# Multi-Criteria Workspace Optimization of Parallel Kinematic Machines


Author

Sh. M. Muneeb Hamid Rasheed

2010-NUST-MS-PhD-Mts-03

Supervisor

Dr. Kunwar Faraz Ahmed Khan

DEPARTMENT OF MECHATRONICS ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD


SEPTEMBER, 2014

# Multi-Criteria Workspace Optimization of Parallel Kinematic Machines

Author

Sh. M. Muneeb Hamid Rasheed

2010-NUST-MS-PhD-Mts-03

A thesis submitted in partial fulfillment of the requirements for the degree of

MS Mechatronics Engineering

Thesis Supervisor:

Dr. Kunwar Faraz Ahmed Khan

Thesis Supervisor's Signature:__________________________________________

DEPARTMENT OF MECHATRONICS ENGINEERING

COLLEGE OF ELECTRICAL & MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY, ISLAMABAD

SEPTEMBER, 2014

# Declaration

I certify that this research work titled "*Multi-Criteria Workspace Optimization of Parallel Kinematic Machines*" is my own work. The work has not been presented elsewhere for assessment. The material that has been used from other sources it has been properly acknowledged / referred.

Signature of Student

Sh. M. Muneeb Hamid Rasheed

2010-NUST-MS-PhD-Mts-03

## Language Correctness Certificate

This thesis has been read by an English expert and is free of typing, syntax, semantic, grammatical and spelling mistakes. Thesis is also according to the format given by the university.

Signature of Student

Sh. M. Muneeb Hamid Rasheed

2010-NUST-MS-PhD-Mts-03

Signature of Supervisor

# Acknowledgements

I am thankful to my Creator Allah Subhana-hu-Wa ta'ala to have guided me throughout this work at every step. O Allah! I am thankful to you for each new idea which You setup in my attention to develop this work. Certainly I couldn't have done anything without Your incalculable support and direction. Whosoever helped me throughout the course of my thesis, whether my paternities or any further individual was Your will, so certainly none be praiseworthy but You.

I am abundantly gratified to my beloved parents who raised me when I was not capable of walking and persistently supported me throughout in all spheres of my life. I am also copiously thankful to my siblings who always encouraged me whenever I was at my lows.

My special thanks to my supervisor Dr. Kunwar Faraz Ahmed Khan for his help throughout my thesis and also for the AMDT and Motion Planning of Mobile robots courses taught by him.

My exceptional obligations to Dr. Aamer Ahmed Baqai for his remarkable upkeep and assistance. Whenever I got trapped, he helped me out with the clarification. Deprived of his help it would not have been possible for me to complete my thesis. His fortitude and direction throughout the entire thesis is highly appreciable.

My special thanks to Dr. Umer Shahbaz, Dr. Umer Izhar and Dr. Adnan Masood for being on my thesis guidance and evaluation committee.

I would also take this opportunity to thank Mr. Ahad Nazeer, Mr. Hammad Nazeer Gillani and Mr. Saad Shaikh for their help, support, cooperation and guidance whenever it was needed.

Finally, I want to express my thankfulness to all the persons who have rendered prized support for the completion of this work.

*Dedicated to my incomparable parents and gorgeous siblings whose fabulous upkeep and assistance steered me to this delightful triumph*

# Abstract

Parallel kinematic machines or parallel robots have been a topic of research for many researchers in the field of robotics for the last two decades. The core idea is to develop systems that can perform a given task in as desirable a way as possible. The serial robots have been deployed in the industry for quite some time; but the ever increasing requirements on accuracy, controllability and capability to perform tasks in an efficient way; has somehow moved the researchers to find alternate systems. The best alternative to the serial robots so far is in the form of parallel robots. These robots have none of the drawbacks associated with the serial robots hence they become the natural choice to replace their serial counterparts. The parallel robots have their own drawbacks e.g. the workspace is small, irregular shaped and has a lot of singularities. For the industry to use these systems to their full potential there is an immense need of research that could somehow enhance the performance of parallel machines. This project intends to develop a methodology to find the optimum design parameters to get the best results on multiple objectives, hence the title of the project “Multi-Criteria Workspace Optimization of Parallel Kinematic Machines”. These objectives include the workspace volume (size of the workspace) and some other factors that control the quality of the workspace.

**Key Words:** Parallel Kinematic Machines, Workspace Optimization, Multi-Criteria, Design Parameters

## Table of Contents

# List of Figures

# List of Tables

# Chapter 1: Introduction

The preceding few years have beheld a tremendous increase in the usage of robots in the industry to perform certain tasks, mainly due to the fact that they provide great flexibility in process planning and optimization. However the machine-driven structure of most common type of robots does not appear to be perfect in many errands. For this reason further categories of architectures have been considered and used more recurrently in the industrialized world. One class of such architectures is parallel robots. The serial robots have inherent drawbacks that limit their use in the industry.

The serial robots when deployed have a tendency to magnify the error at the actuator by a factor equal to the sum of the link lengths. Moreover the structures of serial robots have to be designed in such a way that every link has to be strong enough to bear the two loads one due to the payload and the other due to the weight of all the consequent links. These design constraints make the structure heavier. This implies that to cover a certain workspace the links of a serial robot have to be heavier than a parallel robot that can cover the same workspace. Whereas the control of serial robots has advanced and reached up to a level that almost any kind of serial robot can be controlled these days.

On the contrary the parallel robots have a better payload to weight ratio, the error amplification is smaller in parallel robots as when equated with the serial equivalents. Whereas the workspace of parallel robots is smaller in size, irregular in shape and has a lot of singularities.

The above discussion shows that both parallel and serial robots have a different set of drawbacks associated with them. The current state of automation achieved in the industry and the future trends dictate that the parallel robots can be a potential alternative to the serial robots if the drawbacks/ their effects can somehow be limited to a certain level.

Workspace optimization is scheme to remove the drawbacks/ reduce their effects. It is a field that deals with the improvement of quality of workspace of a machine. By improving the workspace of a machine we mean that the drawbacks associated with the machine have been reduced or eliminated.

The optimization schemes generally used include weighted sum, modified weighted sum, pareto front optimization, particle swarm and genetic algorithms. The selection of an optimization scheme is generally application dependent however; weighted sum and modified weighted sum techniques fail in the applications of this type. The technique used in this work is pareto front optimization.

## 1.1. Document Organization:

Chapter 2 discusses the background of the research. It includes some definitions, thesis statement and the rationale of the research.

Chapter 3 is the literature review. First of all the state of research is described based upon the literature review carried out. After that a brief analysis of the literature review is presented. Followed by the conclusions of the literature review, objectives of the current research, scope of work, strategic plan, methodology (followed during the research), its justification and the sources of data.

