subsequent points can be mapped using the chord methodology. The updated form of optimization problem (C.12) is

$$\begin{aligned} &\min_{\omega} \omega^2 \\ \text{such that } c_1 &= \phi^{\max} - w^{\max}(\mathbf{u}(\omega)) \leq 0; \\ c_2 &= w^{\min}(\mathbf{u}(\omega)) - \phi^{\min} \leq 0 \end{aligned} \tag{5.31}$$

where $\mathbf{u}(\omega)$ is as defined in (5.18), and $w^{\min}$ and $w^{\max}$ are determined using optimization problem (5.26), hence the description of the methodology as multi-level. The basic form of the chord methodology remains otherwise the same as described in Appendix C.

5.5.3 Determination of bifurcation points

It is evident that any point on the dextrous workspace boundary will be associated with either a maximum $\phi^{\max}$ or minimum $\phi^{\min}$ orientation of the manipulator (see Figure 5.3). In addition, since the solution to optimization problem (5.26) is implicit in solving (5.31), each boundary point is also associated with an extreme leg value. In fact, the workspace boundary is associated with a number of curves, each corresponding to a different extreme leg value and associated extreme platform orientation. Points where these curves meet are termed *bifurcation points* since the manipulator may assume one of two distinct extreme states when travelling clockwise or counter-clockwise along the workspace boundary from such a point. When determining the dextrous workspace, a distinction must be made between two different types of bifurcation points.

Type I bifurcation points

Type I bifurcation points occur along the workspace boundary when two intersecting boundary curves are both associated with the *same* extreme orientation, either $\phi^{\min}$ *or* $\phi^{\max}$ of the platform. Each curve will additionally be associated with one leg at an extreme length. In the vicinity of the intersection, the precisely active leg, and associated active boundary curve, can be determined by examination of the final values of the constraints from optimization problem (5.31). In this way, at the intersection or bifurcation point, two legs m and n will both be at known extreme values v_m^{ext} and v_n^{ext}, and the platform will be at a known extreme orientation ϕ^{ext}. We need then to simply solve the inverse kinematics to determine the exact coordinates of the type I bifurcation point $\mathbf{B}^j$. This is done by the solution of the following optimization problem:

$$\min_{\mathbf{u}} \left(v_m(\mathbf{u}, \phi^{\text{ext}}) - v_m^{\text{ext}}\right)^2 + \left(v_n(\mathbf{u}, \phi^{\text{ext}}) - v_n^{\text{ext}}\right)^2 \tag{5.32}$$

which will yield the coordinates of the bifurcation point.

Type II bifurcation points

Type II bifurcation points are associated with a change in the active extreme orientation of the platform. Thus at these points the maximum $w^{\max}$ *and* minimum $w^{\min}$ values of platform orientation will both simultaneously be exactly equal to the maximum and minimum prescribed orientation values $\phi^{\max}$ and $\phi^{\min}$. Therefore to determine such points, constraints c_1 and c_2 given in equations (5.30) and (5.31) must both be *exactly* satisfied. This may be accomplished by solving the following unconstrained optimization problem:

$$\min_{\mathbf{u}} \left\{(\phi^{\max} - w^{\max}(\mathbf{u}))^2 + (\phi^{\min} - w^{\min}(\mathbf{u}))^2\right\} \tag{5.33}$$

The complete multi-level optimization algorithm for dextrous workspace determination is summarized in Algorithm 5.1.

Algorithm 5.1 Dextrous workspace determination

1. Determine the assembled point $\mathbf{u}^a$ of the mechanism using optimization problem (5.28).
2. Determine the point of greatest dexterity $\mathbf{u}^d$ using optimization problem (5.29).
3. Determine an initial point on the workspace boundary by means of optimization problem (5.30).
4. Using optimization problem (5.31), determine successive points along the boundary at chord intervals d. Identify and map type I and II bifurcation points as they occur using either optimization problem (5.32) or (5.33).
5. Terminate when condition (C.14) becomes true or when the specified maximum number of iterations is exceeded.

	x_C	y_C	x_D	x_E	r
M1	-1	0	1	2	1
M2	-0.75	0	0.75	1.5	0.75
M2	-0.5	0	0.5	1	0.25

Table 5.1: Geometric parameters for manipulator designs M1-M3

5.5.4 Numerical results

Three different manipulator geometries, denoted M1-M3 are used in illustrating the proposed methodology. The three sets of five parameters defining the three different manipulator designs are given in Table 5.1. Extreme leg lengths for the manipulators considered here are $l_1^{\text{min}} = l_2^{\text{min}} = \sqrt{2}$, $l_3^{\text{min}} = 1$, $l_1^{\text{max}}, l_2^{\text{max}} = 2$, $l_3^{\text{max}} = \sqrt{3}$. The chord algorithm described in Algorithm 5.1, implemented in FORTRAN on a 1.6 GHz Pentium 4 computer, was applied in determining the various dextrous workspaces of these manipulators. The dextrous workspaces were obtained for various ranges of dexterity and are given in Figures 5.4 to 5.6. For the various dextrous workspaces obtained, Table 5.2 gives the number of points determined on each workspace boundary n_b, as well as the computational time t required for computing each dextrous workspace. Investigation of the performance of the algorithm has revealed that a large portion of the computational cost is related to solving optimization problem (5.26), to determine the manipulator orientation workspaces. In many cases, it may be possible to determine such orientation workspaces analytically, which would dramatically reduce the time needed to compute the dextrous workspace. Here though, the numerical approach has been presented since it provides an alternative, generally applicable methodology.

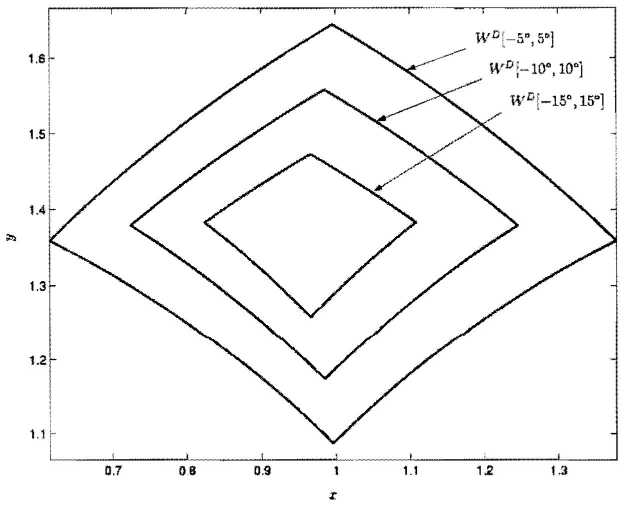

Figure 5.4: Dextrous workspaces of M1

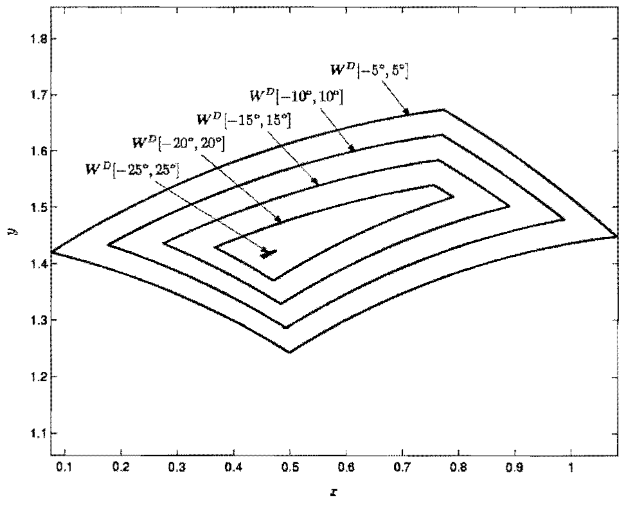

Figure 5.5: Dextrous workspaces of M2

5.6 Optimization for a single prescribed constant orientation workspace

The constrained optimization methodology developed in Section 4.7 is applied here to the 3-dof manipulator in the following form:

> **SO synthesis:** *Determine a manipulator design that reaches, with optimal conditioning, a prescribed constant orientation workspace.*

5.6.1 Optimization formulation

As is Section 4.5.1, the prescribed workspace is defined by polar coordinates (β_p, r_p) centered on a local coordinate system $x' - y'$ at $\mathbf{O}'$. The boundary of the constant workspace $W_c^C[\phi^{\text{fix}}]$ associated with design $\mathbf{d}$ is represented in a similar manner (refer to Figure 4.2). The chord method (see Section 5.4) is used to generate points $\mathbf{b}_c^i$, with corresponding polar coordinates (β_c^i, r_c^i), on the constant orientation workspace boundary.

Dropping the $[\phi^{\text{fix}}]$, which is implicit when referring to constant orientation workspaces W^C for the rest of this section, the part of the prescribed workspace W_p^C not intersecting calculated workspace W_c^C is denoted δW_p^C, and the part of workspace W_c^C not intersecting W_p^C is denoted δW_c^C. The calculation of approximations to the areas δW_p^C and δW_c^C is performed using the numerical scheme described in Section 4.5.1.

SO synthesis is thus achieved by solution of the following optimization problem:

$$\begin{aligned}
&\max_{\mathbf{d}} \left\{ \min_{\mathbf{u} \in W_p^C[\phi^{\text{fix}}]} \kappa^{-1}(\mathbf{d}, \mathbf{u}) \right\} \\
&\text{subject to the inequality constraint} \\
&g(\mathbf{d}) \leq 0
\end{aligned} \tag{5.34}$$

where the intermediate coordinate $w = \phi^{\text{fix}}$ is prescribed and fixed, and where the inequality constraint function $g(\mathbf{d})$ is defined as

$$g(\mathbf{d}) = \begin{cases} \delta W_p^C \text{ if } \delta W_p^C > 0 \\ -r^2 \quad \text{if } \delta W_p^C = 0 \end{cases} \tag{5.35}$$

where r is calculated as before (see Section 4.7). The solution to optimization problem (5.34) seeks to improve the *single worst point* with respect to chosen performance measure, κ^{-1}, within the *prescribed* workspace, W_p^C.

Once again the question of how to determine the smallest value of κ^{-1} over the set $\mathbf{u} \in W_p^C$ arises. It has been shown for the planar 2-dof manipulator, that the maximum value of κ (or minimum of κ^{-1}) will lie on the boundary ∂W_p^C of the prescribed workspace (See Appendix D). An assumption is made that a similar result can be found for the particular 3-dof manipulator to be investigated here. The minimum value of the inverse condition number κ^{-1} can thus be approximated by calculating κ^{-1} at points $\mathbf{b}_p^i$, $i = 1, \ldots, n_{bc}$ simultaneously to the determination of the boundary points $\mathbf{b}_c^i$, $i = 1, \ldots, n_{bc}$. The overall minimum of the κ^{-1} values at these candidate points may then easily be determined. Based on the results presented throughout this chapter it appears that the above assumption is valid.

5.6.2 Numerical results

The method described above, and embodied by optimization problem (5.34), has been applied to the 3-dof planar parallel manipulator for three different prescribed workspaces. These prescribed workspaces are centered at $\mathbf{O}' =$

	N^g	$f(\mathbf{d}^0)$	f^*	g^*	$\mathbf{d}^*$
P1	21	0.759	0.912	$0.2E-5$	$[-1.111, 0.2317, 1.277, 1.967, 0.9159]^T$
P2	17	0.768	0.927	$0.4E-9$	$[-1.153, 0.3003, 1.275, 1.906, 0.9401]^T$
P3	71	0.747	0.943	$0.1E-4$	$[-1.170, 0.3365, 1.463, 1.946, 0.9108]^T$

Table 5.3: SO synthesis solutions

$[1, 1.5]^T$. They are chosen to correspond to workspaces P1-P3 in Section 4.7, scaled down by a factor of 5 for workspaces P1 and P2 and by 7 for P3. This was done so that the prescribed workspaces were of such diameter that feasible solutions for the choice of actuator leg lengths existed. It is assumed the actuators have been chosen and thus that the maximum and minimum leg lengths are predetermined. The remaining five design variables for the problem are thus

$$\mathbf{d} = [x_C, y_C, x_D, x_E, r]^T \tag{5.36}$$

Actuator limits were chosen as $l_i^{\min} = \sqrt{2}, l_i^{\max} = 2,\ i = 1, 2$ and $l_3^{\min} = 1,\ l_3^{\max} = \sqrt{3}$. Actuator leg lengths and the initial design vector $\mathbf{d}^0 = [-1, 0, 1, 2, 1]^T$ were selected to correspond to the manipulator studied by Haug *et al.* [16].

The prescribed workspaces P1-P3, the workspace corresponding to the initial design vector and inverse condition number contours for this startign design are shown in Figure 5.7(a). The Dynamic-Q optimization algorithm (see Chapter 3) was used to perform the optimization with move limit $\rho = 0.1$, termination parameters $\varepsilon_f = 10^{-6}$ and $\varepsilon_x = 10^{-4}$, and a finite difference interval $\Gamma = 10^{-6}$ for calculating the gradients of the optimization functions. The chord length for calculating the workspace was $d = 0.02$. For all manipulators the orientation of the platform was fixed at $\phi_P = 0°$.

Numerical results for each of the runs from the starting point $\mathbf{d}^0$ and for the different prescribed workspaces are reported in Table 5.3, which gives the number of gradient evaluations N^g required for convergence, the initial

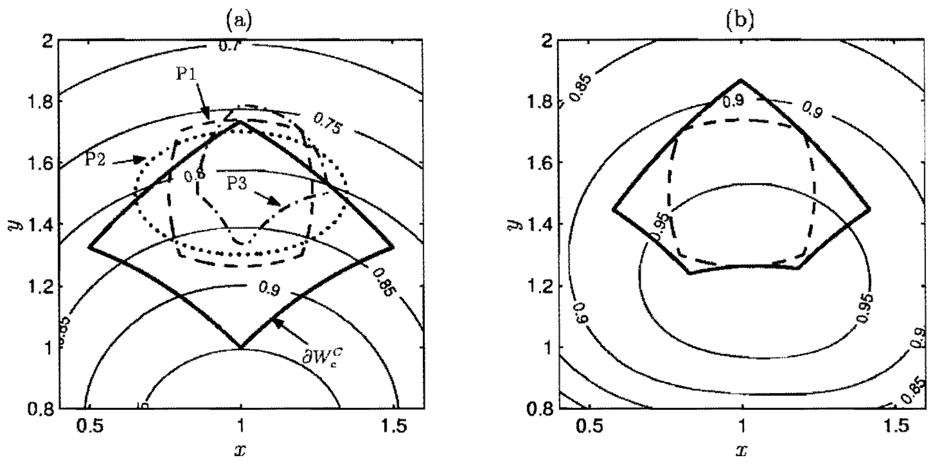

Figure 5.7: SO synthesis (a) prescribed workspaces P1-P3, manipulator workspace and corresponding κ^{-1} contours corresponding to the starting design and (b) prescribed workspace P1 and corresponding optimal manipulator workspace and κ^{-1} contours

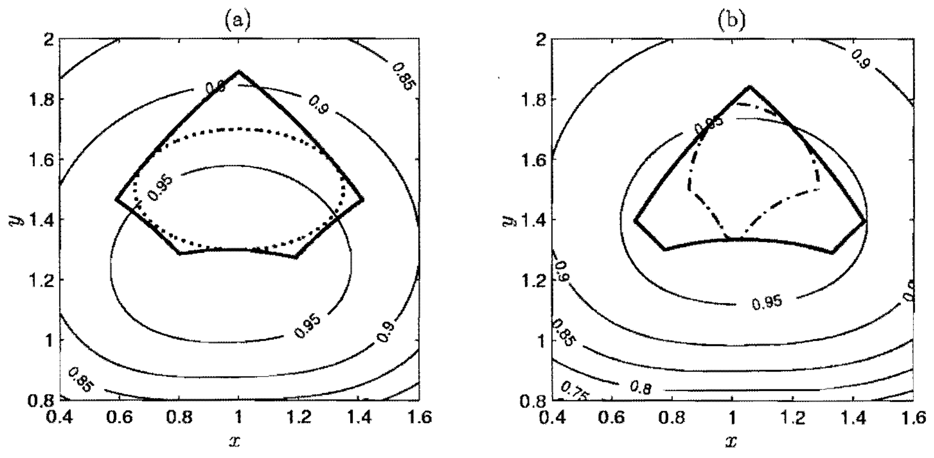

Figure 5.8: SO synthesis manipulator workspace and κ^{-1} contours corresponding to the optimal design for prescribed workspaces (a) P2 and (b) P3

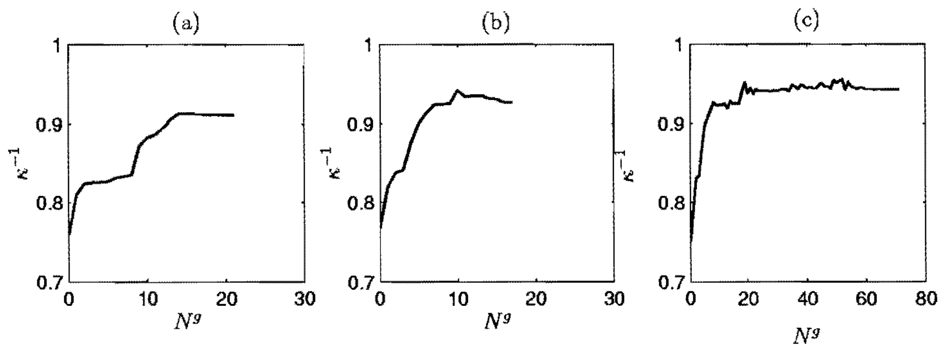

Figure 5.9: SO synthesis convergence histories for (a) P1, (b) P2 and (c) P3

function value $f(\mathbf{d}^0)$, converged objective function value f^*, corresponding inequality constraint function value g^* and components of the optimum design vector $\mathbf{d}^*$. The resultant workspaces and inverse condition number contours corresponding to P1-P3 are shown in Figure 5.7(c) and Figure 5.8(a) and (b) respectively. Finally Figure 5.9 gives the convergence histories for the various prescribed workspaces.

5.7 Optimization for multiple prescribed constant orientation workspaces

MO synthesis: *Determine a manipulator design that reaches, with optimal conditioning, multiple prescribed constant orientation workspaces.*

5.7.1 Optimization formulation

Some strategy needs to be implemented for dealing with the orientational capability of the manipulator. This point is addressed by evaluating the SO optimization problem of the previous section at various angular "slices" through the workspace. This is the approach used by Boudreau and Gosselin [86] in an unconstrained case. Accordingly, in the methodology proposed here, the minimization over $\mathbf{u} = [x, y]^{\mathrm{T}}$ in (5.34) is carried out, not only for a *single* prescribed value of ϕ_P, but over *multiple* slices of the prescribed workspace corresponding to m^{sl} fixed values of ϕ_P. For illustration of the methodology in this section $m^{\mathrm{sl}} = 3$ with slices through the workspace at $\phi_P = \phi^{\mathrm{min}}, \phi^{\mathrm{int}}, \phi^{\mathrm{max}}$. In solving the MO optimization problem the resulting design is expected to fulfil the dexterity requirement of operating over the range of $\phi_P = [\phi^{\mathrm{min}}, \phi^{\mathrm{max}}]$ within the prescribed workspace.

Optimization problem (5.34), modified to allow for optimization over the three values ($m^{\mathrm{sl}} = 3$) of $w = \phi_P$, becomes

$$\begin{aligned} &\max_{\mathbf{d}} \left\{ \min_{\mathbf{u} \in W_p^C[\phi^i],\ i=1,\ldots,m^{\mathrm{sl}}} \kappa^{-1}(\mathbf{d}, \mathbf{u}) \right\} \\ &\text{subject to the inequality constraint} \\ &g(\mathbf{d}) \leq 0 \end{aligned} \tag{5.37}$$

The inequality constraint function is defined as follows:

$$g(\mathbf{d}) = \begin{cases} S = \sum_{i=1}^{m^{\mathrm{sl}}} \delta W_p^C[\phi^i],\ i = 1, \ldots, m^{\mathrm{sl}} & \text{if } S > 0 \\ -r^2 & \text{if } S = 0 \end{cases} \tag{5.38}$$

5.7.2 Numerical results

The prescribed workspaces P1-P3, corresponding to those used in Section 5.6, are shown in Figure 5.10. The manipulator workspaces corresponding to an initial design vector $\mathbf{d}^0 = [-0.75, 0, 0.75, 1.5, 0.75]^{\mathrm{T}}$ for the various constant orientations, as well as the corresponding inverse condition number contours are shown in the same figure. Actuator limits were again chosen as $l_i^{\mathrm{min}} = \sqrt{2}$, $l_i^{\mathrm{max}} = 2, i = 1, 2$ and $l_3^{\mathrm{min}} = 1$, $l_3^{\mathrm{max}} = \sqrt{3}$. The constant orientation slices through the workspace were made at $\phi_P = -5°, 0°, +5°$. Figure 5.10 clearly

	N^g	$f(\mathbf{d}^0)$	f^*	g^*	$\mathbf{d}^*$
P1	33	0.677	0.901	$0.2E-5$	$[-1.034, 0.2484, 1.331, 1.657, 0.8553]^T$
P2	25	0.681	0.917	$0.1E-4$	$[-1.061, 0.2721, 1.345, 1.718, 0.861]^T$
P3	44	0.666	0.915	$0.1E-4$	$[-1.072, 0.3103, 1.420, 1.621, 0.8778]^T$

Table 5.4: MO synthesis solutions

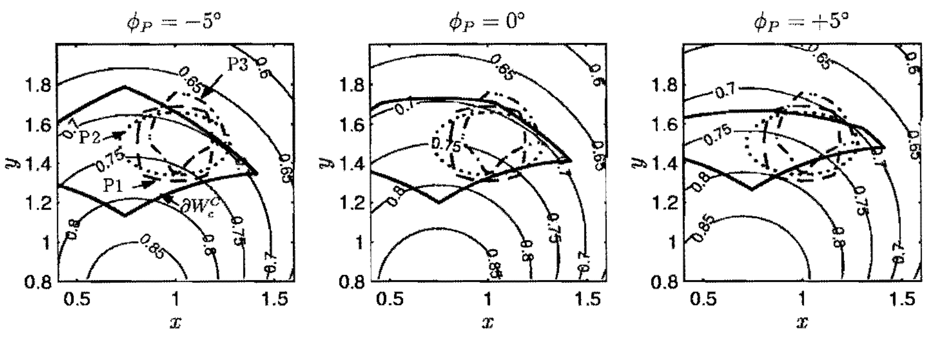

Figure 5.10: MO synthesis prescribed workspaces P1-P3 and manipulator workspace and κ^{-1} contours corresponding to the starting design

shows that the initial design $\mathbf{d}^0$ is infeasible because for each orientational slice the prescribed workspace is not contained in the reachable workspace. The Dynamic-Q optimization algorithm move limit used was $\rho = 0.1$ and the chord length for calculating the workspace was $d = 0.02$. Convergence tolerances used for Dynamic-Q were $\varepsilon_x = 10^{-4}$ and $\varepsilon_f = 10^{-5}$ and a finite difference of $\Gamma = 10^{-6}$ was used for calculating function gradients.

The workspaces, and κ^{-1} contours corresponding to the optimal designs for prescribed workspaces P1-P3 are shown in Figures 5.11 to 5.13. Table 5.4 summarizes the number of gradient evaluations N^g required for convergence, the initial $f(\mathbf{d}^0)$ and final f^* objective function values, inequality constraint function value at convergence c^* and optimal design $\mathbf{d}^*$ for each prescribed workspace. Figure 5.14 shows the convergence histories for the various optimization runs.