Chapter 4 describes the proposed methodology in detail. First of all the most important objectives are defined. Then the most effective optimization schemes are discussed. Then the design process is explained in detail. Then the methodology developed for the workspace optimization is explained. At the end the validation of results is discussed.

Chapter 5 is the validation of results through an example. First of all the geometric model of the mechanism is presented. Then the design factors to be optimized are listed. It is followed by the ranges of the design parameters. After that the objectives for optimization are listed. It is followed by the requirements for/ constraints on the objectives. Then these constraints are translated briefly to the physics of the mechanism. Followed by iterations, impact/ longevity and results follow.

Chapter 6 is the conclusion it discusses the conclusions of the current research and also describes the future work that can be carried out in this domain.

# Chapter 2: Background

## 2.1. Terminologies and Abbreviations:

Important terminologies and abbreviations used in this research are discussed in the following lines.

### 2.1.1.Parallel Manipulators (PKM) :

A closed loop kinematic chain mechanism with its end-effector connected to the base by numerous independent kinematic chains is known as a PKM [1; 12].

This description of a PKM is too broad and unrestricted. We shall rather use the following definition for parallel manipulators in this study:

A PKM consists of an end-effector with 'n' degrees of freedom, and of an immovable base, to each other by minimum two autonomous kinematic chains. Actuation takes place through 'n' simple actuators [1; 13].

PKMs for which quantity of chains is stringently equivalent to the number of DOF of the end effector are called completely parallel manipulators [2; 457].

### 2.1.2.Inverse Kinematic Jacobian:

Inverse kinematic Jacobian is defined as the matrix relating the end-effector velocity to the actuated joint velocities that defines the velocity linear input-output equations. [1; 154].

The inverse kinematic Jacobian matrix is indispensable for the velocity and course control and planning of robots.

### 2.1.3.Manipulability:

It is defined as the forbearance to haphazardly modify the location and positioning at a given position [3]. This quantity is extremely useful for design and control of robots and mission organization [4, 5]. There exist a number of different formulae in the literature that can be used as a quantitative measure of manipulability. One of the many formulae for manipulability is $M = (J\,J^T)^{1/m}$ where 'm' is the number of DOF of the manipulator [3].

**2.1.4.Condition Number:**

It is the product of the 2nd norm of Jacobian matrix and the 2nd norm of the inverse Jacobian matrix [6].

Another description of condition number is specified in [7]. It states that the condition number is the ratio of the smallest and largest singular values of the Jacobian matrix.

**2.1.5.Isotropy:**

Postures with a conditioning index one are known as isotropic poses [1; 169]. It is generally desirable that all the poses in the workspace of a manipulator should be isotropic. Planning a parallel robot that is isotropic in one posture or is isotropic over its thorough workspace is sometimes deliberated to be the design objective.

**2.1.6.Global Conditioning Index:**

The global conditioning index is defined as the integral of reciprocal of the conditioning index over the whole workspace [6].

**2.1.7.Uniformity:**

It is the ratio of smallest and largest value of manipulability i.e. $U = M_{min}/M_{max}$. Its value is always positive and less than or equal to unity and is preferred to be as nearby to unity as conceivable [3].

**2.1.8.Global Manipulability:**

It is the summation of the manipulability over the entire workspace [3].

**2.1.9.Error Amplification Factor:**

As the name suggests it is the factor by which the error on actuator is amplified at the end effector.

**2.1.10. Singular Configurations:**

These are the configurations of the robot for which the robot loses its characteristic inestimable stiffness and the end effector will have uncontainable DOF [1; 179]

### 2.1.11. Effective Regular Workspace:

It is a regular shaped workspace with good dexterity [7].

## 2.2. Synopsis/ Thesis statement:

The field of parallel robots has been explored intensively during the past few years but the focus of the research has been on new designs rather than improving the performance of the existing systems to fit into the current industrial environment. All the work so far carried on in the domain of workspace optimization of parallel robots focuses on the workspace optimization subject to a single objective at a time. The basic idea of this work is to propose a procedure for the optimization of workspace subject to multiple criteria simultaneously. These objectives/ criteria often have conflicting requirements which is a serious challenge in this domain. This will be beneficial for the researchers around the globe since the use of these systems is not limited to conventional systems but a new horizon for these systems is the field of surgical robots.

## 2.3. Rationale:

The aim of this work is to cultivate a procedure for the multi-criteria optimization of workspace. As stated earlier there are some inherent drawbacks associated with the serial robots; parallel robots on the other hand overcome the drawbacks associated with the serial robots but have their own drawbacks. An example of such a drawback is the shape of workspace which is generally not of a regular shape. Another drawback is that there exist parallel singularities in the workspace and there are some others as well.

It is evident from the above discussion that it is more desirable to use a parallel manipulator instead of a serial one as the complexity of the application increases. To increase the use of parallel manipulators or to replace the serial manipulators with their parallel counterparts it is of utmost importance that methodologies be developed to overcome the drawbacks associated with the parallel manipulators. These methodologies should optimize the workspace in such a way that all the desired factors of the workspace be in a desirable range.

# Chapter 3: State of the Art and Research Methodology

## 3.1. Literature Review:

Parallel kinematic machines (PKMs) are acknowledged for their great dynamic performances and little positioning errors. However parallel singularities occur in the workspace where the end-effector cannot resist any exertion; and consequently are too detrimental. These are normally eradicated during the project. The performance indices like maximum speed, force, accuracy and stiffness etc. differ significantly for all points in the workspace and for all directions at a given point for the reason that the Jacobian matrix is not constant and not isotropic. This is a severe downside for machining applications. Few parallel machines are isotropic all the way through their workspace. Conversely their little mechanical rigorousness makes them insufficient for machining applications since their links are subject to twisting. To be of concern for machining applications a PKM should have good workspace properties, that is, Consistent workspace form and tolerable kinetostatic performances all the way through. These kinetostatic performances may include a number of factors like manipulability, dexterity, condition number, stiffness, force transmission factor, velocity transmission factor, symmetry, workspace volume, isotropy etc. Let's say, in milling machines, the machining conditions must remain continuous end to end in the entire tool route.

The customary parallel robots have ascertained their rewards in facets of stiffness, rigidity, dexterity, re-configurability, with the widespread application in machine tools [9-11], motion simulators [12], picking and placing, sensors [13,14]. Parallel platforms are presently being used in many applications as multi DOF systems with large stiffness, large payload to weight ratio, large precision and small inertia [15, 16].These are also the desired features of the joint modules of re-configurable robots. Six legged; six DOF parallel machines have been employed as joint modules of the re-configurable machines in [17]. Due to these properties the PKMs are extensively acknowledged as perfect machines in engineering industries. However inadequate workspace, intricate input-output relationships and richness of singularities in the workspace have deteriorated the parts of above mentioned returns.