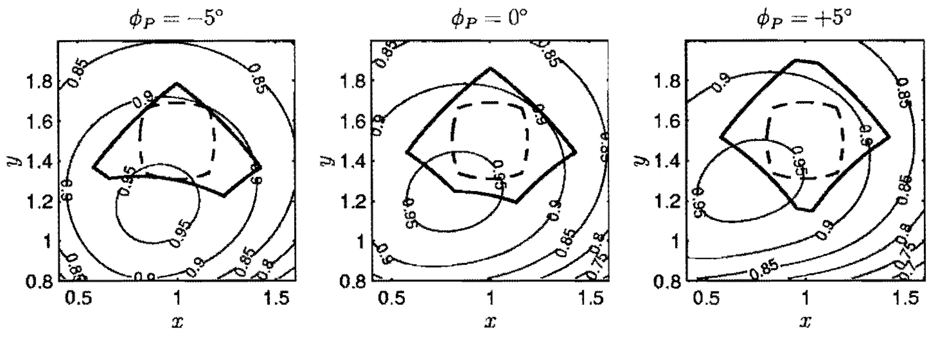

Figure 5.11: MO synthesis manipulator workspace and κ^{-1} contours corresponding to the optimal design for prescribed workspace P1

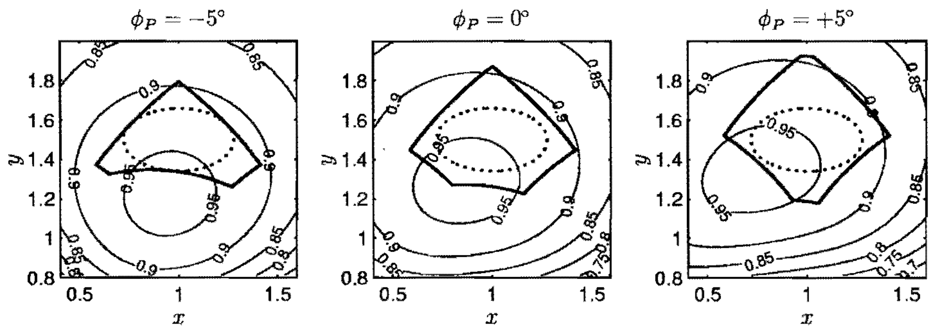

Figure 5.12: MO synthesis manipulator workspace and κ^{-1} contours corresponding to the optimal design for prescribed workspace P2

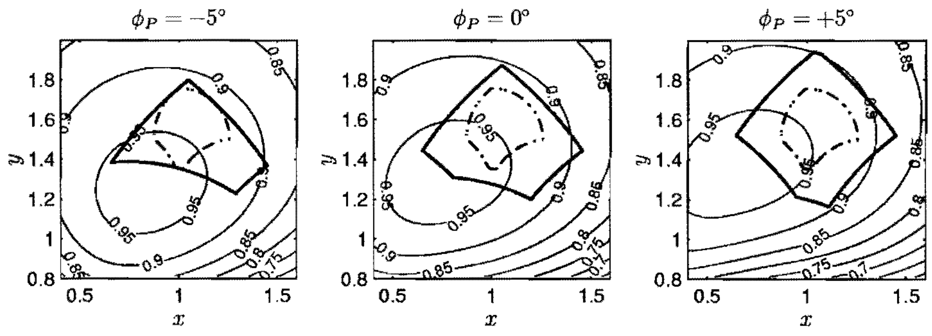

Figure 5.13: MO synthesis manipulator workspace and κ^{-1} contours corresponding to the optimal design for prescribed workspace P3

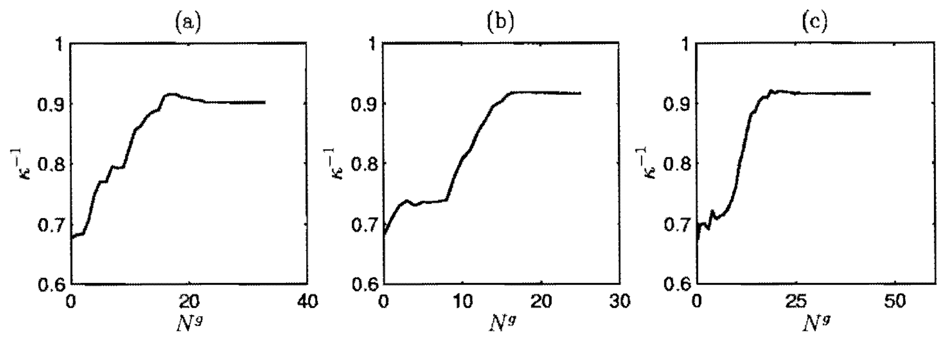

Figure 5.14: MO synthesis convergence histories for (a) P1, (b) P2 and (c) P3

5.8 Optimization for a prescribed dextrous workspace

D synthesis: *Determine a manipulator design that reaches, with optimal conditioning, a prescribed [continuous] dextrous workspace.*

5.8.1 Optimization formulation

Using the methodology for determining dextrous workspaces developed in Section 5.5, the planar parallel manipulator can, as an alternative to MO methodology, be directly synthesized for a prescribed dextrous workspace and optimal conditioning. The form of the optimization problem for achieving this is

$$\begin{aligned}&\max_{\mathbf{d}}\left\{\min_{\mathbf{u}\in W_p^C[\phi^{\min}],W_p^C[\phi^{\max}]}\kappa^{-1}(\mathbf{d},\mathbf{u})\right\}\\&\text{subject to the inequality constraint}\\&g(\mathbf{d})\leq 0\end{aligned}\qquad(5.39)$$

where the inequality constraint function is defined as follows:

$$g(\mathbf{d}) = \begin{cases} \delta W_p^D[\phi^{\min}, \phi^{\max}] & \text{if } \delta W_p^D[\phi^{\min}, \phi^{\max}] > 0 \\ -r^2 & \text{if } \delta W_p^D[\phi^{\min}, \phi^{\max}] = 0 \end{cases} \tag{5.40}$$

The minimum value of the condition number is determined using the same approach as given in Section 5.6. Once again it is assumed that the minimum value of κ^{-1} occurs on the boundary of the prescribed workspace. Furthermore it is expected, based on the results obtained in Section 5.7, that the minimum value will also be associated with an extreme platform orientation, $\phi^{\min}$ or $\phi^{\max}$. Thus in determining the minimum value of κ^{-1} over $W_p^D[\phi^{\min}, \phi^{\min}]$, only the workspace boundaries of $W_p^C[\phi^{\min}]$ and $W_p^C[\phi^{\max}]$, corresponding to the "edges" of the prescribed dextrous workspace, are considered. These assumptions appear to be valid, based on the results obtained here.

5.8.2 Numerical results

The optimization problem embodied in (5.39) has been implemented, once more using the Dynamic-Q algorithm to find optimal designs for prescribed workspace P1-P3. Parameters used for Dynamic-Q were convergence tolerances $\varepsilon_x = 10^{-4}$ and $\varepsilon_f = 10^{-6}$ and a move limit $\rho = 0.05$. Gradients were calculated using *central differences* and a finite difference interval of $\Gamma = 10^{-3}$. A chord length of $d = 0.02$ was used for all calculations. Results obtained are summarized in Figures 5.15 and 5.16 and Table 5.5. The various quantities given in Table 5.5 are the same as those given in Table 5.4.

Comparison of these results with those given for the MO synthesis in Section 5.7.2 indicates that, in general, $f_D^* \leq f_{MO}^*$. This is to be expected, since the specification of a prescribed *dextrous* workspace places a more stringent requirement on the numerical optimization, resulting in lower optimal objective function values in these cases.

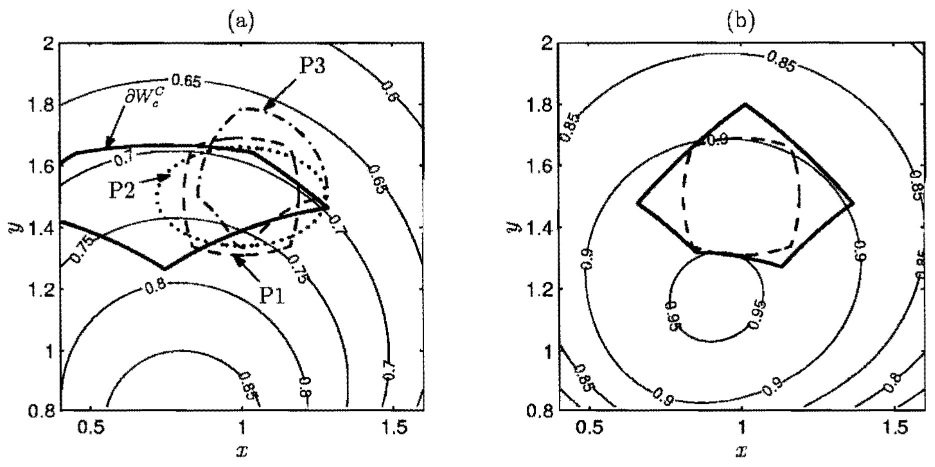

Figure 5.15: D synthesis (a) prescribed workspaces P1-P3, manipulator workspace and corresponding κ^{-1} contours corresponding to the starting design and (b) prescribed workspace P1 and corresponding optimal manipulator workspace and κ^{-1} contours

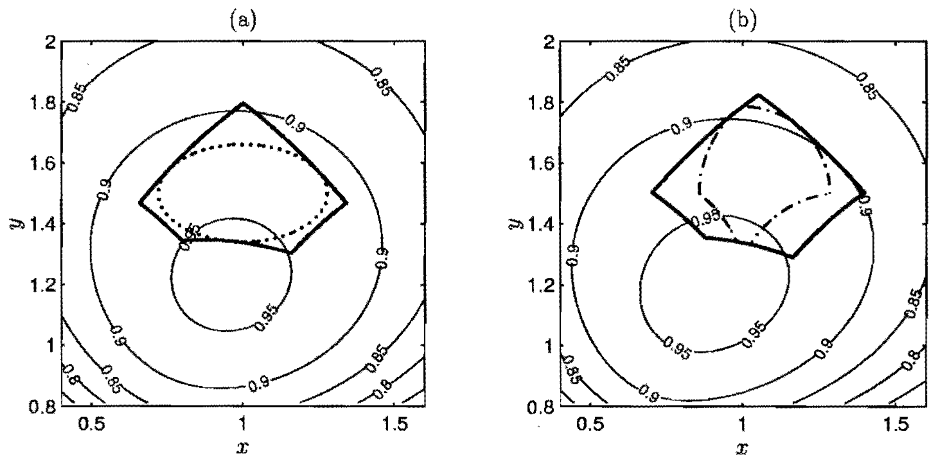

Figure 5.16: D synthesis manipulator workspace and κ^{-1} contours corresponding to the optimal design for prescribed workspaces (a) P2 and (b) P3

	N^g	$f(\mathbf{d}^0)$	f^*	g^*	$\mathbf{d}^*$
P1	33	0.677	0.897	$0.2E-5$	$[-0.9884, 0.2466, 1.339, 1.664, 0.8413]^T$
P2	36	0.681	0.916	$0.6E-7$	$[-1.050, 0.2717, 1.356, 1.716, 0.8500]^T$
P3	52	0.660	0.892	$0.1E-4$	$[-0.9890, 0.2816, 1.362, 1.668, 0.8678]^T$

Table 5.5: D synthesis solutions

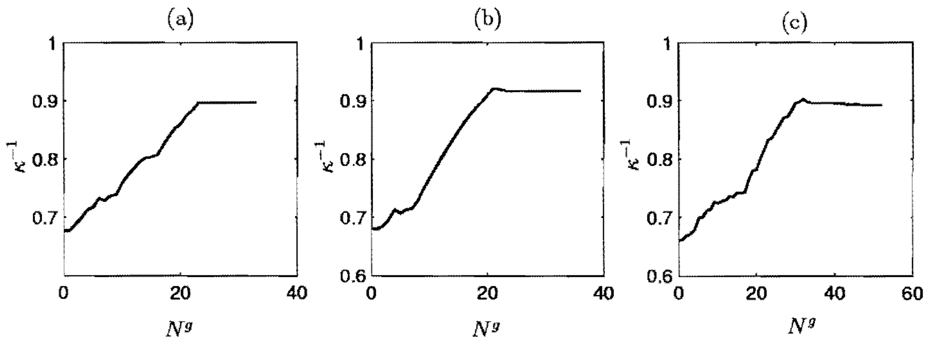

Figure 5.17: D synthesis convergence histories for (a) P1, (b) P2 and (c) P3

5.9 Conclusion

The chord workspace determination methodology, as proposed at the start of this chapter, for the determination of constant orientation and dextrous workspaces, is a reliable and efficient numerical methodology. This has been proven by the fact that optimization of the 3-dof manipulators in the latter part of the chapter inherently requires many manipulator workspaces evaluations, which have been performed robustly by the chord method.

For the 3-dof manipulator studied here the dimensional synthesis results obtained are encouraging, optimum solutions having been obtained with minimal computational effort compared to that which would have been required using evolutionary optimization algorithms. For each of the three synthesis methodologies presented, not only are manipulator dimensions determined so that the prescribed workspace can be reached by the manipulator, but also so that the inverse condition number is as high as possible throughout the prescribed workspace. The proposed methodology produces convincing results, indicating it to be a stable and efficient numerical method for designing planar parallel manipulators. The Dynamic-Q optimization algorithm used in the synthesis methodology exhibits high efficiency in solving the associated optimization problem.

Chapter 6

The planar tendon-driven parallel manipulator

6.1 Introduction

Tendon-driven parallel manipulators represent a relatively recent technology, characterized by the use of cables in place of the linear actuators generally used in parallel manipulators. The use of these manipulators as overhead cranes for materials handling (Dagalakis *et al.* [123], Bostelman *et al.* [124], Verhoeven *et al.* [125]) and worker-access (Bostelman *et al.* [126]) in the heavy, and large-scale manufacturing industries appears to be a promising application of this technology. On a smaller scale another possible application of tendon driven manipulators is in pick-and-place applications. Verhoeven *et al.* [125] also mention applications as fast moving micromanipulators.

Three separate, but inter-related topics are examined in this chapter, and methodologies for addressing these topics are proposed. The first topic addressed is the determination of cable forces for overconstrained tendon-driven manipulators. It is necessary to solve this problem in order to address the second topic, namely the development of a methodology for workspace determination of tendon-driven manipulators. The final topic examined is the dimensional synthesis of tendon-driven manipulators for a large dextrous workspace.

As far as the current state-of-the-art relating to these topics is concerned, Verhoeven and Hiller [127] present a method for determining the cable tension distribution in overconstrained tendon-based parallel manipulators. This method is limited though to the homogenous case, where no external forces are applied to the manipulator end-effector. Lafourcade *et al.* [128] determine cable tensions based on the minimum norm solution. In terms of workspace calculation, Verhoeven and Hiller [129] have proposed a method for determining planar tendon-driven manipulator workspaces considering external forces, but not torques applied to the moving platform. Other authors, for example, Fattah and Agrawal [130] have used a discretization approach to determining and optimizing manipulator workspaces.

In the next section, the tendon-driven parallel manipulators considered in this chapter are presented and the kinematic and static analyses performed. Thereafter methods for determining cable forces are developed and described. Two methodologies for determining workspaces of tendon-driven parallel manipulators are developed and applied to 3- and 4-cable planar tendon-driven parallel manipulators. Finally dimensional synthesis of these manipulators for maximal dextrous workspaces is performed.

6.2 The tendon-driven parallel manipulator

As shown schematically in Figure 6.1, the tendon-driven parallel manipulator (TDPM) considered here consists of a moving platform connected to a fixed frame by means of n cables, with associated displacement vectors $\boldsymbol{\ell}^i = [\ell^i_x, \ell^i_y]^\top$, $i = 1, \ldots, n$. The lengths of the cables, denoted l_i, $i = 1, \ldots, n$

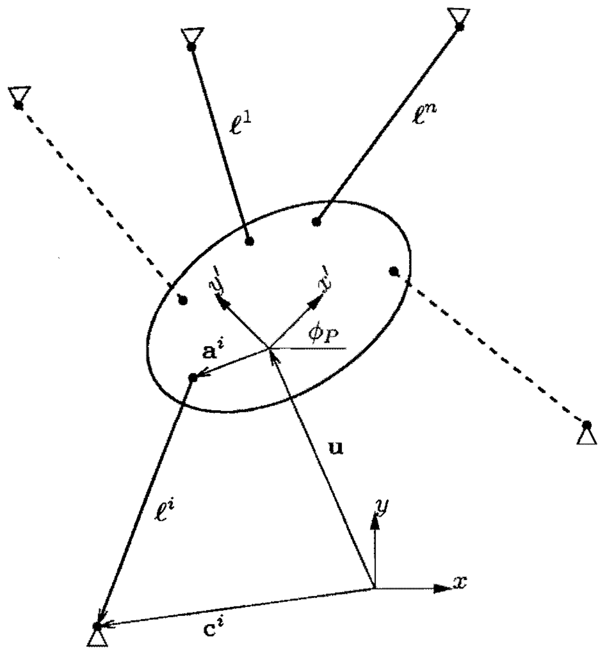

Figure 6.1: Planar tendon-driven manipulator definitions

where $l_i = \ell^i = \|\boldsymbol{\ell}^i\|$, can be varied by winches attached to the fixed frame. A coordinate frame $x' - y'$ is attached to the moving platform. The position of the origin of the platform coordinate frame is $\mathbf{u} = [u_1, u_2]^\top = [x_P, y_P]^\top$, and the platform frame is inclined at an angle ϕ_P to the global $x - y$ coordinate frame. The cables are attached to the fixed frame at $\mathbf{c}^i$, $i = 1, \dots, n$ and to the moving platform at points $\bar{\mathbf{a}}^i, i = 1, \dots, n$, defined relative to the platform coordinate frame $x' - y'$. These vectors may of course be transformed to the global coordinate frame by means of a transformation matrix $\mathbf{T}(\phi_P)$ such that the global attachment vectors $\mathbf{a}^i$ are given by $\mathbf{a}^i = \mathbf{T}(\phi_P)\bar{\mathbf{a}}^i$. The forces in each of the cables are described by vectors $\mathbf{f}^{Ci} = [f_x^{Ci}, f_y^{Ci}]^\top$, $i = 1, 2, \dots, n$. The corresponding magnitudes of the tensions in these cables are denoted $t_i, i = 1, 2, \dots, n$ where $t_i = f^{Ci} = \|\mathbf{f}^{Ci}\|$. External forces and torques acting on the platform are $\mathbf{f}^P = [f_x^P, f_y^P, \tau^P]^\top$.

The specific planar parallel tendon-driven manipulator used to illustrate the methodologies proposed here is shown in Figure 6.2. It is assumed that the

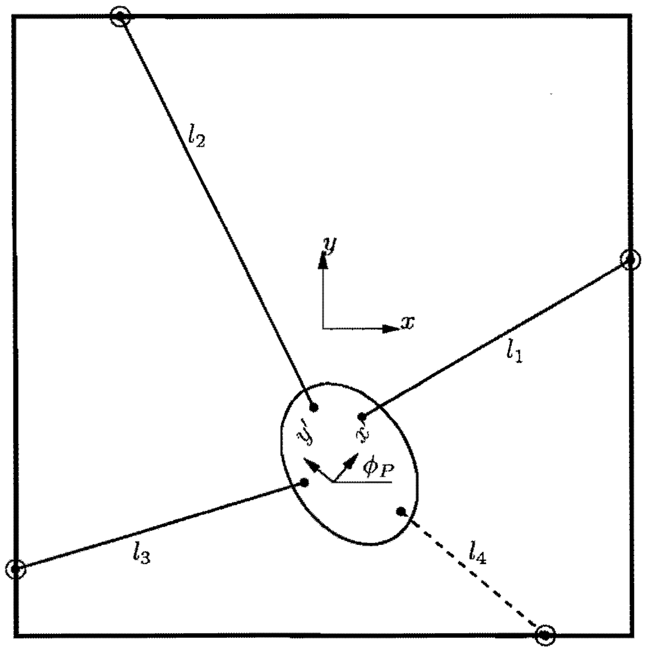

Figure 6.2: The planar tendon-driven manipulator

	3-cable	4-cable
$\bar{a}^1$	$\frac{1}{\sqrt{2}}[0.1, 0.1]^T$	$\frac{1}{\sqrt{2}}[0.1, 0.1]^T$
$\bar{a}^2$	$\frac{1}{\sqrt{2}}[-0.1, 0.1]^T$	$\frac{1}{\sqrt{2}}[-0.1, 0.1]^T$
$\bar{a}^3$	$[0.0, -0.1]^T$	$[-0.1, -0.1]^T$
$\bar{a}^4$	-	$[0.1, -0.1]^T$

Table 6.1: TDPM moving platform cable attachment points

motors are positioned somewhere on a square frame of dimensions 2×2 in arbitrary units. The global origin $x - y$ is positioned at the center of the frame. Two different configurations, a 3-cable and a 4-cable manipulator will be considered. For each of these configurations, the cable attachment points on the moving platform platform are given in Table 6.1, and the exact motor attachment points on the frame in Table 6.2. The workspace of the manipulator is dependent on the load applied to the platform. In this chapter, three different load cases, denoted L1-L3, and given in arbitrary units in Table 6.3, will be considered.