The beauty of PKMs is that it is possible for the mechanism to be designed in such a way that the moving structure does not have to bare the weight of the actuators driving it. This facilitates large powerful actuators to drive somewhat smaller loads. This enables the designer to design a PKM that is far better than its serial counterparts in terms of speed, stiffness and strength. The optimization of PKM's workspace volume depends upon a means of shaping the workspace of a parallel manipulator for a specified set of design variables.

Amid all kinematic properties, workspace is the rudimentary and the most imperative index in design of a parallel manipulator. There are two types of workspace optimization a). One is to produce a manipulator whose workspace holds a given space [18], [19], [20], [21]. b). the other conceivable devising is to catch the geometric parameters of a manipulator that maximize the workspace. A design whose lone purpose is to maximize the workspace is not recommended since there are other workspace properties that effect the performance of the manipulator that need to be taken into account otherwise a maximized workspace with poor workspace properties might not be as useful as a small workspace but with good workspace properties. In most of the practical applications a manipulator with regular shape and good dexterity is more desirable than a manipulator whose workspace has been maximized with poor workspace properties.

Generally there are several performance criteria that a design should meet. However, most of the researchers have considered only two of the basic factors i.e. workspace and condition number. Some literatures have accompanied the design process using one or two other factors as well. Few design literatures have considered numerous measures. But it is an essential to deliberate several measures in design for specific applications. For example in case of a machine tool not only necessitates a great workspace and good condition number, but also decent accuracy, extraordinary stiffness, speed and great force etc.

Parallel manipulators have smaller workspaces relative to the serial manipulators of the same degree of freedom; consequently numerous investigators addressed the workspace optimization of parallel robots [18] [22- 23]. However optimization for such an objective may lead to a manipulator with a workspace that has poor kinetostatic performance measures. To lessen this problem some researchers deliberated on both performance indices and workspace volume instantaneously [24, 25].

In most of the applications we are interested in dexterous workspace rather than the accessible workspace. The collection of points that its end-effector can reach makes the workspace of the manipulator. Kinetostatic performance indices or dexterity measures how well the structure performs in regards to force and motion transportation.

The above paragraphs show the importance that workspace optimization carries and the complexity involved in the process. A general requirement is to have a workspace equivalent to or greater than the serial counterpart but with better kinetostatic indices e.g. stiffness, manipulability, condition number, force transmission factor, velocity transmission factor, high precision, low inertia, isotropy, uniformity, error amplification factor and many more.

The researchers so far have not been able to or have not yet felt the need of defining the core properties that need to be optimized irrespective of the application where the manipulator is to be used and the type of parallel manipulator.

This study is focused on proposing a generic optimization methodology that is applicable on every type of parallel manipulator and for every type of application. In the coming paragraphs we briefly present the work that has so far been done by the researchers around the world. This section will form the bases of the research.

In [7] a study was carried out on this topic. They defined the problem as maximizing the volume of so called maximum effective regular workspace. A regular-shaped workspace having good dexterity is wanted continually. They proposed that for a manipulator that can translate as well as rotate the workspace volume is given as the weighted sum of both translational and rotational workspaces. If we define α to be the set of kinematic parameters of interest then we can say that for permanent ranges of actuators of a manipulator, its workspace volume always increases with increase in its complete dimensions i.e. 'α'. These dimensions are also under certain constraints.

A regular workspace ought to be confined in the total workspace of the manipulator. For each point 'X' in the workspace we need to find its inverse kinematic solution. This should lie in the actuation range of the corresponding actuator. For a point in regular workspace, if there occurs an inverse kinematic solution in the actuator range, the point is accessible.

In order to make sure that the regular workspace generated is effective, constraints are introduced on the dexterity index. The most frequently used dexterity index is the conditioning index of the Jacobian matrix. It is the ration between the smallest and largest singular values of the Jacobian matrix. This value lies in the range [0, 1]. If the manipulator has both translational and rotational DOF then the components of Jacobian bear different units. A design based on this Jacobian will not be reliable. This was first reported by Lipkin and Duffy in [26]. The most commonly used approach for this problem is the introduction of characteristic length [27]. Jacobian is at that time normalized by dividing a characteristic length out of all translational elements. In [28] scholars defined the natural length as the characteristic length that is capable of producing the unsurpassed performance degree and applied this notion in design optimization. With the modified Jacobian the dexterity requirement is stated as "the condition number should be greater than or equal to a constant specified by the user".

The optimization problem is thus identified as to maximize the regular workspace volume on the bases of 'α' the design parameters such that every point included in the workspace should have a conditioning index in the given range and the inverse kinematic solution for every point should also lie in the actuation ranges.

In [29] a research was carried out with three aims

i. The workspace covered by the manipulator encloses a given workspace.

ii. The manipulator owns good conditioning index at every point in the given workspace.

iii. The manipulator owns decent performance on performance indices like accuracy, stiffness, velocity/force transmission factor.

First the workspace is discretized and for every point in the workspace the inverse kinematic solution is found, if it happens to be existent, real and is in the actuation range then this point will certainly be in the workspace produced by the consequential manipulator. They formulated a pair of quadratic inequalities that were used for the optimization of workspace.

A usual requirement on the condition number is that the manipulator should be restricted to be far-off from the singularity manifold or even to be in the vicinity of isotropic configurations. It means that we want the conditioning index of the Jacobian matrix to be less than a given number. The Jacobian is then split into the forward and inverse Jacobian matrices and the condition restated, the conditioning index of the forward Jacobian should be less than a given number and same goes for the inverse Jacobian matrix. The conditioning index of the forward Jacobian matrix restricts the manipulator to be far-off from the forward singularity manifold and to be in the vicinity of forward isotropic configurations. The conditioning index of the inverse Jacobian matrix restricts the manipulator to be isolated from the inverse singularity manifold and to be in the region of inverse isotropic configurations.

Moreover they derived some conditions that would monitor the accuracy, stiffness. Velocity transmission factor and force transmission factor. For accuracy the condition is "maximize the maximum singular value of inverse Jacobian matrix for every point in the workspace and then select the minimum of these values" or equivalently "curtail the minimum singular value of Jacobian matrix for every point in the workspace and then select the maximum of these values".

In real implementation a loose requirement serves the purpose of forcing all the maximum singular values of inverse Jacobian to be smaller than a given bound so that the manipulator possesses a required precision.

For velocity transmission factor the condition is "curtail the minimum singular value of inverse Jacobian matrix for every point in the workspace and then select the maximum of these values" or equivalently "maximize the maximum singular value of Jacobian matrix for every point in the workspace and then select the minimum of these values".

The extreme speed which should be achieved at any point of time throughout the operation of the manipulator is given as a design condition. This necessitates that a set of design factors should fulfil the condition that all the smallest singular values of the inverse Jacobian matrix are greater than a particular bound. Where the bounding value is dictated by the conditions on the velocity to be achieved.

For force transmission factor the condition is "curtail the minimum singular value of Jacobian matrix for every point in the workspace and then select the maximum of these values"

In physical employment free condition is specified so that the manipulator produces a given amount of supreme force at all points in the given workspace. This necessitates that a set of design considerations ought to fulfil the condition that all the smallest singular values of the Jacobian matrix are greater than a given bound. Where the bounding value is dictated by the conditions on the force to be transferred.

For stiffness the condition is to minimize the minimum eigenvalue of the $J^TJ$ in the prescribed workspace and then select the maximum of these values.

It is to be noted that eigenvalue of the $J^TJ$ is equivalent to the singular value of the $J^2$ matrix. This holds for both the largest and smallest values of both the eigenvalues of $J^TJ$ and the singular values of $J^2$.

This means that the condition on stiffness can be restated as “to minimize the minimum singular value of the Jacobian matrix for every point in the workspace and choose the maximum of these values.

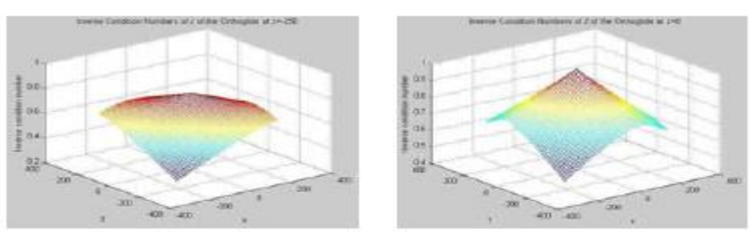
This necessitates that a collection of design parameters should fulfil the condition that all the lowest singular values of the Jacobian matrix are greater than a given bound. Where the bounding value is dictated by the conditions on the stiffness to be achieved.

The conditions on singular values of Jacobian matrix are conflicting with each other so a tradeoff has to be made. An appropriate collection of design parameters ‘α’ ought to satisfy the condition that all the minimum singular values of the Jacobian matrix need to be greater than a certain bounding value and all the maximum singular values of the Jacobian matrix need to be less than a certain bounding value. These bounds are defined by the conditions imposed on the singular values by the individual objectives.

The above condition can be stated as for a certain parameter set ‘α’ and for some point in the given workspace all singular values are in the range of $S_1$ and $S_2$; where $S_1$ and $S_2$ are the bounding values for the singular values.

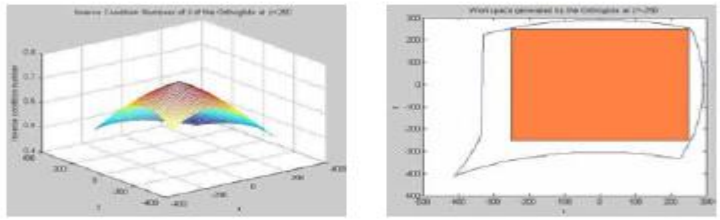
The optimization problem is thus stated as minimizing the objective function subject to the following conditions:

- The inverse kinematic solution of each point lies in the actuation range.
- The condition number for every point is less than a certain number.
- The singular values of the Jacobian matrix at every point lie in the range of s1 and s2.



(a) (b)

*Figure 3-1: (a) Inverse condition Number of J at z=-250; (b) Inverse condition number of J at z=0.*



(a) (b)

*Figure 3-2: (a) Inverse condition Number of J at z=250; (b) Workspace generated by the orthoglide at z=-250.*

In the research carried out in [30] a study on the multi criteria optimization of a 3DOF RAF parallel manipulator was performed. A multi objective function centered on the mathematical notion of power of a point with refrencet to surface is framed. The recommended technique is modest and operational in examining the design vector defining the robot with the minimum workspace and covering a given volume in space.

The paper focuses at designing the RAF parallel robot by calculating the workspace and optimizing the design vector comprising a given volume in space.

The workspace of the RAF robot consists of two parts namely active and passive workspaces. The workspace of the robot is defined as the connection of the active and passive workspaces.

Define the workspace constraints for the active workspace and passive workspace and these can be derived using the kinematics of the robot. Subject to the constraints on the workspace we need to curtail the objective function F (I, P). Where P is any point under consideration and I is the unidentified vector of parameters these parameters are the link lengths.

Methodology adopted here is to minimize the summation of the powers of the vertices which will depend on I.

They first obtain a design that contains all the possible points $P_k$ and then we move on to minimize the sum of powers of the points. There are weighting factors associated with both active and passive workspaces. Changing these factors changes the optimal solution.

Workspace volume is calculated using the algorithm proposed in [31].

In [32] a research was carried out on this subject and they optimized the workspace of a 3dof translational stand for well-conditioned workspace. In this investigation they carried out two optimization studies. The objective to begin with is to maximize the aggregate volume of the manipulator workspace irrespective of the superiority of the workspace. In the 2nd revision they optimized the aggregate volume of well-conditioned workspace by maximizing a global conditioning number. The global conditioning number is a means of measuring the error amplification between the actuators and the end effector. Optimization of manipulator workspace volume is reliant on a method of defining the workspace for a given set of design variables. They computed a statistical value of the workspace size (Volume) by means of the monte-carlo method for the purpose of workspace optimization. The manipulator used in this revision was a 3-DOF translational machine. The design variables deliberated upon were the leg lengths, relative size of the base and moveable platform and the angular position of the legs 2 and 3 from the first leg that is considered to be at home position. The problem was then constrained using constraints on the link length and angular separation between legs. These constraints are

- The total link length is not to exceed 1.
- Every leg should have an angular parting of no less than 5 degrees from other legs.
- Leg lengths cannot be equal to or less than zero.

After the devising of the problem it was optimized using the MATLAB optimization tool box.