	3-cable	4-cable
$\mathbf{c}^1$	$[1, 1]^\mathrm{T}$	$[1, 0.5]^\mathrm{T}$
$\mathbf{c}^2$	$[-1, 1]^\mathrm{T}$	$[-1, 0.5]^\mathrm{T}$
$\mathbf{c}^3$	$[0, -1]^\mathrm{T}$	$[-1, -0.5]^\mathrm{T}$
$\mathbf{c}^4$	-	$[1, -0.5]^\mathrm{T}$

Table 6.2: Geometrical parameters for the 3 and 4-cable TDPM

	f_x^P	f_y^P	τ^P
L1	0	-10	0
L2	5	-10	0
L3	5	-10	1

Table 6.3: Load conditions L1-L3

6.2.1 Kinematic analysis

Consider the n kinematic constraint equations, expressed in generalized co-ordinates $\mathbf{q} = [\mathbf{u}, \mathbf{v}, w]^\mathrm{T}$, relating the platform position $\mathbf{u} = [x_P, y_P]^\mathrm{T}$, the orientation $w = \phi_P$, and the input cable lengths $\mathbf{v} = [l_1, l_2, \ldots, l_n]^\mathrm{T}$:

$$\mathbf{\Phi}(\mathbf{q}) = \mathbf{\Phi}(\mathbf{u}, \mathbf{v}, w) = \mathbf{0} \tag{6.1}$$

From Figure 6.1, it can be seen that the following relationships hold:

$$\boldsymbol{\ell}^i = \mathbf{c}^i - \mathbf{a}^i(\phi_P) - \mathbf{u},\ i = 1, 2, \ldots, n \tag{6.2}$$

The transformation $\mathbf{a}^i(\phi_P) = \mathbf{T}(\phi_P)\bar{\mathbf{a}}^i$ of $\bar{\mathbf{a}}^i$ from the local to the global coordinate system, by means of the matrix $\mathbf{T}(\phi_P)$, is given by

$$\begin{bmatrix} a_x^i \\ a_y^i \end{bmatrix} = \begin{bmatrix} \cos\phi_P & -\sin\phi_P \\ \sin\phi_P & \cos\phi_P \end{bmatrix} \begin{bmatrix} \bar{a}_x^i \\ \bar{a}_y^i \end{bmatrix} \tag{6.3}$$

for $i = 1, 2, \ldots, n$. The corresponding input cable lengths $l_i = \|\boldsymbol{\ell}^i\|$ may be written as:

$$\begin{aligned} l_i &= \left[(\ell_x^i)^2 + (\ell_y^i)^2\right]^{\frac{1}{2}} \\ &= \left[(c_x^i - a_x^i(\phi_P) - x_P)^2 + (c_y^i - a_y^i(\phi_P) - y_P)^2\right]^{\frac{1}{2}} \end{aligned} \tag{6.4}$$

Writing (6.4) in the standard form (6.1) yields

$$\begin{aligned}\boldsymbol{\Phi}(\mathbf{q}) &= \boldsymbol{\Phi}(\mathbf{u},\mathbf{v},w)\\ &= \begin{bmatrix} v_1 - \left[(c_x^1 - a_x^1(w) - u_1)^2 + (c_y^1 - a_y^1(w) - u_2)^2\right]^{\frac{1}{2}} \\ v_2 - \left[(c_x^2 - a_x^2(w) - u_1)^2 + (c_y^2 - a_y^2(w) - u_2)^2\right]^{\frac{1}{2}} \\ \vdots \\ v_n - \left[(c_x^n - a_x^n(w) - u_1)^2 + (c_y^n - a_y^n(w) - u_2)^2\right]^{\frac{1}{2}} \end{bmatrix} \\ &= \mathbf{0}\end{aligned} \tag{6.5}$$

where a subscript x, y or z denotes the x, y or z component of a vector. Substituting (6.3) into (6.5) and differentiating yields

$$\boldsymbol{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} = \begin{bmatrix} \frac{\ell_x^1}{l_1} & \frac{\ell_y^1}{l_1} & \frac{(\mathbf{a}^1\times\boldsymbol{\ell}^1)_z}{l_1} & 1 & 0 & \cdots & 0 \\ \frac{\ell_x^2}{l_2} & \frac{\ell_y^2}{l_2} & \frac{(\mathbf{a}^2\times\boldsymbol{\ell}^2)_z}{l_2} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\ell_x^n}{l_n} & \frac{\ell_y^n}{l_n} & \frac{(\mathbf{a}^n\times\boldsymbol{\ell}^n)_z}{l_n} & 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \dot{w} \\ \dot{v}_1 \\ \dot{v}_2 \\ \vdots \\ \dot{v}_n \end{bmatrix} = \mathbf{0} \tag{6.6}$$

This can alternatively be written as

$$\begin{bmatrix} \dot{v}_1 \\ \dot{v}_2 \\ \vdots \\ \dot{v}_n \end{bmatrix} = - \begin{bmatrix} \frac{\ell_x^1}{l_1} & \frac{\ell_y^1}{l_1} & \frac{(\mathbf{a}^1\times\boldsymbol{\ell}^1)_z}{l_1} \\ \frac{\ell_x^2}{l_2} & \frac{\ell_y^2}{l_2} & \frac{(\mathbf{a}^2\times\boldsymbol{\ell}^2)_z}{l_2} \\ \vdots & \vdots & \vdots \\ \frac{\ell_x^n}{l_n} & \frac{\ell_y^n}{l_n} & \frac{(\mathbf{a}^n\times\boldsymbol{\ell}^n)_z}{l_n} \end{bmatrix} \begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \dot{w} \end{bmatrix} \text{ or } \dot{\mathbf{v}} = -\mathbf{J}\left[\dot{\mathbf{u}}^{\mathrm{T}}, \dot{w}\right]^{\mathrm{T}} \tag{6.7}$$

where $\mathbf{J}$ is the Jacobian of the kinematic constraint equations with respect to the output $\mathbf{u}$ and intermediate w variables.

6.2.2 Static analysis

Force equilibrium implies that for any configuration, specified by $\mathbf{u}$ and w, the following equations must hold:

$$\sum_{i=1}^{n} \mathbf{f}^{Ci} + [f_x^P, f_y^P]^\top = \mathbf{0} \text{ and } \sum_{i=1}^{n} \mathbf{a}^i \times \mathbf{f}^{Ci} + \tau^P = 0 \qquad (6.8)$$

Noting that the tensions $\mathbf{f}^{Ci}$ act parallel to their corresponding cable vectors $\boldsymbol{\ell}^i$, it follows that

$$\mathbf{f}^{Ci} = \frac{\boldsymbol{\ell}^i}{l_i} t_i \qquad (6.9)$$

Equations (6.8) can thus be rewritten as

$$\sum_{i=1}^{n} \frac{\boldsymbol{\ell}^i}{l_i} t_i + [f_x^P, f_y^P]^\top = \mathbf{0} \text{ and } \sum_{i=1}^{n} \mathbf{a}^i \times \frac{\boldsymbol{\ell}^i}{l_i} t_i + \tau^P = 0 \qquad (6.10)$$

where $l_i = \|\boldsymbol{\ell}^i\|$. Writing (6.10) in matrix form gives

$$\begin{bmatrix} f_x^P \\ f_y^P \\ \tau^P \end{bmatrix} = -\left[\begin{array}{ccc|c} \frac{\ell_x^1}{l_1} & \frac{\ell_x^2}{l_2} & \frac{\ell_x^3}{l_3} & \cdots \quad \frac{\ell_x^n}{l_n} \\ \frac{\ell_y^1}{l_1} & \frac{\ell_y^2}{l_2} & \frac{\ell_y^3}{l_3} & \cdots \quad \frac{\ell_y^n}{l_n} \\ \frac{(\mathbf{a}^1\times\boldsymbol{\ell}^1)_z}{l_1} & \frac{(\mathbf{a}^2\times\boldsymbol{\ell}^2)_z}{l_2} & \frac{(\mathbf{a}^3\times\boldsymbol{\ell}^3)_z}{l_3} & \cdots \quad \frac{(\mathbf{a}^n\times\boldsymbol{\ell}^n)_z}{l_n} \end{array}\right] \left[\begin{array}{c} t_1 \\ t_2 \\ t_3 \\ \hline \vdots \\ t_n \end{array}\right] \qquad (6.11)$$

$$\text{or } \mathbf{f}^P = -\mathbf{S}\mathbf{t}$$

where $\mathbf{S}$ is called the *structure matrix* of the manipulator (Verhoeven *et al.* [125]). It is interesting to note that $\mathbf{S} = \mathbf{J}^\top$, where $\mathbf{J}$ is the Jacobian defined by equation (6.7).

6.3 Calculation of the cable tensions

6.3.1 Minimum norm approach

It is evident for the planar case, as pointed out by Verhoeven and Hiller [127] and Fattah and Agrawal [130], that system (6.11) is overconstrained and thus has many solutions if $n > 3$. For $n = 3$ there are 3 equations and 3 unknowns, and thus if the equations are linearly independent, there will be a unique solution simply given by

$$\mathbf{t} = -\mathbf{S}^{-1}\mathbf{f}^P \tag{6.12}$$

For $n > 3$ there are many solutions, assuming $\mathbf{S}\mathbf{S}^\top$ is invertible. The general solution to (6.11) in this case is of the form

$$\mathbf{t} = \mathbf{t}^{\mathrm{mn}} + \mathbf{t}^{\mathrm{nul}} \tag{6.13}$$

where $\mathbf{t}^{\mathrm{mn}}$ is the minimum norm solution of (6.11) and $\mathbf{t}^{\mathrm{nul}}$ is a vector belonging to the nullspace $\mathcal{N}(\mathbf{S})$ of $\mathbf{S}$. The minimum norm solution $\mathbf{t}^{\mathrm{mn}}$ is determined by means of the Moore-Penrose inverse, defined as $\mathbf{S}^+ = \mathbf{S}^\top(\mathbf{S}\mathbf{S}^\top)^{-1}$, and is given by (Fattah and Agrawal [130])

$$\mathbf{t}^{\mathrm{mn}} = -\mathbf{S}^+\mathbf{f}^P = -\mathbf{S}^\top(\mathbf{S}\mathbf{S}^\top)^{-1}\mathbf{f}^P \tag{6.14}$$

Fattah and Agrawal [130] take $\mathbf{t}^{\mathrm{nul}} = \mathbf{0}$ and thus the solution to (6.11) is simply given by setting $\mathbf{t} = \mathbf{t}^{\mathrm{mn}}$ where $\mathbf{t}^{\mathrm{mn}}$ is given by (6.14). As will be illustrated in the next section this approach may result in some feasible points being excluded from the workspace. The minimum norm solution does however have the advantage that it is not computationally demanding, and may thus have some use when performing rough workspace calculations. Lafourcade *et al.* [128] present a simple iterative method whereby $\mathbf{t}^{\mathrm{nul}}$ is gradually increased until feasible cable tensions are determined.

6.3.2 Constrained ℓ_2-norm approach

A new methodology for determining the cable tensions for any specified $\mathbf{u}$ and w is now proposed. Using the partitioning indicated in system (6.11) to define

$$\mathbf{f}^P = -\left[\mathbf{A}\middle|\mathbf{B}\right]\left[\mathbf{t}^{A\top}\middle|\mathbf{t}^{B\top}\right]^\top \tag{6.15}$$

where $\mathbf{A}$ is 3×3 and $\mathbf{B}$ is $3 \times (n-3)$, system (6.11) can be rewritten as:

$$\mathbf{t}^A(\mathbf{t}^B) = -\mathbf{A}^{-1}\mathbf{f}^P - \mathbf{A}^{-1}\mathbf{B}\mathbf{t}^B \tag{6.16}$$

which gives the values of the dependent tensions $\mathbf{t}^A \in \Re^3$ as a function of the independent tensions $\mathbf{t}^B \in \Re^{n-3}$. For a given position $\mathbf{u} = [x_P, y_P]^\mathrm{T}$ and orientation $w = \phi_P$ of the platform, and for identical prescribed lower and upper bounds, $t^{\min}$ and $t^{\max}$ on the tension magnitudes t_i, $i = 1, 2, \ldots, n$, the cable tensions $\mathbf{t}$ may now be determined by solving the following numerical optimization problem[1]:

$$\begin{aligned} &\min_{\mathbf{t}^B} \|\mathbf{t}\|_2^2 \\ &\text{such that } t^{\min} \leq t_i^A(\mathbf{t}^B) \leq t^{\max},\ i = 1, 2, 3 \\ &\text{and } t^{\min} \leq t_j^B \leq t^{\max},\ j = 1, 2, \ldots, (n-3) \end{aligned} \tag{6.17}$$

where $\mathbf{t} = [t_1, t_2, \ldots, t_n]^\mathrm{T} = [\mathbf{t}^{A\mathrm{T}}, \mathbf{t}^{B\mathrm{T}}]^\mathrm{T} = [t_1^A, t_2^A, t_3^A, t_1^B, \ldots, t_{n-3}^B]^\mathrm{T}$. This optimization problem is solved using the LfopC numerical optimization algorithm of Snyman [103] (see Appendix B). Of course, on removal of the inequality constraints, it can be shown that optimization problem (6.17) reduces to equation (6.14) (see Appendix E). The advantage of the methodology proposed here is that either the minimum, or maximum possible cable tensions, in the case where a maximization is performed instead in (6.17), can be determined. The respective solutions correspond to a tendon driven system with minimal energy consumption, or one with maximal stiffness.

Figure 6.3 illustrates the significant difference between the minimum norm (with $\mathbf{t}^{\mathrm{nul}} = \mathbf{0}$) and constrained ℓ_2-norm approach approaches for determining the cable tensions. The illustrative manipulator design analyzed here has frame attachment points $\mathbf{c}^1 = [1, 0.15]^\mathrm{T}$, $\mathbf{c}^2 = [-1, 0.15]^\mathrm{T}$, $\mathbf{c}^3 = [-1, -0.15]^\mathrm{T}$ and $\mathbf{c}^4 = [1, -0.15]^\mathrm{T}$ and platform attachment points as given in Table 6.1. The constant orientation workspaces for $\phi_P = 0$, $\phi_P = 0.2$ and $\phi_P = 0.4$ are given. Minimum and maximum allowable tensions were 5 and 100 respectively. The workspaces indicated by a solid line were calculated using

[1] $\|\cdot\|_2$ denotes the ℓ_2-norm of its argument.

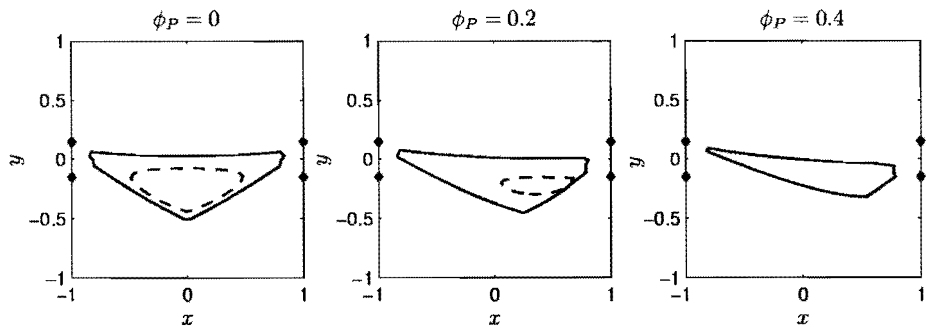

Figure 6.3: Comparison of workspaces obtained with cable forces calculated using the minimum norm (dashed line) and constrained ℓ_2-norm (solid line) approaches.

the constrained ℓ_2-norm approach, and those indicated by a dashed line were calculated using the unconstrained minimum norm approach for determining the cable tensions. Workspaces were determined using the chord method (see Section 6.4.3). It is evident that using the unconstrained minimum norm solution here results in an extreme underestimation of the feasible workspace of the manipulator.

6.3.3 Constrained ℓ_1-norm approach

The constrained ℓ_2-norm approach for determining cable tensions outlined above is limited by the computational effort required by the numerical optimization algorithm. When determining manipulator workspaces, for example, the cable tensions will have to be determined many times. It is thus desirable to have a more efficient method for determining the feasible cable tensions. The constrained ℓ_1-norm approach follows from the constrained ℓ_2-norm approach outlined above, but is posed as a linear programming problem, allowing for a more efficient solution of the cable tensions. For a fixed position and orientation of the moving platform, the constraints on the three dependent cable tensions $\mathbf{t}^A$ are used to calculate the feasible region for the independent cable tensions $\mathbf{t}^B \in \Re^{n-3}$. For each dependent cable, minimum and maximum tension limits dictate that the following inequalities hold:

$$\begin{aligned} t^{\min} - t_i^A(\mathbf{t}^B) \le 0,\ i = 1,2,3 \\ \text{and } t_i^A(\mathbf{t}^B) - t^{\max} \le 0,\ i = 1,2,3 \end{aligned} \tag{6.18}$$

Defining $\mathbf{M} = -\mathbf{A}^{-1}\mathbf{f}^P$ and $\mathbf{N} = -\mathbf{A}^{-1}\mathbf{B}$, substituting (6.16) into (6.18), and including the inequalities limiting the tensions of the independent cables $\mathbf{t}^B$ the feasible region in the independent cable tension space $\mathbf{t}^B \in \Re^{n-3}$ is bounded by the inequality constraints of the following optimization problem[2]:

$$\begin{gathered} \min_{\mathbf{t}_B} \|\mathbf{t}\|_1 = \min_{\mathbf{t}_B} \left(\sum_{p=1}^{n} t_p^A(\mathbf{t}^B) + \sum_{q=1}^{n-3} t_q^B \right) \\ \text{such that} \\ \sum_{j=1}^{n-3} N_{ij} t_j^B \ge t^{\min} - M_i,\ i = 1,2,3; \\ \sum_{j=1}^{n-3} N_{ij} t_j^B \le t^{\max} - M_i,\ i = 1,2,3 \\ \text{and } t^{\min} \le t_k^B \le t^{\max},\ k = 1,2,\dots,(n-3) \end{gathered} \tag{6.19}$$

The minimum or maximum (where a maximization is performed instead) allowable cable tensions may be determined by solving optimization problem (6.19). For $n \ge 5$ this can be done efficiently using linear programming methods. For the case where $n = 4$, inequalities (6.19) reduce to

$$\begin{gathered} N_i t^B \ge (t^{\min} - M_i),\ i = 1,2,3 \\ N_i t^B \le (t^{\max} - M_i),\ i = 1,2,3 \\ 0 \le t^{\min} \le t^B \le t^{\max} \end{gathered} \tag{6.20}$$

The existence of a feasible solution, and the minimum $t^{B\min}$ and maximum $t^{B\max}$ allowable independent cable tensions may in this case be easily and efficiently determined analytically by examining the extreme values of t^B

[2] $\|\cdot\|_1$ denotes the ℓ_1-norm of its argument.

defined by (6.20). The exact algorithm achieving this is stated in Algorithm 6.1.

Algorithm 6.1 Tension limit algorithm

```
for $i = 1 : 3$
    if $N_i > 0$
        $t_i^{\text{low}} = (t^{\text{min}} - M_i)/N_i$
        $t_i^{\text{high}} = (t^{\text{max}} - M_i)/N_i$
    else
        $t_i^{\text{low}} = (t^{\text{max}} - M_i)/N_i$
        $t_i^{\text{high}} = (t^{\text{min}} - M_i)/N_i$
    end if
end for
$t_4^{\text{low}} = t^{\text{min}}$
$t_4^{\text{high}} = t^{\text{max}}$
$t^{B\,\text{min}} = \max_i(t_i^{\text{low}}),\ i = 1, \ldots, 4$
$t^{B\,\text{max}} = \min_i(t_i^{\text{high}}),\ i = 1, \ldots, 4$
```

If $t^{B\,\text{min}} > t^{B\,\text{max}}$, then no feasible solution exists for the cable tensions. Of course, once t^B is known, the dependent cable tensions $\mathbf{t}^A$ may be calculated using (6.16). Note that since the minimum ℓ_1-norm solution may correspond to *either* $t^{B\,\text{min}}$ or $t^{B\,\text{max}}$, it is necessary to evaluate and compare both possibilities in order to determine the correct minimum norm. The same is true when determining the maximum ℓ_1-norm. In terms of control of such a manipulator, the cable tensions $\mathbf{t}$ corresponding to either $t^{B\,\text{min}}$ or $t^{B\,\text{max}}$, or indeed any intermediate values, may be used since these are all feasible solutions for the cable tensions.

6.4 Constant orientation workspace determination

6.4.1 Workspace definition

Unlike regular parallel manipulators, where the workspace is dependent on the input joint limits and the geometrical realization of the manipulator, the workspace for tendon-driven manipulators is primarily determined by the allowable forces in the cables (Verhoeven *et al.* [125]). In particular, the tension in each of the cables should lie between a pretension and maximum tension:

$$t^{\min} \leq t_i \leq t^{\max}, \; i = 1, 2, \ldots, n \tag{6.21}$$

Here it is also required that each cable has a minimum allowable length $l^{\min}$:

$$l_i \geq l^{\min}, \; i = 1, 2, \ldots, n \tag{6.22}$$

The constant orientation workspace $W^C[w^{\text{fix}}]$ (as defined in Section 1.3.1) of the tendon-driven parallel manipulator for a fixed platform orientation $w^{\text{fix}} = \phi_P^{\text{fix}}$ can now be defined as

$$\begin{aligned} W^C[w^{\text{fix}}] = \big\{\mathbf{u} \in \Re^2 : \mathbf{\Phi}(\mathbf{u}, \mathbf{v}, w^{\text{fix}}) = \mathbf{0}, \\ t^{\min} \leq t_i \leq t^{\max} \text{ and } l_i \geq l^{\min}, \; i = 1, 2, \ldots, n\big\} \end{aligned} \tag{6.23}$$

Where the cable tensions $\mathbf{t}$ are calculated using either the minimum norm, constrained ℓ_2-norm, or constrained ℓ_1-norm approach proposed in the previous section, and the cable lengths $\mathbf{v}$ are calculated using (6.4). Other conditions such as platform stiffness and proximity to singularities can also be easily included in the workspace definition by the addition of further inequality constraints. In this chapter, for the the 3-cable manipulator, a lower limit is placed on the determinant of the kinematic Jacobian, to ensure that the manipulator does not approach a singular position:

$$\det(\mathbf{J}) \geq D^{\min} \tag{6.24}$$

Limit	Value
t^{min}	5
t^{max}	100
l^{min}	0.1
D^{min}	10^{-3}

Table 6.4: Numerical values of limits used in calculating TDPM workspaces

The values of the various maximum and minimum limits required in inequalities (6.21), (6.22) and (6.24), as used for all numerical examples in this chapter, are given in Table 6.4.

6.4.2 Discretization method

As mentioned in Section 1.3.2, one method commonly used by researchers for determining parallel manipulator workspaces is the discretization method. For a given fixed orientation ϕ^{fix} of the moving platform, this method simply involves discretizing the output space $\mathbf{u} \in \Re^2$ of the planar TDPM at a given resolution, and then testing each of the resulting mesh points $\mathbf{u}^{ij}$ for compliance with the inequalities (6.21) and (6.22), as well as (6.24) if applicable. If none of these inequalities are violated then the point lies within the manipulator workspace. If any of the constraints are violated, then the point does not lie within the workspace. The basic discretization method used is stated in Algorithm 6.2.

This discretization method has been applied to workspace determination of various manipulator designs, using the constrained ℓ_2-norm method, embodied in (6.17) for determining the cable tensions in the 4-cable case.

Algorithm 6.2 Discretization algorithm

1. Select the required workspace resolution, m, specifying the number of points to be inserted between the maximum and minimum possible limits of the workspace determined in the next step.

2. Determine extreme limits of the region to be discretized by determining $x^{\min} = \min(c_x^i)$, $x^{\max} = \max(c_x^i)$, $y^{\min} = \min(c_y^i)$, and $y^{\max} = \max(c_y^i)$ for $i = 1, 2, \ldots, n$ in each case.

3. Discretize the output space of the manipulator by determining $(m+1)^2$ points $\mathbf{u}^{ij} = [x^{\min} + \frac{i}{m}(x^{\max} - x^{\min}), y^{\min} + \frac{j}{m}(y^{\max} - y^{\min})]^\top$ for $i = 0, 1, 2, \ldots, m$ and $j = 0, 1, 2, \ldots, m$.

4. Test each $\mathbf{u}^{ij}$ for compliance with inequalities (6.21) and (6.22) (and (6.24) for the 3-cable case). If all inequalities are satisfied, record $\mathbf{u}^{ij}$ as a valid point.

3-cable manipulator

Constant orientation workspaces for the 3-cable manipulator described in Section 6.2, and for values of $\bar{\mathbf{a}}^i$ and $\mathbf{c}^i$ as defined in Tables 6.1 and 6.2 respectively, were calculated using the discretization method. Figure 6.4 shows the constant orientation workspaces obtained for[3] $\phi_P = 0$, $\phi_P = 0.05$ and $\phi_P = 0.1$ in the columns, and for load cases L1, L2 and L3 (given in Table 6.3) in the rows. For these cases, using $m = 50$, the average time for workspace calculation was 0.03s, using a FORTRAN code on a 1.6 GHz Pentium 4 computer. Of course, as m is increased, the time required for computation increases proportionally to m^2.

It is interesting to note, for this manipulator design, that the manipulator workspace is highly dependent on the both the load on the moving platform,

[3]All angles in this chapter are expressed in radians.

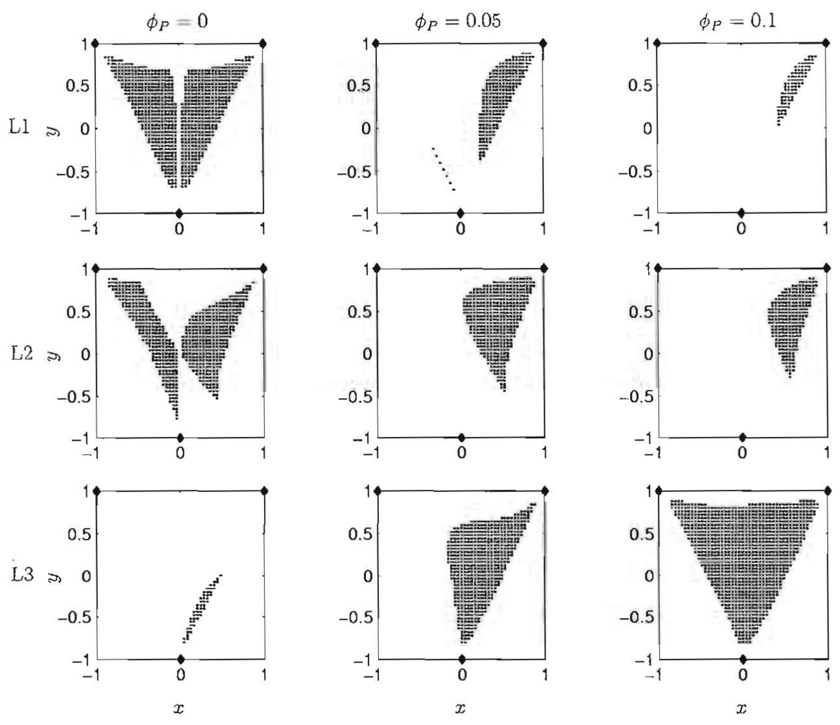

Figure 6.4: Workspaces of the 3-cable TDPM determined using the discretization method

as well as the orientation of the platform. Note that even small changes in the platform orientation result in extreme changes in the workspace. Also of interest is the presence of a singularity at $x = 0$ for the cases where $\phi_P = 0$ which effectively divides the workspace into two usable regions. The workspace boundaries in the vicinity of this singularity are dictated by the limit $D^{\min}$ on the determinant of the Jacobian, implemented by means of inequality (6.24).