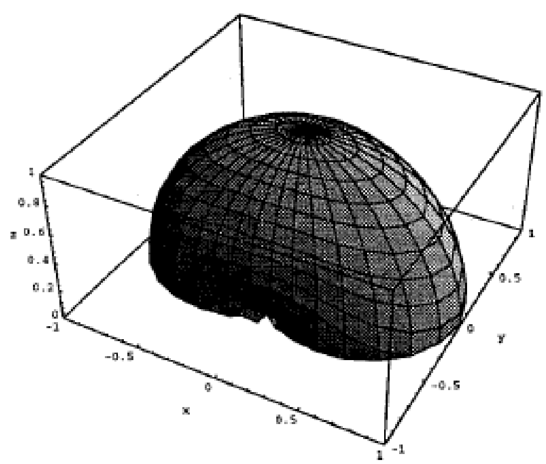



*Figure 3-3: Workspace of manipulator for maximum workspace*

The second study optimized the global conditioning index of the manipulator. The defined the condition number to be the product of 2nd norm of Jacobian matrix and the 2nd norm of inverse of Jacobian matrix. Global condition index was defined as the integral of the inverse of the conditioning index over the complete workspace. To find this value again monte-carlo technique is used and a algebraic expression for the global conditioning number is found. And this expression was then optimized to find out the optimum workspace. The workspace produced by the manipulator in such condition is shown below:

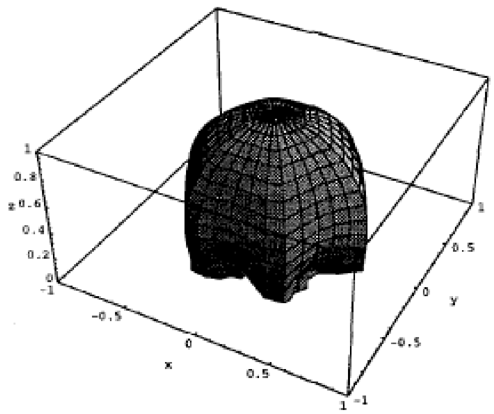



*Figure 3-4: Workspace of manipulator for maximum global condition index*

The comparison can be seen if we compare the curve of global condition index at a given level for both cases. Figure 3-5 shows the curve of global condition index at z = 0.5 plane when only the workspace volume is maximized.

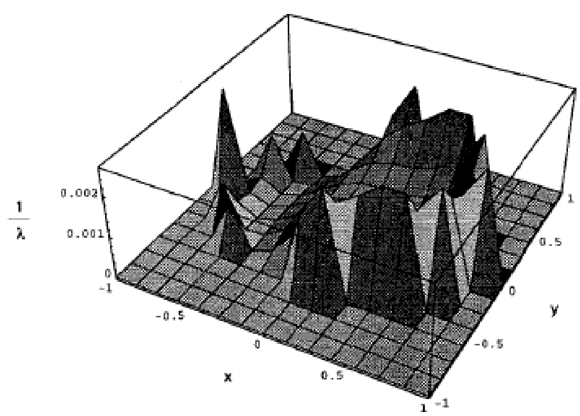



*Figure 3-5: Reciprocal of the condition number at z=0.5 plane for the total workspace optimized manipulator*

Figure 3-6 shows the same for the case when the workspace is optimized considering the global conditioning index.

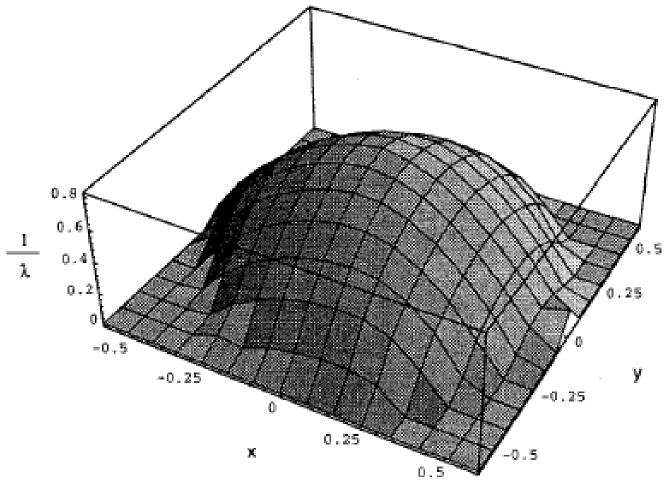



*Figure 3-6: Reciprocal of the condition number at the z=0.5 plane for the global condition index optimized manipulator*

The results of their research showed that a manipulator premeditated to optimize entire workspace volume is considerably dissimilar from the one optimized for a well-conditioned workspace. Furthermore the outcomes showed that a manipulator intended to optimize entire workspace volume results in an ill conditioned workspace. The above comparison shows that it is better to maximize the workspace volume by maximizing the global condition index instead of only workspace volume. This will insure that the performance of the manipulator in the workspace is up to a certain standard.

In [33] the researchers used Particle swarm algorithm for the optimization of a parallel mechanism. The workspace is generated and recorded on basic boundary searching technique. The particle swarm procedure is applied to hunt for the optimum volume of workspace. Basic workspace demonstration and optimization methodologies are established for a fresh parallel mechanism. The suggested techniques are universal and appropriate for visual analysis, modeling and optimization of workspace for the diverse categories of parallel manipulators. Workspace of a parallel mechanism is coarsely distributed in job workspace and joint workspace. The job workspace mentions the motion choices of the moving platform in 2 or 3 dimensions. The entire area was computed to define the performance of a 3D job workspace. After generation of workspace by simplified boundary search method, the geometric parameters of the mechanism are optimized using the particle swarm optimization to maximize the workspace volume.

In [21] the design problem was formulated as to design a parallel mechanism in a way that its workspace encompasses a given workspace with good conditioning number in it.

In [34] the culling algorithm was applied to optimize the global isotropy index (GII) that happens to be the fraction amid the smallest and largest singular values of the kinematic Jacobian matrix in the workspace. The objective function was designed and optimized using the MAX-DET problem. It was subjected to the constraints that the conditioning index of the Jacobian matrix at all points in the workspace is less than $(1 + \Upsilon)$ where $\Upsilon$ is a threshold specified by the user and that the inverse kinematic solution of every point should lie in the actuation range of the actuators.

In [6] the researchers optimized the workspace of a parallel robot by considering the maximum inscribed workspace and reciprocal of the conditioning index of the workspace. The golden search method is used to quest the workspace of the manipulator and mesh the boundary. Optimization considering the maximum inscribed workspace we find the relationship between leg lengths and the geometric parameters and these were optimized. The conditioning index is the product of 2nd norm of the Jacobian matrix and that of the inverse Jacobian matrix. The global conditioning index that deliberates the conditioning index of the Jacobian matrix over the whole workspace is the integral of the conditioning index over the entire workspace. The need for the conditioning index optimization ascends due to the fact that the workspace optimization by maximizing the workspace volume will probably result in poor dexterity and other kinematic characteristics.

In [3] the researchers worked on the optimum creation for workspace and manipulability of parallel flexure mechanism. They started off with the Yoshikawa's manipulability index that is given as the 2nd root of the determinant of $JJ^T$ matrix.

But this index is both scale and order dependent, to remove the order dependencies Kim and Khosla replaced the square root of the formula by the $m^{th}$ root and by doing this they removed the order dependency. Later on the scale dependency was removed by dividing the order independent manipulability with the square of a basic dimension L [35].