4-cable manipulator

Figure 6.5 shows some constant orientation workspaces of the 4-cable manipulator presented in Section 6.2, and defined in Tables 6.1 and 6.2. These

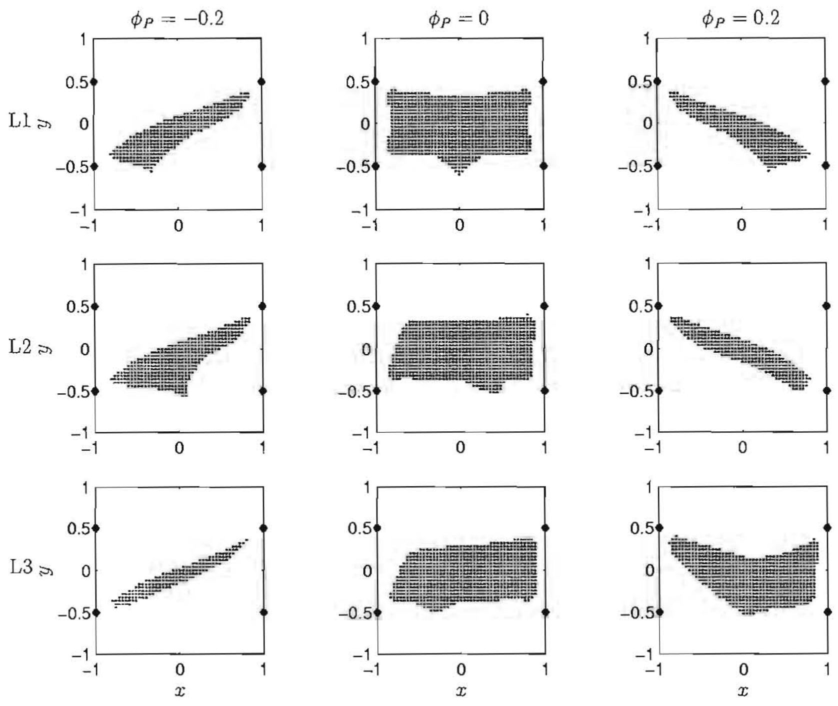

Figure 6.5: Workspaces of the 4-cable TDPM determined using the discretization method

workspaces were calculated using the discretization algorithm with $m = 50$, and the constrained ℓ_2-norm method embodied in (6.17) for determining the cable tensions. The average time for workspace computation was 29.3s, using FORTRAN on a 1.6 GHz Pentium 4 computer. It is immediately evident that the necessity of using a more complicated approach for determining the cable tensions for the 4-cable manipulator dramatically increases the time required to compute the manipulator workspace. Times required to compute workspaces using the constrained ℓ_1-norm approach for determining cable tensions are comparable to the 3-cable case.

Of interest is the fact that for the particular design presented here, the manipulator has a greater workspace than for the 3-cable manipulator. It also appears that the size of the workspace is less sensitive to platform orientation and applied loads than its 3-cable counterpart.

6.4.3 Chord method

The tendon-driven parallel manipulator constant orientation workspaces can also be determined using a modified version of the chord method proposed in Section 5.4. As before the method consists of finding an initial single point on the workspace boundary, and then using an optimization-based search methodology to determine subsequent points along the workspace boundary at constant chord lengths d.

In order to determine a point on the workspace boundary, a single feasible point $\mathbf{u}^0$ somewhere within the workspace boundary must first be determined. This may be accomplished in one of two ways. The first possibility is to run the discretization method, at a coarse resolution to obtain a rough estimation of the workspace. The internal point for the method can then be chosen manually, and the chord methodology used to map the boundary accurately. Alternatively a suitable internal point may be found by solving the following unconstrained optimization problem:

$$\min_{\mathbf{u}} \sum_{i=1}^{n} (t_i - t^{\text{mean}})^2 \tag{6.25}$$

where $t^{\text{mean}} = (t^{\text{max}} - t^{\text{min}})/2$. In the implementation this numerical optimization problem is solved using the efficient Dynamic-Q method developed in Chapter 3.

Once a suitable internal point $\mathbf{u}^0$ has been found, the initial point $\mathbf{b}^1$ on the workspace boundary is found by solution of the following optimization problem which replaces (5.17).

$$\begin{gathered}\max_{r} r^2 \\ \text{such that } t^{\text{min}} \leq t_i(\mathbf{u}(r)) \leq t^{\text{max}} \\ \text{and } l_i(\mathbf{u}(r)) \geq l^{\text{min}}\end{gathered} \tag{6.26}$$

where $\mathbf{u}(r) = \mathbf{u}^0 + r\mathbf{s}^1$ as before and, of course, r is the distance from the internal point $\mathbf{u}^0$. For the 3-cable manipulator, inequality constraint (6.24) is also included in the optimization problem. The cable tensions t_i, $i = 1, 2, \ldots, n$ are calculated using either the constrained ℓ_2-norm or constrained ℓ_1-norm approach with $\mathbf{u} = \mathbf{u}(r)$, and cable lengths l_i, $i = 1, 2, \ldots, n$ are given by (6.4).

The chord methodology, with specified chord length d, is embodied in the following optimization problem for the tendon-driven manipulator:

$$\begin{gathered} \min_{\omega} \omega^2 \\ \text{such that } t^{\min} \leq t_i(\mathbf{u}) \leq t^{\max} \\ \text{and } l_i(\mathbf{u}) \geq l^{\min} \end{gathered} \tag{6.27}$$

where, $\mathbf{u} = \mathbf{u}(\omega)$ is given by (5.18). The solution of (6.27) yields the next point $\mathbf{b}^{i+1}$ along the workspace boundary and a vector $\mathbf{s}^{2i} = [s_x^{2i}, s_y^{2i}]^{\mathsf{T}}$, of magnitude d, pointing from $\mathbf{b}^i$ to $\mathbf{b}^{i+1}$. Once again for the 3-cable manipulator, an additional inequality constraint corresponding to condition (6.24) is also included in the optimization problem. The values of the cable tensions, for any $\mathbf{u}$ and prescribed ϕ^{fix}, are once again calculated using either the constrained ℓ_2-norm or constrained ℓ_1-norm approaches, and the cable lengths using (6.4). The chord method otherwise remains the same as presented in Appendix C.

Choice of optimization method

In previous implementations of the chord method, the numerical optimization algorithms used for solving problems (6.26) and (6.27) were the LfopC method (Snyman [103]), or the more efficient Dynamic-Q method (Chapter 3). These methods performed well in these cases, since the constraints typically appearing in the workspace boundary definition were continuous for the types of parallel manipulators studied. For tendon-driven manipulators, this is not always the case. In particular the constraints corresponding to the cable tension limits (6.21) may be discontinuous on the workspace boundary. Consider Figure 6.6(a) which shows the $\phi_P = 0$ constant orientation

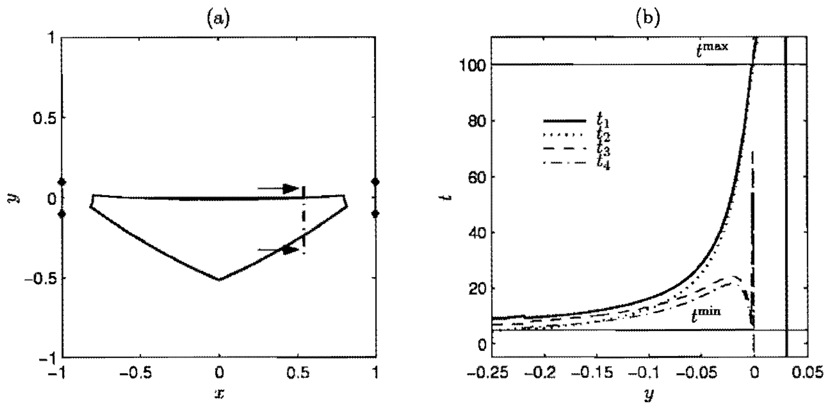

Figure 6.6: (a) Workspace of a 4-cable manipulator with the section indicated by means of the chained line and (b) cable tensions t_i, $i = 1, 2, 3, 4$ along the section

workspace under load case L1 for a 4-cable manipulator with $\mathbf{c}^1 = [1, 0.1]^\mathrm{T}$, $\mathbf{c}^2 = [-1, 0.1]^\mathrm{T}$, $\mathbf{c}^3 = [-1, -0.1]^\mathrm{T}$ and $\mathbf{c}^4 = [1, -0.1]^\mathrm{T}$, and platform attachment points $\bar{\mathbf{a}}^i$ as given in Table 6.1. The chained line indicates the position of a section taken though the workspace (at $x = 0.54$), along which the cable tensions have been calculated (Figure 6.6(b)). It is evident from Figure 6.6(b) that cable tensions t_3 and t_4 are discontinuous at $y = 0$, a point which corresponds to the workspace boundary. Clearly, the result of these discontinuities is that it is not possible to use a gradient-based algorithm for solving problems (6.26) and (6.27) at every point on the workspace boundary.

This problem is addressed by using a bisection method to solve problem (6.27). It is possible to use a bisection method here, since there is only one variable, namely ω, to solve for in optimization problem. The exact bisection algorithm used is given in Algorithm 6.3.

It is thus now possible to solve optimization problem (6.27) using the Dynamic-Q optimization method, reverting to the bisection algorithm when the optimization breaks down. On implementation, however, it was found that the bisection algorithm is more economical in solving optimization (6.27) than

Algorithm 6.3 Bisection algorithm

1. Given two successive boundary points $\mathbf{b}^{i-1}$ and $\mathbf{b}^{i}$, with $\mathbf{s}^{2(i-1)} = \mathbf{b}^{i} - \mathbf{b}^{i-1}$, calculate $\mathbf{s}^{1i}$ using equation (C.13). Set $\boldsymbol{\beta} = -\mathbf{s}^{2(i-1)}$, $\mathbf{u}^{\beta} = \mathbf{b}^{i-1}$, $\boldsymbol{\alpha} = d\mathbf{s}^{1i}$ and $\mathbf{u}^{\alpha} = \mathbf{b}^{i} + \boldsymbol{\alpha}$. Choose termination parameter $\varepsilon^{u}(= 10^{-8})$.
2. Set $\boldsymbol{\gamma} = d(\boldsymbol{\alpha} + \boldsymbol{\beta})/(\|\boldsymbol{\alpha} + \boldsymbol{\beta}\|)$. If the sign of the z-component of $\boldsymbol{\alpha} \times \boldsymbol{\beta}$ is negative, then $\boldsymbol{\gamma} = -\boldsymbol{\gamma}$.
3. Determine whether $\mathbf{u}^{\gamma} = \mathbf{b}^{i} + \boldsymbol{\gamma}$ lies within the workspace by evaluating inequalities (6.21) and (6.22) at this point.
4. If $\mathbf{u}^{\gamma}$ is feasible, set $\boldsymbol{\beta} = \boldsymbol{\gamma}$ and $\mathbf{u}^{\beta} = \mathbf{u}^{\gamma}$, else set $\boldsymbol{\alpha} = \boldsymbol{\gamma}$ and $\mathbf{u}^{\alpha} = \mathbf{u}^{\gamma}$.
5. If $\mathbf{u}^{\gamma}$ is feasible, and termination condition $\left\|\mathbf{u}^{\beta} - \mathbf{u}^{\alpha}\right\| \leq \varepsilon^{u}$ is satisfied, set $\mathbf{b}^{i+1} = \mathbf{u}^{\gamma}$, $\mathbf{s}^{2i} = \boldsymbol{\gamma}$ and stop, else go to Step 2.

Dynamic-Q, and is thus used exclusively here in mapping the workspace boundary.

Determining the bifurcation points

As before it is necessary to implement a special procedure for determining bifurcation points separately as they are encountered along the workspace boundary. The reason for this is illustrated in Figure 6.7(a). The dashed line represents the actual workspace boundary, and the solid line the approximation to the workspace obtained using the chord method. Since the chord method maps the workspace boundary at discrete chord lengths d, it is evident that bifurcation points will not be accurately determined, and the approximation to the workspace will be degraded. Thus determining the locations of the bifurcation points is important in order to obtain a more accurate representation of the manipulator workspace.

3-cable manipulator

Figure 6.8 gives various constant orientation workspaces for the 3-cable manipulator computed by the chord method. The workspaces calculated are chosen to correspond to those given in Section 6.4.2, to allow for comparison with the workspaces computed using the discretization method and depicted in Figure 6.4. In calculating these workspaces a chord length of $d = 0.05$ was used, except for the workspaces corresponding to $\phi_P = 0$ with load cases L2 and L3, where a smaller chord length of $d = 0.03$ was used in order to capture all the significant features of the workspace boundary. Similarly in calculating the small left hand part of the workspace for L1 and $\phi_P = 0.05$ a chord length of $d = 0.02$ was used. The average computational time for these workspaces is approximately 0.8s.

It is evident that the workspace representations obtained by means of the chord method are much more accurate and efficient in terms of information stored than those obtained by the discretization method. On the other hand the discretization method does present an extremely robust method for workspace determination, although limited in terms of accuracy and computational efficiency.

4-cable manipulator

The workspaces for the 4-cable manipulator, determined using the chord method with $d = 0.05$ are given in Figure 6.9. A reduced chord length of $d = 0.02$ was used to determine the workspace corresponding to load case L3 and $\phi_P = -0.2$. Average computational time per workspace was 30.12s using the constrained ℓ_2-norm approach for determining cable tensions. It is evident on comparison with the results using the discretization method given in Section 6.4.2 that, for comparable computational effort, the chord method yields a much more accurate representation of the workspace. Once

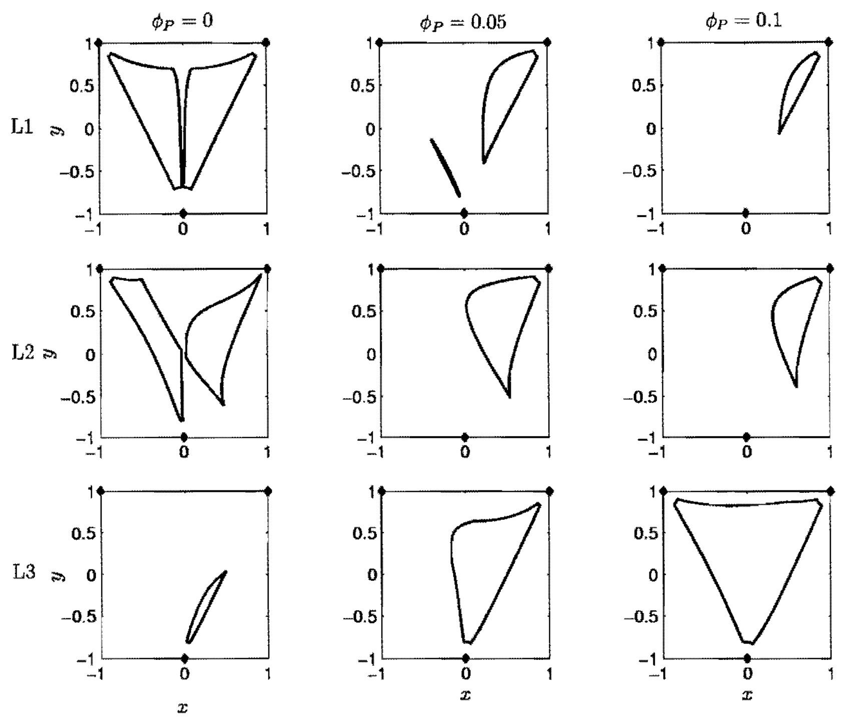

Figure 6.8: Workspaces of the 3-cable TDPM determined using the chord method

again use of the constrained ℓ_1-norm approach for determining the cable tensions results in a much more economical method, with computational times comparable to those of the 3-cable TDPM.

6.5 Dextrous workspace determination

In agreement with the definition given in Section 1.3.1 the dextrous workspace $W^D[\phi^{\min}, \phi^{\max}]$ of the planar tendon-driven manipulator is defined as:

$$\begin{aligned} W^D[\phi^{\min}, \phi^{\max}] = \big\{ &\mathbf{u} \in \Re^2 : \mathbf{\Phi}(\mathbf{u}, \mathbf{v}, w) = \mathbf{0}; \\ &t^{\min} \leq t_i \leq t^{\max},\ i = 1, 2, \ldots, n;\ \text{and} \\ &l_i \geq l^{\min}\ i = 1, 2, \ldots, n \text{ for all } w \in [\phi^{\min}, \phi^{\max}] \big\} \end{aligned} \tag{6.28}$$

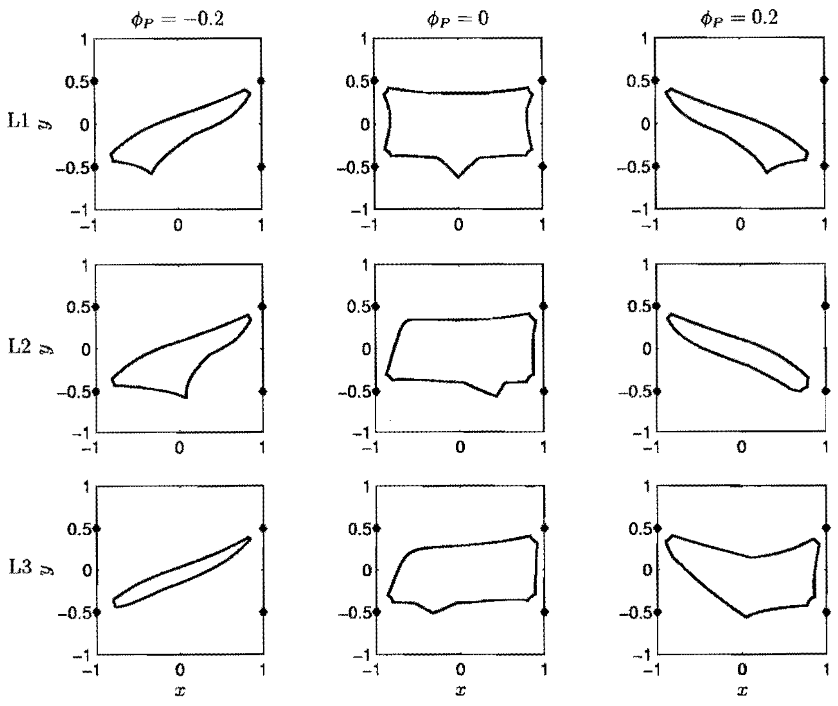

Figure 6.9: Workspaces of the 4-cable TDPM determined using the chord method

The dextrous workspace can be thought of as the intersection of all constant orientation workspaces in the range $[\phi^{\min}, \phi^{\max}]$. Du Plessis and Snyman [18] suggest a numerical approach for determining parallel manipulator dextrous workspaces similar to that used in Section 5.7. Firstly constant orientation workspaces are determined for a finite number m^{sl} of regularly spaced ϕ_P values in the range $[\phi^{\min}, \phi^{\max}]$. The intersection of these constant orientation workspaces then yields the dextrous workspace of the manipulator. In many cases the dextrous workspace may be efficiently and accurately described by simply computing the intersection of the two extreme constant orientation workspaces corresponding to $\phi^{\min}$ and $\phi^{\max}$. Du Plessis and Snyman point out however that it is necessary in this case to also check the validity of this efficient approach by ensuring that the constant orientation workspace at the intermediate central value of $(\phi^{\min} + \phi^{\max})/2$ fully contains the dextrous workspace computed using the extreme values. In the case where the above-mentioned approach is invalid, the dextrous workspace may be progressively more accurately approximated by increasing m^{sl}, the number of regularly spaced values of ϕ_P between $\phi^{\min}$ and $\phi^{\max}$. The discretization algorithm outlined in the previous section can be easily modified to compute the dextrous workspace, by testing for compliance in Step 4 for a *range* of values of ϕ_P, instead of just one fixed value, as is the case for the constant orientation workspace. This approach is used here instead of that developed in Section 5.5, due to its ease of implementation and slightly higher efficiency.

6.6 Dimensional synthesis for maximal dextrous workspace

6.6.1 Optimization formulation

It is evident from the above results presented in Section 6.4 that the manipulator workspace is highly dependent on the manipulator design, load on the end-effector, and end effector orientation. With this in mind, the problem addressed here is to design the manipulator so that it yields the greatest dextrous workspace for a given load on the platform.

It is assumed that the cable frame attachment points $\mathbf{c}^i$ can be arbitrarily positioned anywhere along the fixed square frame. The angle between the global x-axis and cable attachment point $\mathbf{c}^i$ is denoted β^i. The design vector for the optimization problem is thus $\mathbf{d} = [\beta^1, \beta^2, \ldots, \beta^n]^T$. The only constraints imposed on the optimization are that a lower limit $\beta^{\min}$ is placed on the angular separations of the frame cable attachment points. The workspace maximization problem is simply

$$\begin{gathered}\max_{\mathbf{d}} W^D[\phi^{\min},\phi^{\max}]\\ \text{such that } |\beta^{i+1}-\beta^i| \geq \beta^{\min},\ i=1,2,\ldots,n-1 \\ \text{and } |\beta^n-\beta^1| \geq \beta^{\min}\end{gathered} \tag{6.29}$$

where the necessary adjustments are made to the angular measurements to ensure that the angle determined between consecutive cable attachment points is a minimum.

6.6.2 Numerical results

Optimization formulation (6.29) is used in this section to determine optimum manipulator designs with respect to maximal dextrous workspace for the 3 and 4-cable manipulators. For each manipulator type, optimal manipulator configurations were determined for the three different load cases L1-L3 as given in Table 6.3. It is assumed that these loads are applied at the origin of the moving $x'-y'$ frame. Constrained minimum ℓ_1-norm solutions for cable tensions were used and cable tensions limits were $t^{\min}=5$ and $t^{\max}=100$. The optimization problem (6.29) was solved using the Dynamic-Q optimization algorithm (see Chapter 3). As stated previously, this optimization algorithm is suitable for problems where some numerical noise is present in the optimization problem. This is indeed the case here, since the discretization method is used for determining the manipulator dextrous workspaces. Parameters used for the Dynamic-Q method are a move limit of $\rho=0.1$, a finite difference interval of $\Gamma=0.05$ for determining the gradients for the 3-cable manipulator, and $\Gamma=0.1$ for the 4-cable manipulator. Central finite differences were used in calculating these gradients. For the 3-cable manipulator the $[-0.1,0.1]$ dextrous workspace was computed using a resolution m of 100 points, and m^{sl} of 3. For the 4-cable manipulator, which is capable of reaching a larger workspace, the $[-0.2,0.2]$ dextrous workspace was computed using a resolution m of 50 points, and m^{sl} of 21. Note that the number of intermediate values of ϕ_P used for the 3- and 4-cable manipulator differs significantly. This is because just three values of ϕ_P can be used to calculate the 3-cable manipulator dextrous workspaces accurately and efficiently. For the 4-cable manipulator, however, m^{sl} must be increased in order to obtain an accurate description of the dextrous workspace. For both the three and four-cable manipulator and each load case, five feasible random starting designs were chosen for the numerical optimisation embodied in equation (6.29).