Global assessment of manipulability was done by integrating the manipulability over the whole workspace and dividing it with the workspace volume. The other important parameter was the uniformity of manipulability. There are two draw backs that a good uniformity index can remove

- Non-precise motion of platform
- The flexure mechanism cannot move due to high force transmission

The optimization problem was stated as maximizing the workspace area subject to a constraint on global manipulability and uniformity of manipulability. The optimization variables are the dimensions of the Virtual Rigid body model, range of actuators and the initial pose.

In [36] the researcher worked on the collision-lass workspace design and optimization of the 3-DOF gantry-tau PKM. He optimized the workspace for the collision free operating region and found that the by this methodology the total workspace to fixing space ratio is $V_{Installation} / V_{Fixing}$= 3.5228 $m^3$. This ratio is huge equated to most of the other PKMs which characteristically have a ratio of less than one.

In [37] the researchers worked on the dexterous workspace optimization of a tricept PKM. The higher and lower bounds of actuators, spherical and universal joints, link lengths and platform radii are subjected to constraints for this problem. The optimization was performed considering the dexterity measures, viz. the conditioning index as a confined conditioning index and smallest singular values. Variables to be optimized are the radii of the moving and base platforms and the higher part of the middle link length.

In [38] the researchers proposed the optimization of parallel manipulators on the bases of global stiffness using kinetostatic indices. It is proposed that the average value and the standard deviation of the trace of the generalized compliance matrix may be used as the design index. It is eminent that the trace of the compliance matrix is same as the matrix, so the dissemination of the system stiffness/compliance is the dissemination of the trace. In this context the average value and the standard deviation of the trace of the compliance matrix can be understood. Generally a lower mean value indicates less distortion. Likewise an inferior standard deviation means a more unchanging stiffness distribution above the workspace.

In [39] the researchers proposed a unique set of optimization parameters. They defined two objective functions. First of the two objective functions was the moving mass that had to be minimized. The other being the regular shaped workspace that needs to be maximized. Moreover they calculated the condition number by using the Frobenius norm instead of 2-norm. There were three types of constraints in this problem, the geometric constraints, condition number and accuracy.

In [8] the researchers presented the concept of error amplification factor. They first of all formulated a total error transformation matrix for the hexa slide mechanism. They noted that three types of error amplification factors can be defined on the bases of total error transformation matrix and either of the three can be used for the optimization problem.

Global error amplification factor was then defined as the integral of error amplification factor over the complete workspace divided by the workspace volume. The optimization problem was thus stated as to minimize the global error amplification factor subject to the design variable limits and workspace constraints. A total of 60 design parameters existed but the problem was simplified by exploiting the symmetry property of the hexa slide mechanism and the number of design variables is thus reduced to six.

In [40] the researchers carried out an optimization of a 3DOF parallel manipulator. They optimized the manipulator on considering the shape of the workspace and the objective was to achieve a regular shaped workspace. Geometric constraints were introduced on the mechanism and interval analysis method was used for optimization.

In [41] the researchers worked on the workspace optimization of 3-legged universal prismatic universal (UPU) and universal prismatic spherical (UPS) parallel platforms with constraints on the mobility of joints. They used 3 indices to characterize the workspace of the system. First one is the workspace volume, $2^{nd}$ average of inverse of conditioning index and $3^{rd}$ global conditioning number as used in [42]. It is defined as the ratio of summation of inverse conditioning index computed in the entire workspace, to the volume of the workspace. The condition index used in this research is the product of 2nd and 3rd index.

Generally researchers have used genetic algorithms, interval analysis, max-det and few other numerical methods for the subject optimization. The choice of algorithms is totally dependent on the application and the discretion of the researcher.

## 3.2. Research Analysis:

Researches on the subject have indicated that the most important factors that should be considered in the workspace optimization are conditioning number, error amplification factor, singular values of the Jacobian matrix, the mean and standard deviation of trace of the compliance matrix, the moving mass, shape of the workspace, isotropy index, uniformity index and manipulability.

The choice of parameters is application dependent and needs to be decided by considering the application area, the task to be performed and the critical factors that can affect the performance of the manipulator in the given application.

The choice of algorithm is also dependent on the application of the manipulator and its type. Generally genetic algorithms are preferred because of their ability to easily avoid local extrema.

Another important thing in this type of problems is the thresholds that are applied on different factors, these thresholds need to be decided carefully otherwise the results can be drastically misleading and the manipulators thus generated most unreliable and unsuitable for the application.

## 3.3. Conclusion:

Conclusions drawn from the detailed analysis of the literature review are summarized here. The methodologies generally include some analytic models and genetic algorithms. Simultaneous optimization of multiple criteria has not yet been carried out in the field of parallel machines. The most frequently optimized objectives are workspace volume, shape, global conditioning number, force transmission factor, velocity transmission factor and accuracy. Literature shows that these objectives often require opposite conditions i.e. optimizing one objective worsens the performance of the others. Moreover it is concluded that most of the objectives are dependent on the Jacobian and inverse Jacobian. Most of the times to optimize an objective the maximum and minimum singular values of inverse and forward kinematic Jacobians are used. Workspace volume and shape of workspace however do not depend on the Jacobian and its inverse.

## 3.4. Objectives:

This study is related to the workspace optimization of PKMs. The aim of this study is to propose a methodology to optimize the workspace of parallel manipulators considering multiple criteria. The researches till now do not focus on optimizing the workspace by considering all the factors simultaneously. In this work a procedure to optimize the workspace of parallel manipulators by considering multiple criteria simultaneously. The research is divided following sub tasks sections:

- Identifying the objectives to be optimized
- Mathematical interpretation of the objectives
- Devising a methodology
- Applying a suitable optimization technique
- Verifying/ validating the results

## 3.5. Scope:

Scope of this project includes:

- Devising a methodology for workspace optimization
- Validation by comparison with published results

## 3.6. Strategic plan:

This project is divided into different sub-tasks including following steps with sequence

- Literature Review
- Understanding the techniques adopted so far
- Understanding different objectives
- Mathematical and physical interpretation of these objectives
- Finding the optimization objectives
- Devising an optimization methodology
- Applying the methodology on an existing system

- Comparison of results with published results

## 3.7. Methodology:

The methodology adopted for this research can be briefly explained by the following steps:

- Literature Review
- Summarizing the literature review in the form of a literature review report
- Reproducing the work done in some researches
- Defining problem statement
- Developing a methodology for workspace optimization
- Benchmarking a research for the validation of results
- Generating results
- Comparison with the benchmark results

First of all an extensive literature review was carried out. About 50-60 research papers on different techniques and methods adopted in the field were studied. The work done in the relevant fields was then summarized in the form of a literature review report. This report was helpful since it provided a summary of the current state of the art and helped in setting the goals for the current research.

After this report results of some researches were reproduced. It was done in order to gain some insight to the techniques being used and the field of parallel manipulators. Moreover this helped to get a head start in the field.