3-cable manipulator

Results obtained for the three load cases for 3-cable manipulator are given in Tables 6.5 to 6.7. Each Table gives the number of gradient evaluations N^g of the Dynamic-Q algorithm required to find the solution, the randomly chosen starting design $\mathbf{d}^0$ and area A_d^0 of the associated dextrous workspace, the optimized design $\mathbf{d}^*$ and area A_d^* of the optimized dextrous workspace. Figures 6.10 to 6.12 show results of the dimensional synthesis for representative optimization runs R3, R3 and R1 respectively for load cases L1-L3. In each case Figure (a) shows the starting design and associated workspace, and Figure (b) the optimal design and workspace. Interestingly, for the 3-cable case the dextrous workspace increases in size if a torque is applied to the moving platform. Also of interest when examining the results is the presence of local maxima in the design space. As an example of this see run 4 for L2 (Table 6.6) the solution of which corresponds to a local maxima. The solution obtained by run 4 of L1 (Table 6.5) is the mirror image (about $x = 0$) of the other solutions. The optimization problem thus has two global minima.

4-cable manipulator

Optimization results for the 4-cable manipulator are reported in Tables 6.8 to 6.10. Figures 6.13 to 6.15 once more show representative results from

Run	N^g	$\mathbf{d}^0$	A_d^0	$\mathbf{d}^*$	A_d^*
1	52	$[0.6607, 2.748, 4.367]^\top$	0.0660	$[0.7676, 3.430, 3.605]^\top$	0.3152
2	34	$[0.6725, 2.880, 4.633]^\top$	0.0880	$[0.7631, 3.519, 3.693]^\top$	0.3168
3	39	$[0.2814, 3.978, 4.278]^\top$	0.1996	$[0.7645, 3.477, 3.651]^\top$	0.3172
4	50	$[0.735, 2.418, 5.159]^\top$	0.0024	$[-0.3146, 2.379, 5.794]^\top$	0.3168

Table 6.5: 3-cable TDPM optimized designs for L1

Run	N^g	$\mathbf{d}^0$	A_d^0	$\mathbf{d}^*$	A_d^*
1	17	$[0.9758, 4.056, 4.225]^\top$	0.0216	$[0.6179, 3.703, 3.878]^\top$	0.5136
2	21	$[0.5014, 2.808, 3.616]^\top$	0.0524	$[0.7728, 3.348, 3.523]^\top$	0.5120
3	41	$[0.6861, 2.431, 4.461]^\top$	0.0812	$[0.7667, 3.454, 3.628]^\top$	0.5228
4	55	$[1.864, 2.621, 4.251]^\top$	0.0100	$[1.683, 1.858, 4.389]^\top$	0.2504

Table 6.6: 3-cable TDPM optimized designs for L2

Run	N^g	$\mathbf{d}^0$	A_d^0	$\mathbf{d}^*$	A_d^*
1	83	$[0.7983, 1.281, 3.373]^\top$	0.0500	$[0.6543, 2.357, 4.252]^\top$	1.024
3	5	$[0.6097, 2.319, 4.197]^\top$	0.8364	$[0.6550, 2.371, 4.215]^\top$	1.019
3	48	$[0.3040, 1.220, 5.040]^\top$	0.0028	$[0.6358, 2.366, 4.263]^\top$	1.019
4	45	$[0.5922, 1.718, 4.865]^\top$	0.1736	$[0.6500, 2.369, 4.233]^\top$	1.021

Table 6.7: 3-cable TDPM optimized designs for L3

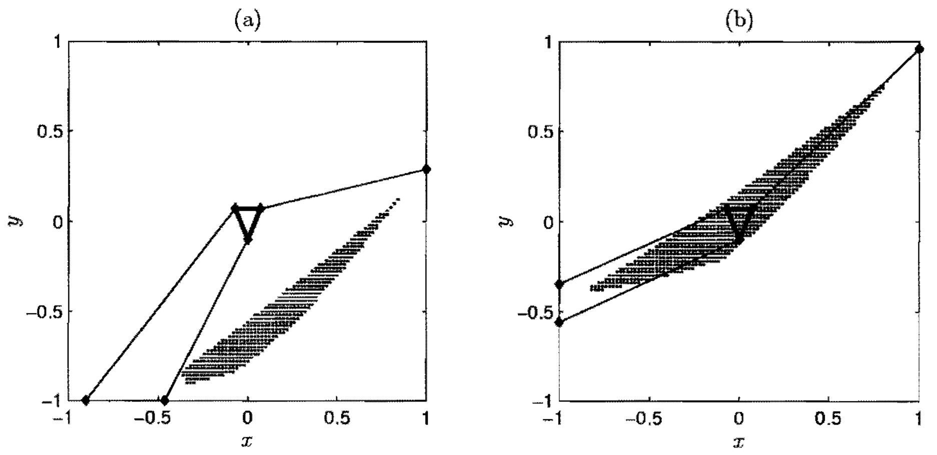

Figure 6.10: 3-cable TDPM (a) starting and (b) optimized design for L1 (R3)

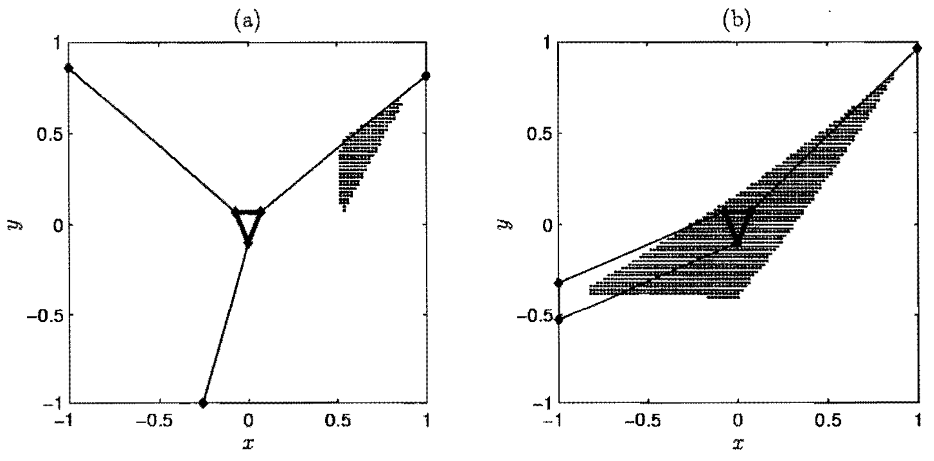

Figure 6.11: 3-cable TDPM (a) starting and (b) optimized design for L2 (R3)

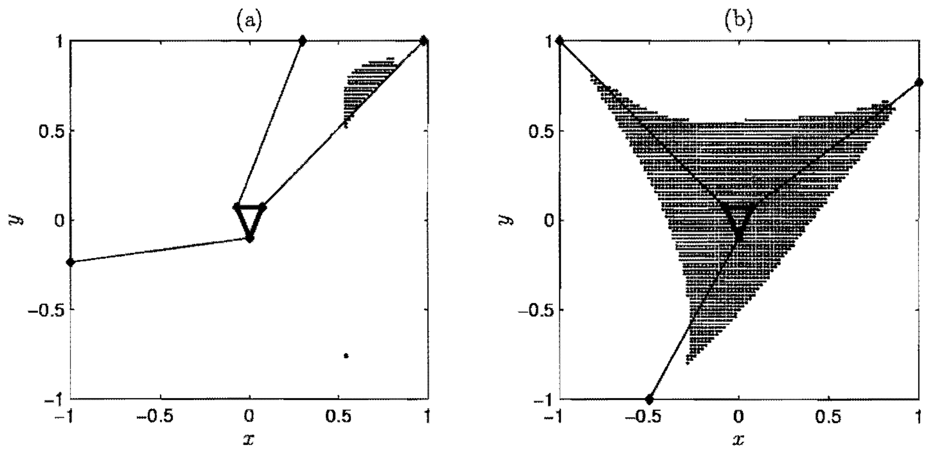

Figure 6.12: 3-cable TDPM (a) starting and (b) optimized design for L3 (R1)

Run	N^g	$\mathbf{d}^0$	A_d^0	$\mathbf{d}^*$	A_d^*
1	48	$[1.829, 2.321, 3.392, 3.741]^\top$	0.0256	$[0.8205, 2.338, 4.595, 4.769]^\top$	1.1040
2	47	$[2.139, 3.124, 3.216, 5.478]^\top$	0.1904	$[0.7878, 3.112, 3.287, 5.582]^\top$	0.9568
3	48	$[2.288, 2.601, 3.876, 4.117]^\top$	0.0144	$[0.8207, 2.335, 4.601, 4.775]^\top$	1.0992
4	14	$[0.7109, 2.219, 4.795, 4.997]^\top$	0.9472	$[0.8123, 2.325, 4.625, 4.799]^\top$	1.0992

Table 6.8: 4-cable TDPM optimized designs for L1

Run	N^g	$\mathbf{d}^0$	A_d^0	$\mathbf{d}^*$	A_d^*
1	34	$[0.0924, 2.607, 2.813, 5.982]^\top$	0.1424	$[0.7403, 3.173, 3.348, 5.564]^\top$	0.9760
2	39	$[1.917, 3.369, 3.745, 3.828]^\top$	0.0272	$[0.8215, 2.334, 4.585, 4.760]^\top$	1.1504
3	56	$[0.4313, 3.742, 4.899, 5.242]^\top$	0.0368	$[0.8194, 2.334, 4.591, 4.765]^\top$	1.1504
4	82	$[0.5235, 2.065, 4.784, 4.913]^\top$	0.5088	$[0.8146, 2.326, 4.609, 4.783]^\top$	1.1472

Table 6.9: 4-cable TDPM optimized designs for L2

the synthesis. For the 4-cable case, local maxima are also found during the design optimization. See for example run 2 for L1, run 1 for L2 and run 3 for L3.

6.7 Conclusion

The new constrained ℓ_2- and ℓ_1-norm approaches for determining cable tensions in overconstrained tendon-driven manipulators are reliable, and indeed critical for the accurate and correct determination of tendon-driven manipulator workspaces. Two methodologies for determining workspaces of planar

Run	N^g	$\mathbf{d}^0$	A_d^0	$\mathbf{d}^*$	A_d^*
1	55	$[1.347, 1.622, 2.036, 5.416]^\top$	0.0048	$[0.7572, 2.922, 3.097, 5.449]^\top$	1.3008
2	17	$[0.4202, 2.962, 3.531, 6.158]^\top$	0.1968	$[0.7636, 2.929, 3.104, 5.439]^\top$	1.2944
3	37	$[0.6810, 0.9393, 3.676, 5.416]^\top$	0.0016	$[1.452, 1.626, 3.843, 5.406]^\top$	0.8304
4	23	$[0.1010, 2.683, 4.338, 4.407]^\top$	0.2032	$[0.8095, 2.330, 4.444, 4.619]^\top$	1.1920

Table 6.10: 4-cable TDPM optimized designs for L3

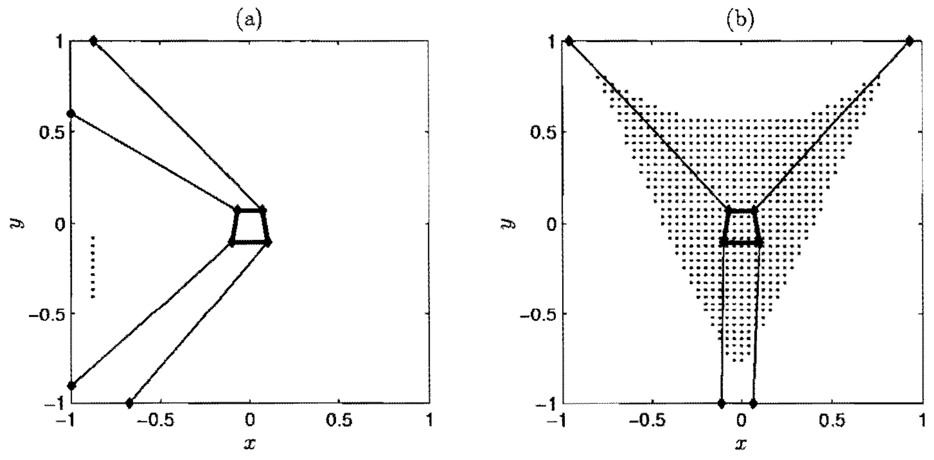

Figure 6.13: 4-cable TDPM (a) starting and (b) optimized design for L1 (R3)

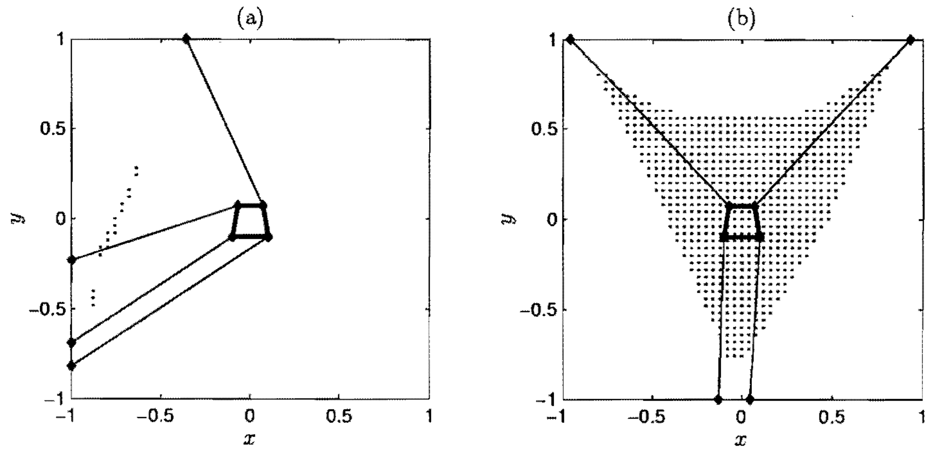

Figure 6.14: 4-cable TDPM (a) starting and (b) optimized design for L2 (R2)

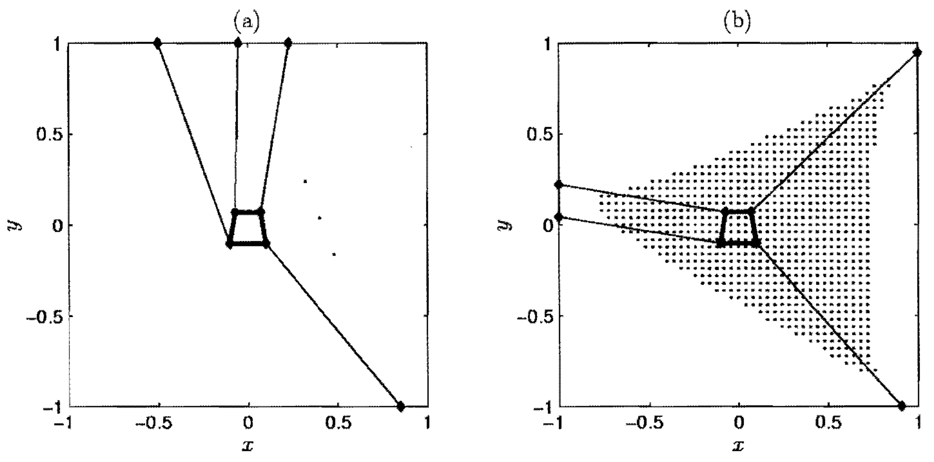

Figure 6.15: 4-cable TDPM (a) starting and (b) optimized design for L3 (R1)

tendon-driven manipulators are presented and evaluated by application to the 3-cable and 4-cable tendon-driven parallel manipulators. The discretization method is robust and reliable, but has a high computational requirement in comparison to the accuracy of the workspace determination. The chord method is accurate, reliable and efficient, but may require some user interaction in selecting the initial point for starting the workspace computation. In practice a combination of the two methods proves to be the most reliable and accurate, first using the discretization method at a low resolution to obtain a rough estimate of the workspace, and then reverting to the chord method to obtain an accurate and efficient mapping of the workspace boundary.

The dimensional synthesis yields TDPM designs with maximal dextrous workspaces for given static loads on the platform. The high dependance of TDPM workspaces on their design is illustrated, demonstrating the importance of dimensional synthesis of such manipulators.

Chapter 7

Conclusion

The main topic of this study was the optimal dimensional synthesis of planar parallel manipulators by means of numerical optimization techniques. In order to accomplish this, three specific issues needed to be addressed. These three issues, and achievements in each of these areas are discussed separately in the next three sections. In the final section of this chapter, recommendations for future refinement and development of the methods, developed during this study, are proposed.

7.1 Optimization algorithms

Two optimization algorithms were proposed and evaluated in the first part of this study. The spherical quadratic steepest descent (SQSD) optimization algorithm, presented in Chapter 2, provides a method for solving unconstrained optimization problems. Comparison of the performance of the algorithm with the classical steepest descent (SD) method indicates that the introduction of spherical quadratic subproblems dramatically improves the robustness of the method. In addition, the elimination of explicit line searches results in a much more efficient method than the SD method. The SQSD method performs well when compared to conjugate gradient methods. Particularly impressive is the ability of the SQSD method to solve ill-conditioned problems containing large numbers of variables, where the conjugate gradient methods break down, or take very long to converge. A proof of convergence for the SQSD method applied to general positive-definite quadratic functions is also given.

A version of the SQSD method, modified to solve constrained problems, is presented and tested in Chapter 3. This method is called the Dynamic-Q method, since the dynamic trajectory method of Snyman (see Appendix B) is used to solve the successive quadratic subproblems via a penalty function formulation. When compared to an SQP method using standard test problems, the Dynamic-Q method exhibits comparable efficiency and robustness. The Dynamic-Q method is however believed to be superior when applied to practical engineering problems containing phenomena such as numerical noise (as illustrated in Chapters 4 to 6). It also has the advantage that no Hessian information is required. This makes it a much more viable method for problems with very large number of variables.

User-friendly implementations of the SQSD and Dynamic-Q optimization algorithms have been programmed in FORTRAN and MATLAB.

7.2 Workspace determination

The original chord method, described in Appendix C, has been refined during the course of this study. The refinement, whereby the number of variables, contained in the optimization problem used for determining successive points on the workspace boundary, is reduced, results in a more efficient and accurate algorithm than the original implementation (see Section 5.4). In addition, a new scheme for determining bifurcation points has been developed in Section 6.4. This new scheme is slightly less accurate than the original methodology, but more generally applicable.

For the first time, the chord method has been applied to the determination of constant orientation workspaces of planar 3-$R\underline{P}R$ parallel manipulators. Additionally, the numerical multi-level optimization approach for dextrous workspace determination, based on the chord method and presented in Section 5.5, has successfully been used to determine dextrous workspaces of planar parallel manipulators. Constant orientation and dextrous workspaces are determined accurately and automatically.

The determination of workspaces of tendon-driven parallel manipulators is a challenging problem because the workspace is dependent primarily on the forces in the tendons. When considering over-constrained manipulators this factor is particularly important, since there is no unique solution to the cable tensions for a given position and load on the platform. Two new methodologies for determining cable tensions are proposed in Section 6.3.3. A further problem, related to workspace determination, is that the cable tensions may be discontinuous as the platform moves from one configuration to another. Two methodologies for determining workspaces of planar tendon-driven manipulators were developed. The first method, based on the discretization approach, is robust but in practical terms the accuracy of the method is limited by its high computational cost. As an alternative, the chord method is successfully applied to determining workspaces of tendon-driven manipulators, resulting in accurate and efficient determination of workspace boundaries. These methodologies and results are presented in Chapter 6.

7.3 Dimensional synthesis of manipulators

In Chapter 4 various strategies for optimizing parallel manipulators are investigated. The methodology thought to be the most practical is that presented in Section 4.7, which seeks to optimize the performance of the manipulator, while ensuring that a certain prescribed workspace can be reached. This methodology is successfully applied to a 2-$R\underline{P}R$ planar parallel manipulator, and then to the more complex 3-$R\underline{P}R$ manipulator in Chapter 5. For the 3-$R\underline{P}R$ manipulator various approaches are suggested for dealing with the orientational degree of freedom of the manipulator. In all cases optimal designs were found efficiently using the Dynamic-Q algorithm developed in Chapter 3.

An alternative synthesis approach was adopted in Chapter 6 for tendon-driven parallel manipulators. Here the objective of the optimization is to maximize the dextrous workspace which can be reached by the moving platform. The discretization method is used to evaluate the workspace areas in this case. The Dynamic-Q method proves its robustness by optimizing the manipulator, despite the numerical noise caused by determining the workspace in this manner.

7.4 Recommendations

It is believed that the full potential of the numerical methodologies developed in this work, applied here to planar manipulators, will be demonstrated when applied to more complex spatial cases. In terms of workspace determination it appears that the method for determining dextrous workspaces, presented in Section 5.5, can be extended to spatial parallel manipulators as well. The resulting methodology may provide an efficient numerical solution to this challenging problem.

Similarly, the constrained ℓ_1-norm approach developed in Section 6.3.3 for determination of cable tensions of overconstrained tendon-driven manipulators, as well as the methods for determining workspaces of tendon driven manipulators, could all be extended to the spatial case.

Finally, the methodologies for manipulator dimensional synthesis provide a meaningful alternative to existing methods. The formulations should be easily extended to include other performance criteria, and more complex manipulators. The inclusion of more design variables in these cases should easily and efficiently be dealt with by the Dynamic-Q method.

Bibliography

[1] J-P. Merlet. *Parallel Robots*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[2] J-P Merlet. Parallel manipulators: state of the art and perspectives. *Advanced Robotics*, 8(6):589–596, 1994.

[3] B. Dasgupta and T.S. Mruthyunjaya. The Stewart platform manipulator: a review. *Mechanism and Machine Theory*, 35:15–40, 2000.

[4] J-P Merlet. Personal website, Accessed September 2003. `http://www-sop.inria.fr/coprin/equipe/merlet/merlet_eng.html`.

[5] V. Zamanov and Z. Sotirov. A contribution to the serial and parallel manipulator duality. In *Proceedings of the 8th World Congress on the Theory of Machines and Mechanisms*, pages 517–520, Prague, Czechoslovakia, 26–31 August 1991.

[6] K.J. Waldron and K.H. Hunt. Series-parallel dualities in actively coordinated mechanisms. *The International Journal of Robotics Research*, 10(5):473–480, October 1991.

[7] J. Duffy. *Statics and Kinematics with Applications to Robotics*. Cambridge University Press, Cambridge, 1996.

[8] E.F. Fichter and E.D. MacDowell. A novel design for a robot arm. In *Proceedings of the ASME International Computer Technology Conference*, pages 250–256, San Francisco, August 1980.

[9] J.E. Gwinnett. Amusement devices. US Patent No. 1,789,680, 20 January 1931. `http://www.delphion.com/details?&pn10=US01789680`.

[10] I. Bonev. The true origins of parallel robots. *Online article*, Accessed September 2003. `http://www.parallemic.org/Reviews/Review007.html`.

[11] W.L.V. Pollard. Position controlling apparatus. US Patent No. 2,286,571, 16 June 1942.

[12] D. Stewart. A platform with six degrees of freedom. *Proceedings of the Institute of Mechanical Engineers*, 180(15):371–386, 1965-66. Part 1.

[13] V.E. Gough and S.G. Whitehall. Universal tyre test machine. In *Proceedings of the FISITA 9th International Technical Congress*, pages 117–137, May 1962.

[14] K.L. Cappel. Motion simulator. US Patent No. 3,295,224, 3 January 1967.

[15] J-P Merlet, C.M. Gosselin, and N. Mouly. Workspaces of planar parallel manipulators. *Mechanism and Machine Theory*, 33(1/2):7–20, 1998.