The next step was defining the problem statement and setting up the objectives of the project. This was done after a lot of deliberation on the literature review report and the reproduction of results of different researches.

Next step was to develop a methodology for the simultaneous multi-criteria workspace optimization of PKMs. The methodology focused on achieving the goals accurately and in a less complex manner.

A research was benchmarked so that the results generated by our research could be validated against some already published work in the said field. In the next step the methodology was implemented on a mechanism and results were generated. These results were then validated against the results of the benchmark research.

## 3.8. How is it beneficial?

The methodology is beneficial since it was a stepwise scheme. The difficulty level was raised slightly and according to the need. Since there hasn't been a lot of work in this domain it was kept in mind that the validation of results was going to be a tough ask hence special consideration was given to this aspect and a novel approach was adopted to validate the results. It was helpful since before getting in the actual work the researcher was fully aware of the current state of the art and the domain itself. It is beneficial since unlike the existing state of the art the methodology focuses on the simultaneous multi-objective workspace optimization of PKMs. Moreover the choice of optimization algorithm is also important since there are a number of algorithms that are used for the multi-objective optimization. These include algorithms ranging from as simple as weighted sum to as complex as genetic algorithms.

## 3.9. Resource/ Source of data:

The resource of data has mostly been the literature reviewed. Since there has not been a significant work in the field of multi objective workspace optimization of PKMs, majority of the data has been extracted from the literature as concepts. No major values or techniques have been found from the literature. Primarily the source of data has been the research papers studied.

# Chapter 4: Proposed Methodology

The quality of workspace of PKMs depends on the objectives that are used to optimize the workspace. Each factor has its own significance and has a considerable effect on the quality of the workspace generated by the manipulator. But important fact is that most of the applications will require the optimization with respect to more than one objectives.

Off course we cannot expect that all the objectives have the same weightages since every application demands different constraints on different objectives. In order to generate correct results we have to keep this fact in mind.

## 4.1. Most effective objectives:

The criterion to judge most effective objective is not simple. The effectiveness of an objective is dictated by the application where the manipulator has to be used. But most of the applications can be handled if suitable constraints are imposed on the following factors:

- Workspace Volume
- Global Condition Number
- Accuracy
- Velocity Transmission Factor
- Force Transmission Factor
- Inverse Condition Number

Handling these objectives effectively and optimizing them generally results in the optimization of manipulator for most of the applications. However there are some other objectives that mau need to be optimized for some applications.

## 4.2. Most effective optimization schemes:

The most effective optimization scheme could have been the weighted sums in this kind of multi criteria problems. The problem here is that the weights of the objectives are not fixed rather they are highly dependent on the application where the manipulator has to be used. So the weights sum optimization cannot be used.

Next we considered the evolutionary algorithms in general and genetic algorithms in particular. A genetic algorithm (or GA) is a search technique used in computing to find true or approximate solutions to optimization and search problems. They are categorized as global search heuristics. This is a particular class of evolutionary algorithms that uses techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination).

Genetic algorithms are implemented as a computer simulation in which a population of abstract representations (called chromosomes or the genotype or the genome) of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem evolves toward better solutions.

The evolution usually starts from a population of randomly generated individuals and happens in generations.

In each generation, the fitness of every individual in the population is evaluated, multiple individuals are selected from the current population (based on their fitness), and modified (recombined and possibly mutated) to form a new population.

A typical genetic algorithm requires two things to be defined:

- A genetic representation of the solution domain
- A fitness function to evaluate the solution domain

The most common type of genetic algorithm works like this:

A population is created with a group of individuals created randomly. The individuals in the population are then evaluated against a fitness function. The evaluation or fitness function is provided by the programmer and gives the individuals a score based on how well they perform at the given task. Two individuals are then selected based on their fitness, the higher the fitness, the higher the chance of being selected. These individuals then "reproduce" to create one or more offspring, after which the offspring are mutated randomly. This continues until a suitable solution has been found or a certain number of generations have passed, depending on the needs of the programmer.

Initially many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Traditionally, the population is generated randomly, covering the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as this process may be very time-consuming. Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This helps keep the diversity of the population large, preventing premature convergence on poor solutions.

The next step is to generate a second generation population of solutions from those selected through genetic operators; crossover (also called recombination), and/or mutation. For each new solution to be produced, a pair of "parent" solutions is selected for breeding from the pool selected previously. By producing a "child" solution using the above methods of crossover and mutation, a new solution is created which typically shares many of the characteristics of its "parents". New parents are selected for each child, and the process continues until a new population of solutions of appropriate size is generated. These processes ultimately result in the next generation population of chromosomes that is different from the initial generation. Generally the average fitness will have increased by this procedure for the population, since only the best organisms from the first generation are selected for breeding, along with a small proportion of less fit solutions, for reasons already mentioned above.

This generational process is repeated until a termination condition has been reached.

Common terminating conditions are:

- A solution is found that satisfies minimum criteria
- Fixed number of generations reached
- Allocated budget (computation time/money) reached
- The highest ranking solution's fitness is reaching or has reached a plateau such that successive iterations no longer produce better results
- Manual inspection
- Any Combinations of the above

Following are the key advantages associated with the GA's:

- Concept is easy to understand
- Modular, separate from application
- Supports multi-objective optimization
- Good for "noisy" environments
- Always an answer; answer gets better with time
- Inherently parallel; easily distributed
- Many ways to speed up and improve a GA-based application as knowledge about problem domain is gained
- Easy to exploit previous or alternate solutions
- Flexible building blocks for hybrid applications
- Substantial history and range of use

Following are the key disadvantages associated with the GA's:

- Choosing basic implementation issues
    - representation

    - population size, mutation rate, ...
    - selection, deletion policies
    - crossover, mutation operators
- Termination Criteria
- Performance, scalability
- Solution is only as good as the evaluation function (often hardest part)

GA's can be used in the applications where:

- Alternate solutions are too slow or overly complicated
- Need an exploratory tool to examine new approaches
- Problem is similar to one that has already been successfully solved by using a GA
- Want to hybridize with an existing solution
- Benefits of the GA technology meet key problem requirements

Keeping the above applications and advantages of GA's in view we can safely assume that the best possible algorithm for the problem at hand is GA's. But the problem lies in the disadvantages stated above. The problem at hand is one where the fitness function is totally application dependent meaning thereby that a fitness function that is perfect for the optimization of one objective may be the worst for the optimization of other objective since the objectives are conflicting. Therefore defining a fitness function for such an application is near impossible. Hence we are left with no choice but to use the pareto front optimization is instead of GA's for the problem at hand.

So the choice left is the use of pareto front optimization algorithms. These algorithms are easy to use and generate a set of non-dominant points the selection of design point is based on the application where the manipulator is to be used.