[16] E.J. Haug, C.M. Luh, C.M. Adkins, and J.Y. Wang. Numerical algorithms for mapping boundaries of manipulator workspaces. Concurrent Engineering Tools for Dynamic Analysis and Optimization: IUTAM Fifth Summer School on Mechanics, Aalborg, Denmark, 1994.

[17] E.J. Haug, J.Y. Wang, and J.K. Wu. Dextrous workspaces of manipulators. 1. Analytical Criteria. *Mechanics of Structures and Machines*, 20(3):321–361, 1992.

[18] L.J. Du Plessis and J.A. Snyman. A numerical method for the determination of dextrous workspaces of Gough-Stewart platforms. *International Journal for Numerical Methods in Engineering*, 52:345–369, 2001.

[19] J-P Merlet. Determination of the orientation workspace of parallel manipulators. *Journal of Intelligent and Robotic Systems*, 13:143–160, 1995.

[20] I.A. Bonev, D. Zlatanov, and C.M. Gosslelin. Advantages of the modified Euler angles in the design and control of PKMs. In R. Neugebauer, editor, *Proceedings of PKS2002 Parallel Kinematics Seminar*, pages 171–187, Chemnitz, Germany, 23–25 April 2002.

[21] C.M. Gosselin and J. Angeles. The optimum kinematic design of a planar three-degree-of-freedom manipulator. *Journal of Mechanisms, Transmissions and Automation in Design*, 110:35–41, 1988.

[22] C.M. Gosselin and M. Jean. Determination of the workspace of planar parallel manipulators with joint limits. *Robotics and Autonomous Systems*, 17:129–138, 1996.

[23] C.M. Gosselin. Determination of the workspace of 6-dof parallel manipulators. *Journal of Mechanical Design*, 112:331–336, September 1990.

[24] J-P Merlet. Détermination de l'espace de travail d'un robot parallèle pour une orientation constante. *Mechanism and Machine Theory*, 29(8):1099–1113, 1994.

[25] I.A. Bonev and J. Ryu. A geometrical method for computing the constant-orientation workspace of 6-$\underline{P}RRS$ parallel manipulators. *Mechanism and Machine Theory*, 36:1–13, 2001.

[26] I.A. Bonev and C.M. Gosselin. A geometric algorithm for the computation of the constant-orientation workspace of 6-$\underline{R}US$ parallel manipulators. In *Proceedings of the ASME 2000 Design Engineering Technical*

Conferences, Baltimore, Maryland, 10-13 September 2000. CD ROM Proceedings, Paper no. DETC2000/MECH-14106.

[27] T. Huang, J. Wang, C.M. Gosselin, and D. Whitehouse. Determination of closed form solution to the 2-d orientation workspace of Gough-Stewart parallel manipulators. *IEEE Transactions on Robotics and Automation*, 15(6):1121–1125, December 1999.

[28] A. Bajpai and B. Roth. Workspace and mobility of a closed-loop manipulator. *The International Journal of Robotics Research*, 5(2):131–142, 1986.

[29] R.L. Williams II and C.F. Reinholtz. Closed-form workspace determination and optimization for parallel robotic mechanisms. In *Proceedings of ASME 20th Biennial Mechanisms Conference*, pages 341–351, Kissimmee, Orlando, September 1988.

[30] D-Y Jo and E.J. Haug. Workspace analysis of multibody mechanical systems using continuation methods. *Journal of Mechanisms, Transmissions and Automation in Design*, 111:581–589, 1989.

[31] D-Y Jo and E.J. Haug. Workspace analysis of closed-loop mechanisms with unilateral constraints. *ASME Advances in Design Automation*, 19(3):53–60, 1989.

[32] E.J. Haug, F.A. Adkins, C. Qiu, and J. Yen. Analysis of barriers to control of manipulators within accessible output sets. Technical report, University of Iowa, Center for Computer Aided Design and Department of Mechanical Engineering, 1994.

[33] E.J. Haug, F.A. Adkins, and C-M Luh. Operational envelopes for working bodies of mechanisms and manipulators. *Journal of Mechanical Design*, 120:84–91, March 1998.

[34] F.A. Adkins and E.J. Haug. Operational envelope of a spatial Stewart platform. *Journal of Mechanical Design*, 119:330–332, June 1997.

[35] E.J. Haug, F.A. Adkins, C-M Luh, and J-Y Wang. Domains of interference between working bodies in mechanisms and manipulators. In *ASME Advances in Design Automation*, volume 82, pages 651–658, September 1995.

[36] E.J. Haug, F.A. Adkins, and C-M Luh. Domains of operation and interference for bodies in mechanisms and manipulators. In J-P Merlet and B. Ravani, editors, *Computational Kinematics*, pages 193–202. Kluwer Academic Publishers, 1995.

[37] E.J. Haug, F.A. Adkins, and C-M Luh. Numerical methods for mechanism and manipulator workspace analysis. In J. Angeles and E. Zakhariev, editors, *Computational Methods in Mechanical Systems: Mechanism Analysis, Synthesis and Optimization*, pages 145–163. Springer-Verlag, 1998.

[38] D.C.H. Yang and T.W. Lee. Feasibility study of robotic manipulators from a kinematic viewpoint. *Journal of Mechanisms, Transmissions and Automation in Design*, 106:191–198, June 1984.

[39] M. Sorli and M. Ceccarelli. On the workspace of a 6-dof platform with articulated double-parallelograms. In *Proceedings of the 6th International Conference on Advanced Robotics*, pages 147–152, Tokyo, 1993.

[40] J.J. Cervantes-Sánchez and J.G. Rendón-Sánchez. A simplified approach for obtaining the workspace of a class of 2-dof planar parallel manipulators. *Mechanism and Machine Theory*, 34:1057–1073, 1999.

[41] M. Ceccarelli and M. Sorli. The effects of design parameters on the workspace of the Turin robot. *The International Journal of Robotics Research*, 17(8):886–902, August 1998.

[42] E.F. Fichter. A Stewart platform-based manipulator: general theory and practical construction. *The International Journal of Robotics Research*, 5(2):157–182, 1986.

[43] K.M. Lee and D.K. Shah. Kinematic analysis of a three-degrees-of-freedom in-parallel actuated manipulator. *IEEE Journal of Robotics and Automation*, 4(3):354–360, June 1988.

[44] O. Masory and J. Wang. Workspace evaluation of Stewart platforms. *ASME Robotics, Spatial Mechanisms, and Mechanical Systems*, 45:337–346, 1992.

[45] T. Arai, T. Tanikawa, J-P Merlet, and T Sendai. Development of a new parallel manipulator with fixed linear actuator. In *Proceedings of the ASME Japan/USA Symposium on Flexible Automation*, volume 1, pages 145–149, 1996.

[46] R. Stamper, L-W. Tsai, and G.C. Walsh. Optimization of a three-dof translational platform for well-conditioned workspace. In *IEEE International Conference on Robotics and Automation*, pages 3250–3255, Albuquerque, New Mexico, 21–28 April 1997.

[47] Z. Wang, Z. Wang, W. Liu, and Y. Lei. A study on workspace, boundary workspace analysis and workpiece positioning for parallel machine tools. *Mechanism and Machine Theory*, 36:605–622, 2001.

[48] J.A. Snyman, L.J. du Plessis, and J. Duffy. An optimization approach to the determination of the boundaries of manipulator workspaces. *Journal of Mechanical Design*, 122:447–455, 2000.

[49] A.M. Hay and J.A. Snyman. The determination of nonconvex workspaces of generally constrained planar Stewart platforms. *Computers and Mathematics with Applications*, 40:1043–1060, 2000.

[50] A.M. Hay and J.A. Snyman. The chord method for the determination of non-convex workspaces of planar parallel manipulators. *Computers and Mathematics with Applications*, 43:1135–1151, 2002.

[51] L.J. Du Plessis. An optimization approach to the determination of manipulator workspaces. Master's thesis, University of Pretoria, Department of Mechanical and Aeronautical Engineering, Pretoria, 1998.

[52] J.A. Snyman and A.M. Hay. The Dynamic-Q optimisation method: An alternative to SQP? *Computers and Mathematics with Applications*, 44:1589–1598, 2002.

[53] A.M. Hay. Methods for the determination of accessible workspaces of planar Stewart platforms of general design. Master's thesis, University of Pretoria, Department of Mechanical and Aeronautical Engineering, Pretoria, 1999.

[54] L.T. Wang and J. Hsieh. Extreme reaches and reachable workspace analysis of general parallel robotic manipulators. *Journal of Robotic Systems*, 15(3):145–159, 1998.

[55] S. Bhattacharya, H. Hatwal, and A. Ghosh. On the optimum design of Stewart platform type parallel manipulators. *Robotica*, 13:133–140, 1995.

[56] D. Chakarov. Study of the passive compliance of parallel manipulators. *Mechanism and Machine Theory*, 34:373–389, 1999.

[57] V. Hayward, C. Nemri, X. Chen, and B. Duplat. Kinematic decoupling in mechanisms and application to a passive hand controller design. *Journal of Robotic Systems*, 10(5):767–790, 1993.

[58] X-J. Liu, Z-L. Jin, and F. Gao. Optimum design of 3-dof spherical parallel manipulators with respect to the conditioning and stiffness indices. *Mechanism and Machine Theory*, 35:1257–1267, 2000.

[59] D. Zhang, F. Xi, and C. Mechefske. Optimization of reconfigurable parallel mechanisms with revolute actuators. In *CD-ROM Proceedings of the 1st CIRP International Conference on Reconfigurable Manufacturing*, Ann Arbor, Michigan, May 2001.

[60] N. Simaan and M. Shoham. Stiffness synthesis of a variable geometry planer robot. In J. Lenarčič and F. Thomas, editors, *Advances in Robot Kinematics*, pages 463–472, Caldes de Malavella, Spain, June 2002. Kluwer Academic Publishers.

[61] C.M. Gosselin. On the design of efficient parallel mechanisms. In J. Angeles and E. Zakhariev, editors, *Computational Methods in Mechanical Systems: Mechanism Analysis, Synthesis and Optimization*, pages 68–96. Springer-Verlag, 1998.

[62] J. Wang and C.M. Gosselin. Static balancing of spatial three-degree-of-freedom parallel mechanisms. *Mechanism and Machine Theory*, 34(3):437–452, 1999.

[63] J. Wang and C.M. Gosselin. Static balancing of spatial four-degree-of-freedom parallel mechanisms. *Mechanism and Machine Theory*, 35(4):563–592, 2000.

[64] C.M. Gosselin and J. Wang. Static balancing of spatial six-degree-of-freedom parallel mechanisms with revolute actuators. *Journal of Robotic Systems*, 17(3):159–170, 2000.

[65] C.M. Gosselin, J. Wang, T. Laliberté, and I. Ebert-Uphoff. On the design of a statically balanced 6-dof parallel manipulator. In *Proceedings of the 10th World Conference on the Theory of Machines and Mechanisms*, pages 1045–1050, Oulu, Finland, 21–24 June 1999.

[66] J.L. Herder. Some considerations regarding statically balanced parallel mechanisms. In C.M. Gosselin and I. Ebert-Uphoff, editors, *Proceedings of the Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 40–45, Quebec City, Quebec, 3–4 October 2002.

[67] M. Weck and M. Giesler. Task-oriented multi-objective-optimization of parallel kinematics for machine tools. In R. Neugebauer, editor,

Proceedings of PKS2002 Parallel Kinematics Seminar, pages 197–211, Chemnitz, Germany, 23–25 April 2002.

[68] C.A. Klein and B.E. Blaho. Dexterity measures for the design and control of kinematically redundant manipulators. *The International Journal of Robotics Research*, 6(2):72–83, Summer 1987.

[69] C.M. Gosselin and J. Angeles. The optimum kinematic design of a spherical three-degree-of-freedom parallel manipulator. *Journal of Mechanisms, Transmissions and Automation in Design*, 111:202–207, 1989.

[70] C.M. Gosselin and E. Lavoie. On the kinematic design of spherical three-degree-of-freedom parallel manipulators. *The International Journal of Robotics Research*, 12(4):394–402, August 1993.

[71] K.H. Pittens and R.P. Podhorodeski. A family of Stewart platforms with optimal dexterity. *Journal of Robotic Systems*, 10(4):463–479, 1993.

[72] K.E. Zanganeh and J. Angeles. Kinematic isotropy and the optimum design of parallel manipulators. *The International Journal of Robotics Research*, 16(2):185–197, April 1997.

[73] J-P Merlet. An initiative for the kinematics study of parallel robots. In C.M. Gosselin and I. Ebert-Uphoff, editors, *Proceedings of the Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 2–9, Quebec City, Quebec, 3–4 October 2002. Keynote address.

[74] C.M. Gosselin and J. Angeles. A global performance index for the kinematic optimization of robotic manipulators. *Journal of Mechanical Design*, 113:220–226, 1991.

[75] L-W. Tsai and S. Joshi. Kinematics and optimization of a spatial 3-UPU parallel manipulator. *Journal of Mechanical Design*, 122:439–446, 2000.

[76] R. Kurtz and V. Hayward. Multiple-goal kinematic optimization of a parallel spherical mechanism with actuator redundancy. *IEEE Transactions on Robotics and Automation*, 8(5):644–651, 1992.

[77] S. Leguay-Durand and C. Reboulet. Optimal design of a redundant spherical parallel manipulator. *Robotica*, 15:399–405, 1997.

[78] R.S. Stoughton and T. Arai. A modified Stewart platform manipulator with improved dexterity. *IEEE Transactions on Robotics and Automation*, 9(2):166–173, 1993.

[79] J. Lee, J. Duffy, and M. Keler. The optimum quality index for the stability of in-parallel planar platform devices. In *Proceedings of the ASME 24th Biennial Mechanisms Conference*, Irvine, California, 18–22 August 1996. Paper no. 96-DETC/MECH1135.

[80] J. Lee, J. Duffy, and K.H. Hunt. The optimum quality index for some spatial in-parallel devices. In *Proceedings of the 1997 Florida Conference on Recent Advances in Robotics*, Miami, Florida, 10–11 April 1997.

[81] Y. Zhang and J Duffy. The optimum quality index for a redundant 4-4 in-parallel manipulator. In *RoManSy 12: Theory and Practice of Robots and Manipulators, Proceedings of the Twelfth CISM-IFToMM Symposium*, pages 289–296, Paris, 1998.

[82] J. Lee, J. Duffy, and K.H. Hunt. A practical quality index based on the octahedral manipulator. *The International Journal of Robotics Research*, 17(10):1081–1090, 1998.

[83] J.A. Carretero, R.P. Podhorodeski, M.A. Nahon, and C.M. Gosselin. Kinematic analysis and optimization of a new three degree-of-freedom

spatial parallel manipulator. *Journal of Mechanical Design*, 122:17–24, March 2000.

[84] C.M. Gosselin and M. Guillot. The synthesis of manipulators with prescribed workspace. *Journal of Mechanical Design*, 113:451–455, 1991.

[85] R. Boudreau and C.M. Gosselin. The synthesis of planar parallel manipulators with a genetic algorithm. *Journal of Mechanical Design*, 121:533–537, 1999.

[86] R. Boudreau and C.M. Gosselin. La synthèse d'une plate-forme de Gough-Stewart pour un espace atteignable prescrit. *Mechanism and Machine Theory*, 36:327–342, 2001.

[87] A.P. Murray, F. Pierrot, P. Dauchez, and J.M. McCarthy. A planar quaternion approch to the kinematic synthesis of a parallel manipulator. *Robotica*, 15:361–365, 1997.

[88] A. Murray and M. Hanchak. Kinematic synthesis of planar platforms with RPR, PRR and RRR chains. In J. Lenarčič and M.M. Stanišić, editors, *Advances in Robot Kinematics*, pages 285–292, Piran, Slovenia, June 2000. Kluwer Academic Publishers.

[89] A. Perez and McCarthy. Dual quaternion synthesis of a 2-TPR constrained parallel robot. In C.M. Gosselin and I. Ebert-Uphoff, editors, *Proceedings of the Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 150–157, Quebec City, Quebec, 3–4 October 2002.

[90] J-P Merlet. Designing a parallel manipulator for a specific workspace. Research Report 2527, INRIA, Sophia-Antipolis, France, April 1995.

[91] J-P Merlet. Designing a parallel manipulator for a specific workspace. *The International Journal of Robotics Research*, 16(4):545–556, 1997.

[92] J-P Merlet. DEMOCRAT: A design methodology for the conception of robots with parallel architecture. *Robotica*, 15:367–373, 1997.

[93] Z. Ji. Analysis of design parameters in platform manipulators. *Journal of Mechanical Design*, 118:526–531, 1996.

[94] J. Kirchner and R. Neugebauer. How to optimize parallel link mechanisms – proposal of a new strategy. In *Year 2000 Parallel Kinematic Machines International Conference*, pages 307–315, Ann Arbor, Michigan, 13–15 September 2000.

[95] M. Gallant and B. Boudreau. The synthesis of planar parallel manipulators with prismatic joint for an optimal, singularity-free workspace. *Journal of Robotic Systems*, 19(1):13–24, 2002.

[96] C.M. Gosselin. Optimization of multi-dof mechanisms. In J. Angeles and E. Zakhariev, editors, *Computational Methods in Mechanical Systems: Mechanism Analysis, Synthesis and Optimization*, pages 97–129. Springer-Verlag, 1998.

[97] M.S. Bazaraa, H.D. Sherali, and C.M. Shetty. *Nonlinear Programming, Theory and Algorithms*. John Wiley, New York, 1993. p.300.

[98] R. Fletcher and C.M. Reeves. Function minimization by conjugate gradients. *Computer Journal*, 7:149–154, 1964.

[99] E. Polak and G. Ribière. Note sur la convergence de methods de directions conjugées. *Revue Française Informat, Recherche Operationelle*, 13:35–43, 1969.

[100] J.A. Snyman. Practical mathematical optimization: An introduction to basic optimization theory and algorithms. Optimization course notes, Department of Mechanical Engineering, University of Pretoria, 2003.

[101] J.A. Snyman. A new and dynamic method for unconstrained minimization. *Applied Mathematical Modelling*, 6:449–462, 1982.

[102] J.A. Snyman. An improved version of the original leap-frog method for unconstrained minimization. *Applied Mathematical Modelling*, 7:216–218, 1983.

[103] J.A. Snyman. The LfopC leap-frog algorithm for constrained optimization. *Computers and Mathematics with Applications*, 40:1085–1096, 2000.

[104] D. Goldberg. *Genetic algorithms in search optimization and machine learning.* Addison-Wesley, Reading, MA, 1989.

[105] Z. Michalewicz. *Genetic algorithms and data structures : evolution programs.* Springer-Verlag, Berlin, 3rd edition, 1992.

[106] S. Kirkpatrick, J.R. Gellatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1985.

[107] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1091, 1953.

[108] C.M. Gosselin, L. Perrault, and C. Vaillancourt. Simulation and computer-aided kinematic design of three-degree-of-freedom spherical parallel manipulators. *Journal of Robotic Systems*, 12(12):857–869, 1995.

[109] J-P Merlet. The need for a systematic methodology for the evaluation and optimal design of parallel manipulators. In R. Neugebauer, editor, *Proceedings of PKS2002 Parallel Kinematics Seminar*, pages 49–61, Chemnitz, Germany, 23–25 April 2002. Keynote address.

[110] J-P Merlet. Still a long way to go on the road for parallel mechanisms. In *Proceedings of the ASME 2002 Design Engineering Technical Conferences*, Montréal, Quebec, 29 September–2 October 2002. CD-ROM proceedings: Keynote address.

[111] J.C. Gilbert and J. Nocedal. Global convergence properties of conjugate gradient methods. *SIAM Journal on Optimization*, 2:21–42, 1992.

[112] J.J. Moré and D. Thuente. Linesearch algorithms with guaranteed sufficient decrease. *ACM Transactions on Mathematical Software*, 20:286–307, 1994.

[113] S.S. Rao. *Engineering Optimization, Theory and Practice.* John Wiley, New York, 1994.

[114] J.A. Snyman. Unconstrained minimization by combining the dynamic and conjugate gradient methods. *Quaestiones Mathematicae*, 8:33–44, 1985.

[115] D.M. Himmelblau. *Applied Nonlinear Programming.* McGraw-Hill, New York, 1972.

[116] A.I. Manevich. Perfection of the conjugate directions method for unconstrained minimization problems. In *Third World Congress of Structural and Multidisciplinary Optimization*, volume 1, Buffalo, New York, 17–21 May 1999.

[117] P.Y. Papalambros and D.J. Wilde. *Principles of Optimal Design.* Cambridge University Press, Cambridge, 1993.

[118] M.J.D. Powell. Algorithms for nonlinear constraints that use Lagrangian functions. *Mathematical Programming*, 14:224–248, 1978.

[119] J.A. Snyman, W.J. Roux, and N. Stander. A dynamic penalty function method for the solution of structural optimization problems. *Applied Mathematical Modelling*, 18:435–460, 1994.

[120] W. Hock and K. Schittkowski. *Lecture Notes in Economics and Mathematical Systems. No 187: Test examples for nonlinear programming codes.* Springer-Verlag, Berlin, Heidelberg, New York., 1981.

[121] K. Svanberg. The MMA for modeling and solving optimization problems. In *Proceedings of the 3rd World Conference on Structural and Multidisciplinary Optimization*, Buffalo, New York, 17–21 May 1999.

[122] J.A. Snyman and A.M. Hay. The chord method for the determination of non-convex workspaces of planar parallel platforms. In J. Lenarčič and M.M. Stanišić, editors, *Advances in Robot Kinematics*, pages 285–292, Piran, Slovenia, June 2000. Kluwer Academic Publishers.

[123] N.G. Dagalakis, J.S. Albus, B.L. Wang, J. Unger, and J.D. Lee. Stiffness study of a parallel link robot crane for shipbuilding applications. *ASME Journal of Offshore Mechanics and Arctic Engineering*, 111(3):183–193, 1989.

[124] R. Bostelman, J. Albus, N. Dagalakis, A. Jacoff, and A. Gross. Applications of the NIST Robocrane. *Robotics and Manufacturing*, 5:403–410, 1994.

[125] R. Verhoeven, M. Hiller, and S. Tadokoro. Workspace of tendon-driven Stewart platforms: Basics, classification, details on the planar 2-dof class. In *Proceedings of the 4th International Conference on Motion and Vibration Control*, ETH Zurich, Switzerland, 25–28 August 1998.

[126] R. Bostelman, W. Shackleford, F. Proctor, J. Albus, and A. Lytle. The Flying Carpet: A tool to improve ship repair efficiency. Technical report, National Institute of Standards and Technology,USA, 2002. Online report: `www.nsrp.org/projects/deliverables/flying_carpet.pdf`.

[127] R. Verhoeven and M. Hiller. Tension distribution in tendon-based Stewart platforms. In J. Lenarčič and F. Thomas, editors, *Advances in Robot Kinematics*, pages 117–124, Caldes de Malavella, Spain, June 2002. Kluwer Academic Publishers.

[128] P. Lafourcade, M. Llibre, and C. Reboulet. Design of a parallel wire-driven manipulator for wind tunnels. In C.M. Gosselin and I. Ebert-Uphoff, editors, *Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 187–194, Quebec City, Quebec, Canada, 3–4 October 2002.

[129] R. Verhoeven and M. Hiller. Estimating the controllable workspace of tendon-based Stewart platforms. In J. Lenarčič and M.M. Stanišić, editors, *Advances in Robot Kinematics*, pages 277–284, Piran, Slovenia, June 2000. Kluwer Academic Publishers.

[130] A. Fattah and S.K. Agrawal. Design of cable-suspended planar parallel robots for an optimal workspace. In C.M. Gosselin and I. Ebert-Uphoff, editors, *Proceedings of the Workshop on Fundamental Issues and Future Research Directions for Parallel Mechanisms and Manipulators*, pages 195–202, Quebec City, Quebec, 3–4 October 2002.

Appendix A

Test functions used for SQSD

Minimize $f(\mathbf{x})$:

1. $f(\mathbf{x}) = x_1^2 + 2x_2^2 + 3x_3^2 - 2x_1 - 4x_2 - 6x_3 + 6$, $\mathbf{x}^0 = [3,3,3]^\top$, $\mathbf{x}^* = [1,1,1]^\top$, $f(\mathbf{x}^*) = 0.0$

2. $f(\mathbf{x}) = x_1^4 - 2x_1^2x_2 + x_1^2 + x_2^2 - 2x_1 + 1$, $\mathbf{x}^0 = [3,3]^\top$, $\mathbf{x}^* = [1,1]^\top$, $f(\mathbf{x}^*) = 0.0$

3. $f(\mathbf{x}) = x_1^4 - 8x_1^3 + 25x_1^2 + 4x_2^2 - 4x_1x_2 - 32x_1 + 16$, $\mathbf{x}^0 = [3,3]^\top$, $\mathbf{x}^* = [2,1]^\top$, $f(\mathbf{x}^*) = 0.0$

4. $f(\mathbf{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$, $\mathbf{x}^0 = [-1.2,1]^\top$, $\mathbf{x}^* = [1,1]^\top$, $f(\mathbf{x}^*) = 0.0$ (Rosenbrock's parabolic valley [113])

5. $f(\mathbf{x}) = x_1^4 + x_1^3 - x_1 + x_2^4 - x_2^2 + x_2 + x_3^2 - x_3 + x_1x_2x_3$, (Zlobec's function [114])

   (a) $\mathbf{x}^0 = [1,-1,1]^\top$ and

   (b) $\mathbf{x}^0 = [0,0,0]^\top$, $\mathbf{x}^* = [0.57085597, -0.93955591, 0.76817555]^\top$, $f(\mathbf{x}^*) = -1.91177218907$

6. $f(\mathbf{x}) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$, $\mathbf{x}^0 = [3, -1, 0, 1]^\mathsf{T}$, $\mathbf{x}^* = [0, 0, 0, 0]^\mathsf{T}$, $f(\mathbf{x}^*) = 0.0$ (Powell's quartic function [113])

7. $f(\mathbf{x}) = -\left\{\frac{1}{1+(x_1-x_2)^2} + \sin\left(\frac{1}{2}\pi x_2 x_3\right) + \exp\left[-\left(\frac{x_1+x_3}{x_2} - 2\right)^2\right]\right\}$, $\mathbf{x}^0 = [0, 1, 2]^\mathsf{T}, \mathbf{x}^* = [1, 1, 1]^\mathsf{T}$, $f(\mathbf{x}) = -3.0$ [113]

8. $f(\mathbf{x}) = \{-13 + x_1 + [(5 - x_2)x_2 - 2]x_2\}^2 + \{-29 + x_1 + [(x_2 + 1)x_2 - 14]x_2\}^2$, $\mathbf{x}^0 = [1/2, -2]^\mathsf{T}$, $\mathbf{x}^* = [5, 4]^\mathsf{T}, f(\mathbf{x}^*) = 0.0$ (Freudenstein and Roth function [113])

9. $f(\mathbf{x}) = 100(x_2 - x_1^3)^2 + (1 - x_1)^2$, $\mathbf{x}^0 = [-1.2, 1]^\mathsf{T}$, $\mathbf{x}^* = [1, 1]^\mathsf{T}$, $f(\mathbf{x}^*) = 0.0$ (cubic valley [115])

10. $f(\mathbf{x}) = [1.5 - x_1(1 - x_2)]^2 + [2.25 - x_1(1 - x_2^2)]^2 + [2.625 - x_1(1 - x_2^3)]^2$, $\mathbf{x}^0 = [1, 1]^\mathsf{T}$, $\mathbf{x}^* = [3, 1/2]^\mathsf{T}$, $f(\mathbf{x}^*) = 0.0$ (Beale's function [113])

11. $f(\mathbf{x}) = [10(x_2 - x_1^2)]^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 + 10(x_2 + x_4 - 2)^2 + 0.1(x_2 - x_4)^2$, $\mathbf{x}^0 = [-3, 1, -3, -1]^\mathsf{T}$, $\mathbf{x}^* = [1, 1, 1, 1]^\mathsf{T}$, $f(\mathbf{x}^*) = 0.0$ (Wood's function [113])

12. $f(\mathbf{x}) = \sum_{i=1}^{n} i x_i^2$, $\mathbf{x}^0 = [3, 3, \ldots, 3]^\mathsf{T}$, $\mathbf{x}^* = [0, 0, \ldots, 0]^\mathsf{T}$, $f(\mathbf{x}^*) = 0.0$ (extended homogeneous quadratic functions)

13. $f(\mathbf{x}) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$, $\mathbf{x}^0 = [-1.2, 1, -1.2, 1, \ldots]^\mathsf{T}$, $\mathbf{x}^* = [1, 1, \ldots, 1]^\mathsf{T}$, $f(\mathbf{x}^*) = 0.0$ (extended Rosenbrock functions [113])

14. $f(\mathbf{x}) = \sum_{i=1}^{n} (1 - x_i)^2 / 2^{i-1}$, $\mathbf{x}^0 = [0, 0, \ldots, 0]^\mathsf{T}$, $\mathbf{x}^* = [1, 1, \ldots, 1]^\mathsf{T}$, $f(\mathbf{x}^*) = 0.0$ (extended Manevich functions [116])

Appendix B

The dynamic trajectory optimization algorithm

B.1 Background

The dynamic trajectory method (also called the "leap-frog" method) for the *unconstrained* minimization of a scalar function $f(\mathbf{x})$ of n real variables represented by the vector $\mathbf{x} = [x_1, x_2, \ldots, x_n]^T$ was originally proposed by Snyman [101, 102]. The original algorithm has also been modified to handle constrained problems by means of a penalty function formulation (Snyman *et al.* [119] and Snyman [103]). The method possesses the following characteristics:

- It uses only function *gradient* information $\nabla f(\mathbf{x})$,
- *No* explicit *line searches* are performed,
- It is extremely *robust*: handles steep valleys and discontinuities in functions and gradients,
- Algorithm seeks a *low local minimum* and can therefore be used as a

basic component in a methodology for global minimization,

- The method is not as efficient as classical methods on smooth or near-quadratic functions

B.2 Basic dynamic model

In its unconstrained form, the leap-frog optimizer (Lfop) determines the minimum of a function $f(\mathbf{x})$, by considering the associated dynamic problem of the motion of a particle (of unit mass) in an n-dimensional conservative force field, where the potential energy of the particle at a point $\mathbf{x}(t)$ and time t, is given by $f(\mathbf{x})$. The method thus requires the solution of the equations of motion of the particle. At $\mathbf{x}$ the force on the particle is given by

$$\mathbf{a} = \ddot{\mathbf{x}}(t) = -\boldsymbol{\nabla} f(\mathbf{x}(t)) \tag{B.1}$$

subject to initial conditions

$$\mathbf{x}(0) = \mathbf{x}^0, \;\; \dot{\mathbf{x}}(0) = \mathbf{v}^0 \tag{B.2}$$

To explain how the dynamic trajectory method works, consider the solution of the above problem over the time interval $[0, t]$. It follows that

$$\begin{gathered} \tfrac{1}{2}\|\dot{\mathbf{x}}(t)\|^2 - \tfrac{1}{2}\|\mathbf{v}^0\|^2 = f(\mathbf{x}^0) - f(\mathbf{x}(t)) \\ T(t) - T(0) = f(0) - f(t) \\ \text{or } f(t) + T(t) = f(0) + T(0) = K \end{gathered} \tag{B.3}$$

Here $T(t)$ is used to denote the kinetic energy of the particle at time t and K is a constant determined by the initial values. The last expression in (B.3) indicates that energy is conserved. It can also be seen that $\Delta f = -\Delta T$, therefore as long as T increases, f decreases. This forms the basis of the dynamic trajectory method.

B.3 Basic algorithm for unconstrained problems

Given $f(\mathbf{x})$, and starting point $\mathbf{x}(0) = \mathbf{x}^0$, the Lfop algorithm computes an approximation to the trajectory followed by the particle in the force field by solving the initial value problem (B.1) and (B.2).

The algorithm monitors the velocity $\dot{\mathbf{x}}(t) = \mathbf{v}(t)$ of the particle. Clearly as long as $T = \frac{1}{2}\|\mathbf{v}(t)\|^2$ is *increasing* along the trajectory, $f(\mathbf{x}(t))$ is decreasing and the algorithm is minimizing the function. However, whenever T *decreases* along the trajectory, the objective function (potential energy) is increasing. An interfering strategy is then applied to extract kinetic energy from the particle. The consequence of this strategy, based on an energy conservation argument, is that a systematic reduction in the potential energy $f(\mathbf{x})$ of the particle is obtained, and the likelihood of descent is increased. The particle is thus forced to follow a path to a local minimum at $\mathbf{x}^*$.

The numerical integration of the initial value problem (B.1) and (B.2) is achieved using the "leap-frog" (Euler forward-Euler backward) method, by computing for $k = 0, 1, 2, \ldots$, and time step Δt

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{x}^k + \mathbf{v}^k \Delta t \\ \mathbf{v}^{k+1} &= \mathbf{v}^k + \mathbf{a}^{k+1} \Delta t \end{aligned} \tag{B.4}$$

where $\mathbf{a}^{k+1} = -\boldsymbol{\nabla} f(\mathbf{x}^{k+1})$ and $\mathbf{v}^0 = \frac{1}{2}\mathbf{a}^0 \Delta t$.

A typical interfering strategy, implemented when $\|\mathbf{v}^{k+1}\| \leq \|\mathbf{v}^k\|$ is to set

$$\mathbf{v}^k = \frac{\mathbf{v}^{k+1} + \mathbf{v}^k}{4} \text{ and } \mathbf{x}^k = \frac{\mathbf{x}^{k+1} + \mathbf{x}^k}{2} \tag{B.5}$$

and then use these new values of $\mathbf{v}^k$ and $\mathbf{x}^k$ to compute the new $\mathbf{v}^{k+1}$ using (B.4).

The method contains some heuristic elements to determine an initial Δt, to allow for reduction and magnification of Δt, and to control the step size used in the algorithm.

B.4 Modification for constrained problems

The Lfop algorithm outlined above can be modified to handle constrained problems by means of the penalty function approach (LfopC) [119, 103]. Given a *constrained* optimization problem with objective function $f(\mathbf{x})$, $\mathbf{x} \in \Re^n$, inequality constraint functions $g_j(\mathbf{x}) \leq 0,\ j = 1, 2, \ldots, p$, equality constraint functions $h_k(\mathbf{x}) \leq 0,\ k = 1, 2, \ldots, q$ and penalty parameters α_j and β_k, the associated *unconstrained* optimization problem given by the penalty function formulation is

$$Q(\mathbf{x}) = f(\mathbf{x}) + \textstyle\sum_{j=1}^{p} \varrho_j g_j^2(\mathbf{x}) + \sum_{k=1}^{q} \beta_k h_k^2(\mathbf{x}) \tag{B.6}$$

$$\text{where } \varrho_j = \begin{cases} 0 \text{ if } g_j(\mathbf{x}) \leq 0 \\ \alpha_j \text{ if } g_j(\mathbf{x}) > 0 \end{cases}$$

For simplicity the penalty parameters α_j and β_k usually take on the same positive value $\alpha_j = \beta_k = \mu$. It can be shown that as μ tends to infinity, the unconstrained minimum of $Q(\mathbf{x})$ yields the solution to the constrained optimization problem. The Lfop dynamic trajectory method is applied to the penalty function formulation of the constrained problem in three phases.

PHASE 0: Given some starting point $\mathbf{x}^0$, apply Lfop with some overall penalty parameter $\mu = \mu_0 (= 10^2)$ to $Q(\mathbf{x}, \mu_0)$ to give $\mathbf{x}^*(\mu_0)$.

PHASE 1: With $\mathbf{x}^0 := \mathbf{x}^*(\mu_0)$, apply Lfop with increased overall penalty parameter $\mu = \mu_1 (= 10^4) \gg \mu_0$ to $Q(\mathbf{x}, \mu_1)$ to give $\mathbf{x}^*(\mu_1)$. Identify the set of n_a active constraints corresponding to the set of subscripts $I_a = (u1, u2, \ldots, un_a)$ for which $g_{uj}(\mathbf{x}^*(\mu_1)) > 0,\ j = 1, 2, \ldots, n_a$.

PHASE 2: With $\mathbf{x}^0 := \mathbf{x}^*(\mu_1)$, apply Lfop to

$$\min_{\mathbf{x}} Q_a(\mathbf{x}, \mu_1) = \sum_{j=1}^{n_a} \mu_1 g_{uj}^2(\mathbf{x}) + \sum_{k=1}^{q} \mu_1 h_k^2(\mathbf{x}) \tag{B.7}$$

to give $\mathbf{x}^*$.

Appendix C

Review of the chord method for workspace determination

C.1 Introduction

This appendix summarizes the chord method for maximal workspace determination as proposed by Hay and Snyman [50]. Further references related to the chord method, and the optimization approach for workspace determination are given in Section 1.3.2. In the next two sections of this appendix, definitions necessary for the discussion of the workspace determination method are given. The most important basic components of the chord method are determining an initial point on the workspace boundary, mapping the workspace boundary using constant chord length searches, and the accurate determination of bifurcation points which occur along the workspace boundary. Each of these components of are discussed in separate sections.

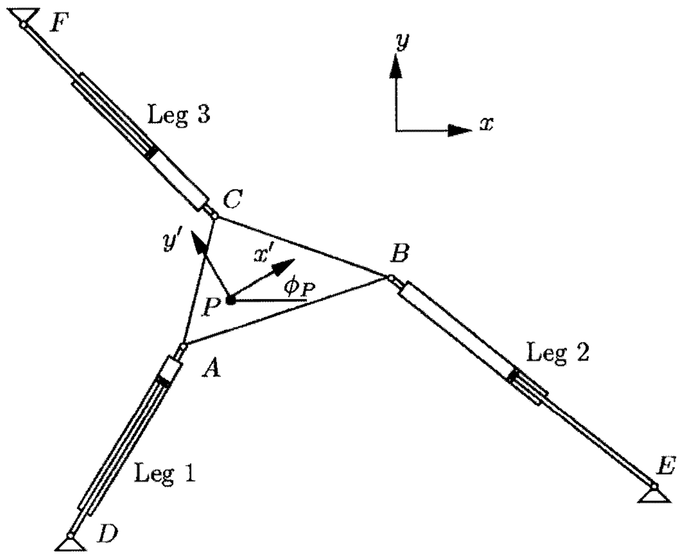

Figure C.1: A general planar parallel manipulator

C.2 A general planar parallel manipulator

Consider the planar parallel manipulator shown in Figure C.1, for which the input coordinates are the leg lengths $\mathbf{v} = [l_1, l_2, l_3]^\mathrm{T}$, the output coordinates are the coordinates of the working point P of the moving platform $\mathbf{u} = [x_P, y_P]^\mathrm{T}$, and the remaining intermediate coordinate is the orientation of the moving platform $w = \phi_P$. Refer to Section 4.2 for definitions of these coordinates.

Physical limits on the input variables (leg lengths) take the form of inequality constraints

$$\mathbf{v}^{\text{min}} \leq \mathbf{v} \leq \mathbf{v}^{\text{max}} \tag{C.1}$$

Similarly, any other physical limits which might be imposed by the manipulator construction (such as limits on the passive joint angles) can be expressed in terms of input, output and intermediate variables by

$$\mathbf{g}^{\text{min}} \leq \mathbf{g}(\mathbf{u}, \mathbf{v}, w) \leq \mathbf{g}^{\text{max}} \tag{C.2}$$

The vectors $\mathbf{v}^{\text{min}}$, $\mathbf{v}^{\text{max}}$, $\mathbf{g}^{\text{min}}$ and $\mathbf{g}^{\text{max}}$ contain the numerical lower and upper limits to the constraints specified in (C.1) and (C.2).

C.3 Maximal workspace definition

The kinematic constraint equations (4.1), ensuring assembly of the mechanism, may be rewritten in terms of the defined coordinates:

$$\mathbf{\Phi}(\mathbf{u}, \mathbf{v}, w) = \mathbf{0} \tag{C.3}$$

The *maximal* workspace W^M of the manipulator, in agreement with the definition given in Section 1.3.1, is defined as

$$\begin{aligned} W^M \quad = \quad & \{\mathbf{u} \in \Re^{nu} : \mathbf{\Phi}(\mathbf{u}, \mathbf{v}, w) = \mathbf{0};\ \mathbf{v} \text{ satisfying (C.1)}; \\ & \mathbf{g}(\mathbf{u}, \mathbf{v}, w) \text{ satisfying (C.2)}\} \end{aligned} \tag{C.4}$$

The boundary ∂W^M of the maximal workspace may then be defined as

$$\begin{aligned} \partial W^M \quad = \quad & \{\mathbf{u} \in \Re^{nu} : \mathbf{u} \in W^M \text{ and } \exists \text{ an } \mathbf{s} \in \Re^{nu} \text{ such that for} \\ & \mathbf{u}' = \mathbf{u} + \lambda\mathbf{s},\ \ \lambda \in \Re \text{ arbitrarily small and either} \\ & \text{positive or negative, no } \mathbf{v} \text{ and } w \text{ exist that satisfy} \\ & \mathbf{\Phi}(\mathbf{u}', \mathbf{v}, w) = \mathbf{0} \text{ as well as inequalities (C.1)} - \text{(C.2)}\} \end{aligned} \tag{C.5}$$

C.4 Finding an initial point of the workspace boundary

As a starting point for the chord method, a suitable radiating point $\mathbf{u}^0$ must be found within the workspace. For a parallel manipulator, the inverse kinematics are easy to solve. Thus, given $\mathbf{u}$ and w, system (C.3) may easily be solved to give $\mathbf{v}$ in terms of $\mathbf{u}$ and w:

$$\mathbf{v} = \mathbf{v}(\mathbf{u}, w) \tag{C.6}$$

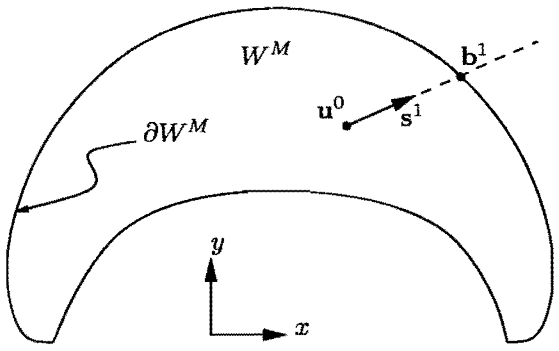

Figure C.2: Finding an initial point on the maximal workspace boundary

Since we are analyzing a planar manipulator, the maximal workspace W^M is two-dimensional. Depending on the particular geometry of the manipulator a suitable choice for a radiating point $\mathbf{u}^0$, inside the workspace, may be self-evident. If not, then $\mathbf{u}^0$ may be obtained from (C.6) by solving for $\mathbf{u}$ and w in:

$$\bar{\mathbf{v}} = \mathbf{v}(\mathbf{u}, w) \tag{C.7}$$

where $\bar{\mathbf{v}} = (\mathbf{v}^{\min} + \mathbf{v}^{\max})/2$.

In practice this can be done by solving the least squares optimization problem

$$\min_{\mathbf{u},w} |\mathbf{v}(\mathbf{u}, w) - \bar{\mathbf{v}}||^2 \tag{C.8}$$

Consistent with the definition of W^M in (C.5), an initial point $\mathbf{b}^1 = (x^1, y^1)^T$ on the boundary, in an arbitrarily chosen direction (designated by a unit vector $\mathbf{s}^1$) from $\mathbf{u}^0$, may be determined by solving the following constrained optimization problem:

$$\begin{gathered}\max_{\mathbf{u},w} \|\mathbf{u} - \mathbf{u}^0\| \\ \text{subject to } \mathbf{v}^{\min} \leq \mathbf{v}(\mathbf{u}, w) \leq \mathbf{v}^{\max}; \\ \mathbf{g}^{\min} \leq \mathbf{g}(\mathbf{u}, \mathbf{v}, w) \leq \mathbf{g}^{\max} \\ \text{and equality constraint } h(\mathbf{u}, \mathbf{s}^1) = \frac{\mathbf{u}-\mathbf{u}^0}{\|\mathbf{u}-\mathbf{u}^0\|} \cdot \mathbf{s}^1 - 1 = 0\end{gathered} \tag{C.9}$$

The solution of Problem (C.9) is illustrated in Figure C.2. Essentially optimization problem (C.9) seeks to find the point of intersection ($\mathbf{b}^1$) of a ray, emanating from $\mathbf{u}^0$ in direction $\mathbf{s}^1$, with the workspace boundary.

C.5 Basic chord methodology

Consider any boundary point $\mathbf{b}^i$ with an associated *unit* vector $\mathbf{s}^{1i}$ pointing *out* of the workspace. A vector $\mathbf{s}^{2i}$ from $\mathbf{b}^i$ to an arbitrary output point $\mathbf{u}$, corresponding to a position of the working point of the manipulator, is

$$\mathbf{s}^{2i} = \mathbf{u} - \mathbf{b}^i \qquad \text{(C.10)}$$

Dropping the superscript i, the angle ω between the unit vector $\mathbf{s}^1$ and vector $\mathbf{s}^2$, defined in the right hand sense, is given by

$$\omega = \begin{cases} \cos^{-1}\left(\frac{\mathbf{s}^1\cdot\mathbf{s}^2}{\|\mathbf{s}^2\|}\right) & \text{if } \alpha \geq 0 \\ 2\pi - \cos^{-1}\left(\frac{\mathbf{s}^1\cdot\mathbf{s}^2}{\|\mathbf{s}^2\|}\right) & \text{if } \alpha < 0 \end{cases} \qquad \text{(C.11)}$$

where $\mathbf{s}^1 \times \mathbf{s}^2 = \alpha\hat{z}$ and $\hat{z}$ is the unit vector in the $z-$direction.

Clearly ω is a function of the output coordinates $\mathbf{u}$.

Given such a point $\mathbf{b}^i$ on the workspace boundary, the next point at a constant chord length d along the workspace boundary may be determined by means of a modified version of optimization problem (C.9):

$$\begin{gathered} \min_{\mathbf{u},w} \omega \\ \text{subject to } \mathbf{v}^{\min} \leq \mathbf{v}(\mathbf{u}, w) \leq \mathbf{v}^{\max}; \\ \mathbf{g}^{\min} \leq \mathbf{g}(\mathbf{u}, \mathbf{v}, w) \leq \mathbf{g}^{\max} \\ \text{and equality constraint } h = \|\mathbf{u} - \mathbf{b}^i\| - d = 0 \end{gathered} \qquad \text{(C.12)}$$

The solution to this optimization problem is depicted in Figure C.3. Having solved problem (C.12), $\mathbf{s}^{2i} = \mathbf{b}^{i+1} - \mathbf{b}^i$ with components s_x^{2i} and s_y^{2i} is precisely known and a *new* reference vector $\mathbf{s}^{2(i+1)}$, associated with the new boundary point $\mathbf{b}^{i+1}$ can now be determined as follows:[1]

[1]The function $\tan 2^{-1}$ has two input arguments and returns an answer in the range $[0, 2\pi]$.

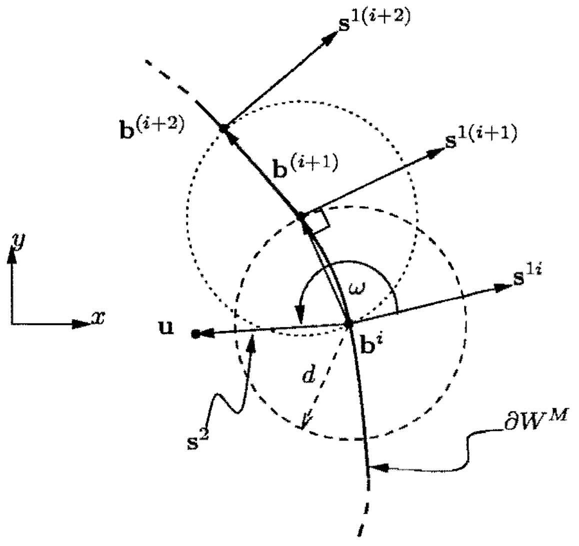

Figure C.3: The chord methodology

$$\mathbf{s}^{1(i+1)} = \begin{bmatrix} \cos\left(\tan 2^{-1}\left(\frac{s_y^{2i}}{s_x^{2i}}\right) - \frac{\pi}{2}\right) \\ \sin\left(\tan 2^{-1}\left(\frac{s_y^{2i}}{s_x^{2i}}\right) - \frac{\pi}{2}\right) \end{bmatrix} \tag{C.13}$$

This equation defines a vector perpendicular to $\mathbf{s}^{2i}$ and pointing out of the workspace. Since it has already been shown how an initial point $\mathbf{b}^1$ and reference vector can be found, it follows that the workspace boundary ∂W^M can be mapped numerically by successively solving optimization problem (C.12) for $i = 1, 2, 3, \ldots$, each time using the solution to the previous problem as the starting point for the new optimization problem. Equation (C.13) is used to determine the associated reference vector for each new boundary point, $\mathbf{b}^{i+1}$.

The algorithm is terminated when a specified maximum number of iterations is exceeded or when

$$\begin{aligned} &\|\mathbf{b}^i - \mathbf{b}^1\| \leq d \\ \text{and } &\|\mathbf{b}^i - \mathbf{b}^2\| \leq d \end{aligned} \tag{C.14}$$

which is an indication of closure of the workspace boundary.

The formulation given above maps the workspace boundary in a counter-clockwise manner. In order to map in the clockwise direction it is necessary to modify the definition of ω in (C.11) to

$$\omega = \begin{cases} \cos^{-1}\left(\frac{\mathbf{s}^1\cdot\mathbf{s}^2}{\|\mathbf{s}^2\|}\right) & \text{if } \alpha \leq 0 \\ 2\pi - \cos^{-1}\left(\frac{\mathbf{s}^1\cdot\mathbf{s}^2}{\|\mathbf{s}^2\|}\right) & \text{if } \alpha > 0 \end{cases} \qquad \text{(C.15)}$$
$$\text{where } \mathbf{s}^1 \times \mathbf{s}^2 = \alpha\hat{z}$$

and to change calculation (C.13) of the reference vector $\mathbf{s}^{1(i+1)}$ to

$$\mathbf{s}^{1(i+1)} = \begin{bmatrix} \cos\left(\tan 2^{-1}\left(\frac{s_y^{2i}}{s_x^{2i}}\right) + \frac{\pi}{2}\right) \\ \sin\left(\tan 2^{-1}\left(\frac{s_y^{2i}}{s_x^{2i}}\right) + \frac{\pi}{2}\right) \end{bmatrix} \qquad \text{(C.16)}$$

The same termination conditions (C.14) apply.

C.6 Determination of bifurcation points

Whenever the manipulator moves along a trajectory such that motion is restricted in some direction, the manipulator is said to be moving along a *bifurcation* path. For maximal workspaces of the planar 3-dof manipulators, such paths correspond to configurations either when two legs remain at extreme lengths while the third varies between extreme values or when one leg is at an extreme length and remains collinear with the working point while the others vary between extreme values. As is to be expected, the *boundary* of the maximal workspace consists of portions of bifurcation paths. The remaining portions of the bifurcation paths are internal bifurcation paths, which correspond to positions within the workspace where the manipulator motion is restricted (Haug *et al.* [32]). Points of intersection of bifurcation paths in the output space usually correspond to bifurcation *points.*

Bifurcation points are positions in the output space where the manipulator can branch or change directly from following one bifurcation path to another.

It is necessary to determine the exact positions of bifurcation points to ensure accurate mapping of the workspace boundary. The precise determination of bifurcation points is done by, having identified the constraints active at the bifurcation point during the mapping procedure, then solving for the point of intersection of these constraints using an optimization approach. Further details of this approach can be found in Snyman *et al.* [48], and Hay and Snyman [49, 50].

Appendix D

Location of the minimum value of κ^{-1} for the two-dof manipulator

THEOREM. *For the 2-dof planar parallel manipulator with $y_A = y_B$, $x_B > x_A$ and $y > y_A$, and any arbitrarily chosen prescribed workspace W_p, the minimum value of κ^{-1} always lies on the boundary ∂W_p of the workspace W_p.*

PROOF. Consider the 2-dof planar parallel manipulator shown in Figure 4.1. The coordinates of point P (output coordinates) are $\mathbf{u} = [x_P, y_P]^\top = [u_1, u_2]^\top$, the input coordinates are $\mathbf{v} = [l_1, l_2]^\top = [v_1, v_2]^\top$. The generalized coordinates are thus $\mathbf{q} = [\mathbf{u}, \mathbf{v}]^\top = [x_P, y_P, l_1, l_2]^\top$.

Dropping the subscript P when referring to x_P and y_P, the actuator leg lengths are given by

$$\begin{aligned} l_1^2 &= (x - x_A)^2 + (y - y_A)^2 \\ l_2^2 &= (x - x_B)^2 + (y - y_B)^2 \end{aligned} \tag{D.1}$$

The kinematic constraints $\mathbf{\Phi}(\mathbf{q})$ are thus

$$\mathbf{\Phi}(\mathbf{q}) = \begin{bmatrix} l_1^2 - (x - x_A)^2 - (y - y_A)^2 \\ l_2^2 - (x - x_B)^2 - (y - y_B)^2 \end{bmatrix} = \mathbf{0} \tag{D.2}$$

Differentiation of (D.2) with respect to time yields

$$\mathbf{\Phi_q}\dot{\mathbf{q}} = \mathbf{0} \tag{D.3}$$

where $\mathbf{\Phi_q}$ contains the partial derivatives of (D.2) with respect to $\mathbf{q}$.

$$\mathbf{\Phi_q} = \left[\begin{array}{cc|cc} -2(x - x_A) & -2(y - y_A) & 2l_1 & 0 \\ -2(x - x_B) & -2(y - y_B) & 0 & 2l_2 \end{array}\right] = [\mathbf{J_u}|\mathbf{J_v}] \tag{D.4}$$

Here the partitioning separates $\mathbf{\Phi_q}$ into entries related to the input coordinates, denoted $\mathbf{J_u}$ and entries related the output coordinates, denoted $\mathbf{J_v}$. Using this partitioning, (D.3) can be manipulated as follows:

$$\begin{aligned} \mathbf{\Phi_q}\dot{\mathbf{q}} = [\mathbf{J_u}|\mathbf{J_v}] \left[\begin{array}{c} \dot{\mathbf{u}} \\ \hline \dot{\mathbf{v}} \end{array}\right] &= \mathbf{J_u}\dot{\mathbf{u}} + \mathbf{J_v}\dot{\mathbf{v}} = \mathbf{0} \\ \mathbf{J_u}\dot{\mathbf{u}} &= -\mathbf{J_v}\dot{\mathbf{v}} \\ \dot{\mathbf{v}} &= -\mathbf{J_v}^{-1}\mathbf{J_u}\dot{\mathbf{u}} = \mathbf{J}\dot{\mathbf{u}} \end{aligned} \tag{D.5}$$

where $\mathbf{J}$ is the Jacobian of the manipulator. Substituting the expressions for $\mathbf{J_u}$ and $\mathbf{J_v}$, contained in (D.4), into (D.5), an explicit expression for the Jacobian of the 2-dof planar manipulator may be obtained.

$$\begin{aligned} \mathbf{J} &= -\mathbf{J_v}^{-1}\mathbf{J_u} \\ &= -\begin{bmatrix} 1/2l_1 & 0 \\ 0 & 1/2l_2 \end{bmatrix} \begin{bmatrix} -2(x - x_A) & -2(y - y_A) \\ -2(x - x_B) & -2(y - y_B) \end{bmatrix} \\ &= \begin{bmatrix} (x - x_A)/l_1 & (y - y_A)/l_1 \\ (x - x_B)/l_2 & (y - y_B)/l_2 \end{bmatrix} \end{aligned} \tag{D.6}$$

The inverse of $\mathbf{J}$ is

$$\mathbf{J}^{-1} = \frac{1}{\det(\mathbf{J})} \begin{bmatrix} (y - y_B)/l_2 & -(y - y_A)/l_1 \\ -(x - x_B)/l_2 & (x - x_A)/l_1 \end{bmatrix} \tag{D.7}$$

where the determinant of **J** is

$$\det(\mathbf{J}) = \frac{(x-x_A)(y-y_B)-(x-x_B)(y-y_A)}{l_1 l_2} \tag{D.8}$$

The condition number κ of the Jacobian is given by $\kappa = \|\mathbf{J}\|\|\mathbf{J}^{-1}\|$ (equation (4.19)), where the norm of **J** is defined as $\|\mathbf{J}\| = [\mathrm{tr}(\mathbf{JWJ}^\top)]^{\frac{1}{2}}$ (equation (4.20)). In order to find the norm of **J**, we first need

$$\begin{aligned}\mathbf{JWJ}^\top &= \begin{bmatrix} \frac{x-x_A}{l_1} & \frac{y-y_A}{l_1} \\ \frac{x-x_B}{l_2} & \frac{y-y_B}{l_2} \end{bmatrix} \begin{bmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \frac{x-x_A}{l_1} & \frac{x-x_B}{l_2} \\ \frac{y-y_A}{l_1} & \frac{y-y_B}{l_2} \end{bmatrix} \\ &= \begin{bmatrix} \frac{(x-x_A)^2+(y-y_A)^2}{2l_1^2} & \cdot \\ \cdot & \frac{(x-x_B)^2+(y-y_B)^2}{2l_2^2} \end{bmatrix}\end{aligned} \tag{D.9}$$

Thus the norm is given by

$$\begin{aligned}\|\mathbf{J}\| &= [\mathrm{tr}(\mathbf{JWJ}^\top)]^{\frac{1}{2}} \\ &= \left[\frac{(x-x_A)^2+(y-y_A)^2}{2l_1^2} + \frac{(x-x_B)^2+(y-y_B)^2}{2l_2^2}\right]^{\frac{1}{2}} \\ &= \left[\frac{l_1^2}{2l_1^2} + \frac{l_2^2}{2l_2^2}\right]^{\frac{1}{2}} \\ &= 1\end{aligned} \tag{D.10}$$

In a similar way it can be shown that

$$\mathbf{J}^{-1}\mathbf{WJ}^{-1\top} = \frac{1}{2(\det(\mathbf{J}))^2}\begin{bmatrix} \frac{y-y_B}{l_2} + \frac{y-y_A}{l_1} & \cdot \\ \cdot & \frac{x-x_B}{l_2} + \frac{x-x_A}{l_1} \end{bmatrix} \tag{D.11}$$

where $\det(\mathbf{J})$ is given by (D.8). The norm of $\mathbf{J}^{-1}$ is thus

$$\begin{aligned}\|\mathbf{J}^{-1}\| &= [\mathrm{tr}(\mathbf{J}^{-1}\mathbf{WJ}^{-1\top})]^{\frac{1}{2}} \\ &= \frac{1}{\det(\mathbf{J})}\end{aligned} \tag{D.12}$$

Substituting (D.10) and (D.12) into (4.19), the condition number κ is given by

$$\begin{aligned}\kappa &= \|\mathbf{J}\|\|\mathbf{J}^{-1}\| \\ &= \frac{1}{\det(\mathbf{J})}\end{aligned} \tag{D.13}$$

The *inverse* condition number is thus

$$\begin{aligned}\kappa^{-1} &= \det(\mathbf{J}) \\ &= \frac{(x-x_A)(y-y_B)-(x-x_B)(y-y_A)}{l_1 l_2}\end{aligned} \tag{D.14}$$

Assuming that $y_B = y_A$, (D.14) can be simplified to

$$\kappa^{-1} = \frac{(y-y_A)(x_B-x_A)}{l_1 l_2} \tag{D.15}$$

The necessary condition for a maxima or minima is $\nabla\kappa^{-1} = 0$. The partial derivative of κ^{-1} with respect to x is

$$\begin{aligned}\frac{\partial\kappa^{-1}}{\partial x} &= -\frac{(y-y_A)(x_B-x_A)}{l_1^3 l_2^3}\left[(x-x_A)l_2^2+(x-x_B)l_1^2\right] \\ &= -\frac{(x_B-x_A)}{l_1^3 l_2^3}(y-y_A)(2x-x_A-x_B) \\ &\quad \times\left[(x-x_A)(x-x_B)+(y-y_A)^2\right]\end{aligned} \tag{D.16}$$

Assuming $(x_B - x_A) > 0$ for all that follows, there are three cases for which $\frac{\partial\kappa^{-1}}{\partial x} = 0$.

Case X1:

$$\begin{gathered} y - y_A = 0 \\ \therefore y = y_A \end{gathered} \tag{D.17}$$

Case X2:

$$\begin{gathered} 2x - x_A - x_B = 0 \\ \therefore x = \tfrac{x_A+x_B}{2} \end{gathered} \tag{D.18}$$

Case X3:

$$\begin{gathered} (x-x_A)(x-x_B)+(y-y_A)^2 = 0 \\ \therefore \left[x - \left(\tfrac{x_A+x_B}{2}\right)\right]^2 + (y-y_A)^2 = \left(\tfrac{x_A-x_B}{2}\right)^2 \end{gathered} \tag{D.19}$$

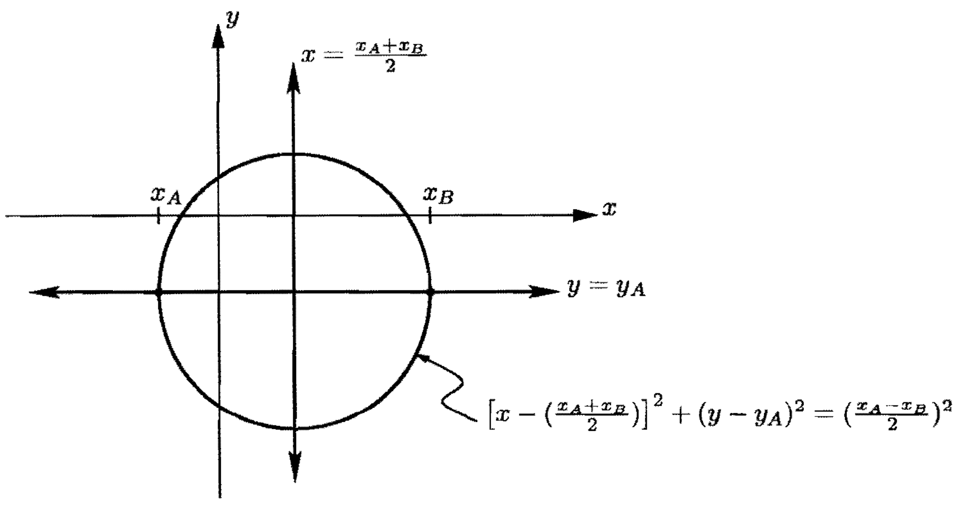
y
$x = \frac{x_A+x_B}{2}$
x_A
x_B
x
$y = y_A$
$[x - (\frac{x_A+x_B}{2})]^2 + (y-y_A)^2 = (\frac{x_A-x_B}{2})^2$

Figure D.1: Solutions to $\frac{\partial\kappa^{-1}}{\partial x} = 0$

These three cases are depicted graphically in Figure D.1.

The partial derivative of κ^{-1} with respect to y is

$$\begin{aligned}\frac{\partial\kappa^{-1}}{\partial y} &= -\frac{(x_B-x_A)}{l_1^3 l_2^3}\left[l_1^2 l_2^2 - (y-y_A)^2 l_2^2 - (y-y_A)^2 l_1^2\right] \\ &= -\frac{(x_B-x_A)}{l_1^3 l_2^3}[(x-x_A)(x-x_B) + (y-y_A)^2] \\ &\quad \times[(x-x_A)(x-x_B) - (y-y_A)^2]\end{aligned} \tag{D.20}$$

There are two cases for which $\frac{\partial\kappa^{-1}}{\partial y} = 0$.

Case Y1:

$$\begin{aligned}(x-x_A)(x-x_B) + (y-y_A)^2 = 0 \\ \therefore \left[x - \left(\tfrac{x_A+x_B}{2}\right)\right]^2 + (y-y_A)^2 = \left(\tfrac{x_A-x_B}{2}\right)^2\end{aligned} \tag{D.21}$$

Case Y2:

$$\begin{aligned}(x-x_A)(x-x_B) - (y-y_A)^2 = 0 \\ \therefore y = \pm\sqrt{(x-x_A)(x-x_B)} + y_A\end{aligned} \tag{D.22}$$

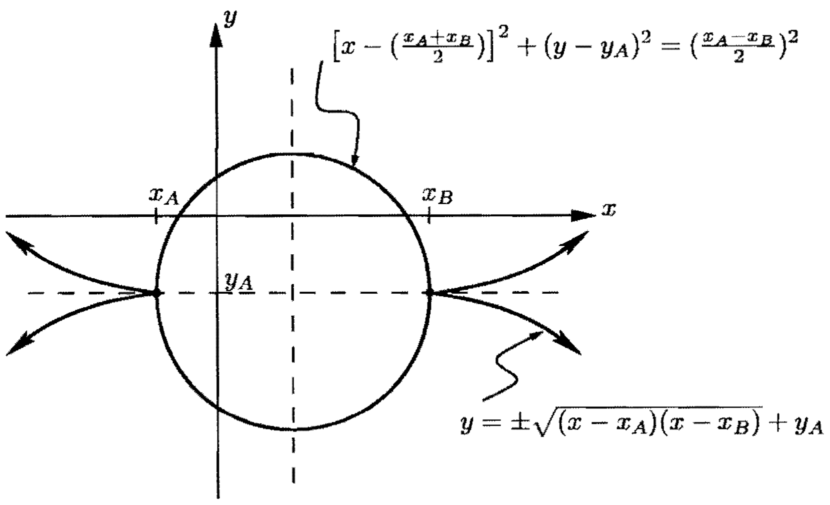
y
$\left[x-\left(\frac{x_A+x_B}{2}\right)\right]^2+(y-y_A)^2=\left(\frac{x_A-x_B}{2}\right)^2$
x_A
x_B
x
y_A
$y=\pm\sqrt{(x-x_A)(x-x_B)}+y_A$

Figure D.2: Solutions to $\frac{\partial\kappa^{-1}}{\partial y}=0$

These two cases are depicted in Figure D.2. Note that for case Y2, real values of y from (D.22) only occur when $x \leq x_A$ or $x \geq x_B$. Comparing the above results it is evident that both necessary conditions are only simultaneously satisfied by the set of points U corresponding to (D.19) and (D.21) from cases X3 and Y1:

$$\mathbf{U}=\left\{\mathbf{u}\in\Re^2:\left[x-\left(\frac{x_A+x_B}{2}\right)\right]^2+(y-y_A)^2=\left(\frac{x_A-x_B}{2}\right)^2\right\} \tag{D.23}$$

which is the set of points on a circle of radius $(x_A-x_B)/2$ centered at $((x_A+x_B)/2, y_A)$.

Transform equation (D.15) to polar coordinates (r,θ) by substituting $y = y_A + r\sin\theta$ and $x = (x_A+x_B)/2 + r\cos\theta$:

$$\kappa^{-1}=\frac{(x_B-x_A)r\sin\theta}{\left[\left(\frac{x_B-x_A}{2}+r\cos\theta\right)^2+r^2\sin^2\theta\right]^{\frac{1}{2}}\left[\left(-\frac{x_B-x_A}{2}+r\cos\theta\right)^2+r^2\sin^2\theta\right]^{\frac{1}{2}}} \tag{D.24}$$

Squaring both sides of equation (D.24) and simplifying yields

$$(\kappa^{-1})^2=\frac{4r^2(\frac{x_B-x_A}{2})^2-4r^2\left(\frac{x_B-x_A}{2}\right)^2\cos^2\theta}{\left[\left(\frac{x_B-x_A}{2}\right)^2+r^2\right]^2-4\left(\frac{x_B-x_A}{2}\right)^2r^2\cos^2\theta} \tag{D.25}$$

It is evident that when $r = (x_B - x_A)/2$, which corresponds to the set of points $\mathbf{U}$, that $(\kappa^{-1})^2 = 1$. It follows that $\kappa^{-1} = \pm 1$. However, $\kappa^{-1} \in [0,1]^\mathsf{T}$, thus $\kappa^{-1} = 1$ and the set $\mathbf{U}$ corresponds to a "maximum ridge".

Assume that $y > y_A$ and that there exists a point $\mathbf{u}'$ *inside* the prescribed workspace W_p such that $\kappa^{-1}(\mathbf{u}')$ is a minima. It follows that $\boldsymbol{\nabla}\kappa^{-1}(\mathbf{u}') = \mathbf{0}$. However it has been shown that the only solution to $\boldsymbol{\nabla}\kappa^{-1}(\mathbf{u}') = \mathbf{0}$ is $\mathbf{U}$, and this set of points corresponds to a maxima. It thus follows that the minimum value of κ^{-1} must lie on the boundary ∂W_p of the workspace W_p.

Appendix E

Minimum norm solution of a set of equations

THEOREM. *On removal of inequality constraints, equation (6.17) is equivalent to (6.14)*

PROOF. In optimization terms, the minimum norm solution of (6.11) may be stated as

$$\min_{\mathbf{t}} \frac{1}{2}\mathbf{t}^{\mathsf{T}}\mathbf{t} \qquad \text{(E.1)}$$
$$\text{such that } \mathbf{St} + \mathbf{f}^{P} = \mathbf{0}$$

where $\mathbf{S}$ is an $m \times n$ matrix with $m < n$. This optimization problem is equivalent to (6.17) with inequality constraints removed.

The Lagrangian $\mathcal{L}$ of this optimization problem is

$$\mathcal{L}(\mathbf{t}, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{t}^{\mathsf{T}}\mathbf{t} + \boldsymbol{\lambda}^{\mathsf{T}}(\mathbf{St} + \mathbf{f}^{P}) \qquad \text{(E.2)}$$

where $\boldsymbol{\lambda} \in \Re^{m}$ denotes the vector of Lagrange multipliers.

The necessary conditions for a minimum are:

$$\nabla_{\mathbf{t}}\mathcal{L} = \mathbf{0} \qquad \text{(E.3)}$$

Applying these to (E.2) yields

$$\mathbf{t} + (\boldsymbol{\lambda}^{\mathsf{T}}\mathbf{S})^{\mathsf{T}} = \mathbf{0} \tag{E.4}$$

$$\therefore \quad \mathbf{t} + \mathbf{S}^{\mathsf{T}}\boldsymbol{\lambda} = \mathbf{0} \tag{E.5}$$

$$\therefore \quad \mathbf{S}\mathbf{t} + \mathbf{S}\mathbf{S}^{\mathsf{T}}\boldsymbol{\lambda} = \mathbf{0} \tag{E.6}$$

$$\therefore \quad \boldsymbol{\lambda} = -(\mathbf{S}\mathbf{S}^{\mathsf{T}})^{-1}\mathbf{S}\mathbf{t} \tag{E.7}$$

By substituting (E.7) into (E.5) it follows that

$$\mathbf{t} - \mathbf{S}^{\mathsf{T}}(\mathbf{S}\mathbf{S}^{\mathsf{T}})^{-1}\underbrace{\mathbf{S}\mathbf{t}}_{-\mathbf{f}^P} = \mathbf{0} \tag{E.8}$$

$$\therefore \quad \mathbf{t} = -\mathbf{S}^{\mathsf{T}}(\mathbf{S}\mathbf{S}^{\mathsf{T}})^{-1}\mathbf{f}^P \tag{E.9}$$

which is the minimum norm solution given in (6.14).