This project is aimed to provide the simultaneous multi objective workspace optimization of PKMs. The idea is that the methodology developed should be generic, the user needs to know their mechanism and the methodology can handle the rest. The selection of design point from the pareto front again is a task for the user since this selection is application specific.

## 4.3. Design process:

The methodology is designed with the intent of optimizing the workspace of any given parallel manipulator. This has to be done keeping in mind the level of difficulty for the implementation of optimization scheme and the trends generally being followed in the field. Another important factor is considering the current state of the art in the field so that the results generated by the methodology developed are validated against some published work.

The simpler the methodology is the more efficient it would be. The project emphasizes on the simplicity with the physical system involved and the functionality of the system as well.

The methodology is implemented a number of times with different conditions and the results are compared with the benchmark.

## 4.4. The Methodology:

After detailed literature review and deliberation on the research already carried out in the fields of optimization as well as parallel machines, a methodology has been proposed.

- First of all for any given mechanism the design parameters (geometric parameters that effect the objectives) are identified
- The ranges for design parameters and number of points in this range are calculated
- Once the design parameters have been recognized and the points calculated, the optimization objectives are identified
- The third step is to define a cube in the 3-D space where the manipulator has to operate
- The cube is discretized into $X^3$ number of points
- All the points generated are checked for the inverse kinematics solution
- The points satisfying the inverse kinematics solution are added to the workspace against every value of the design parameters
- The objectives are calculated against every point in the workspace
- Objective matrices are normalized
- Normalized matrices are sent to the optimization algorithm
- The optimum values are returned by the algorithm
- The values of objectives are fetched from the already created matrices

First of all the design parameters of the manipulator (parameters to be optimized) are identified. These are the parameters that represent the active joints of the mechanism and in some cases reflect the geometric properties of the PKM that effect the properties of the workspace generated. Once these parameters have been identified, the maximum and minimum values of these parameters are found depending upon the actuators being used or the effects of the geometric properties of the PKM on the performance of the manipulator. These ranges are then divided into a suitable number of points. The choice of this number (the division of the range) depends on the computational power available, however the greater the number of points the better the results are.

After having performed the first step we identify the optimization objectives. The section of objectives is subject to the application where the manipulator has to be used. The objectives that have to be optimized in most of the cases are the following:

- Workspace Volume
- Global Condition Number
- Accuracy
- Velocity Transmission Factor
- Force Transmission Factor
- Inverse Condition Number

At this point it is important that the detail of a brief explanation of these objectives be resented.

The global condition number has been defined in chapter one. This depends on the norms of Jacobian and its inverse. The formulas follow in the next chapter.

Condition number has also been defined in chapter one. Inverse of condition number is simply the reciprocal of the conditioning index. This is a sign of the isotropy of the system. The closer the value is to 1 the better the system performs.

Velocity transmission factor is a measure of how effectively the manipulator transmits the velocity to the work piece. This is important with regards to the machining and surgical robots. To increase the velocity transmission factor we need to maximize the smallest singular value of the inverse Jacobian matrix.

Force transmission factor is a measure of how effectively the manipulator transmits the force to the work piece. This too is important with regards to the machining and surgical robots. To increase the force transmission factor we need to maximize the smallest singular value of the Jacobian matrix.

Accuracy, as the name suggest, is the measure of how accurately the manipulator can position itself at a given point. This parameter is important in every kind of applications of the parallel manipulators. To increase the accuracy of the manipulator we need to minimize the largest singular value of the inverse Jacobian matrix.

After the identification of optimization objectives the next step is generating a cube that encloses the 3-D space in which the manipulator has to operate. The 3-dimensional space is discretized into X3 number of points where 'X' is the amount of divisions of each axis. The points so created are then used for further calculations.

Each of the 3-dimensional points is then checked as a candidate point for being in the workspace of the manipulator. A candidate point that fulfills inverse kinematics equation of the manipulator is considered to be in the workspace of the PKM. The set of these points is called constrained workspace of the manipulator. Once the workspace of the PKM is calculated the matrices for the objectives functions are created.

Every factor other than the workspace volume is dependent either on Jacobian or inverse Jacobian of the manipulator.

This set of calculations is repeated for all the values of the design variables. The matrices so created are normalized and sent to the optimization algorithm which optimizes the data and gives the optimum points in the design space.

Increase in the number of divisions in the 3-dimensional space results in the improvement of results. However this also increases the computational power required to perform the calculations.

For the optimization algorithm, a number of choices were considered first of all the simplest method of weighted sums was considered. The key advantage of this method is the simple mathematics involved in it and the ease with which this can be implemented. The problem however is; since there hasn't been any research in the simultaneous optimization; the relative weights of our objectives are not available in the literature and there aren't any guidelines available in the literature to decide the relative weights.

The next choice in this regard was the implementation of the evolutionary algorithms. This option was not exercised due to the fact that the implementation of evolutionary algorithms in itself is a huge task.

The natural choice in this scenario for this task was the pareto front optimization. Pareto front optimization is a simple approach. Pareto front is the collection of non-dominant points in the design space. A non- dominant point is a point in the design space where if u want to improve the performance of one of the objectives it worsens at least one of the other objectives. Selection of the final result from the pareto front however is again application specific.

Calculating pareto front for 6 objectives is a tedious task hence a new approach to calculating multiple objective pareto front was adopted. There are six objectives that need to be optimized. So to start with a pareto front with two objectives is calculated; the two objectives used are workspace volume and global conditioning index. In the next step another pareto front is calculated, in this step however; global conditioning index is replaced by any other objective while workspace volume remains there. Similarly 5 pareto fronts are calculated one each for every objective against the workspace volume. The points of these pareto fronts in objectives space are mapped into the design space. This yield five sets of points. Now the final design points are selected by taking out all the points that appear in every pareto front. This approach simplifies the calculations however; it does not affect the results.

## 4.5. Validation of results:

The research paper in [43] was benchmarked for the validation of results. The mechanism used is a 3-UPU parallel kinematic machine. Results generated in the benchmark however are based on the single objective optimization.

# Chapter 5: Validation

The machine in this thesis is a 3-UPU PKM. It consists of two triangular plates connected to each other with three legs. The legs are connected to the triangular plates with the help of universal joints. The legs themselves are prismatic joints. The basic model of the mechanism is explained in this chapter.

## 5.1. Geometric Model:

Consider the base triangular plate of the 3-UPU mechanism. The three vertices of the triangular base plate are known as A1, A2 and A3. The vertices of the moving triangular plate are B1, B2 and B3. Consider the following triangular figure; here the centroid and vertices of the triangle are labeled. The centroid of the base triangular plate is the global origin or the origin of the global co-ordinate system.

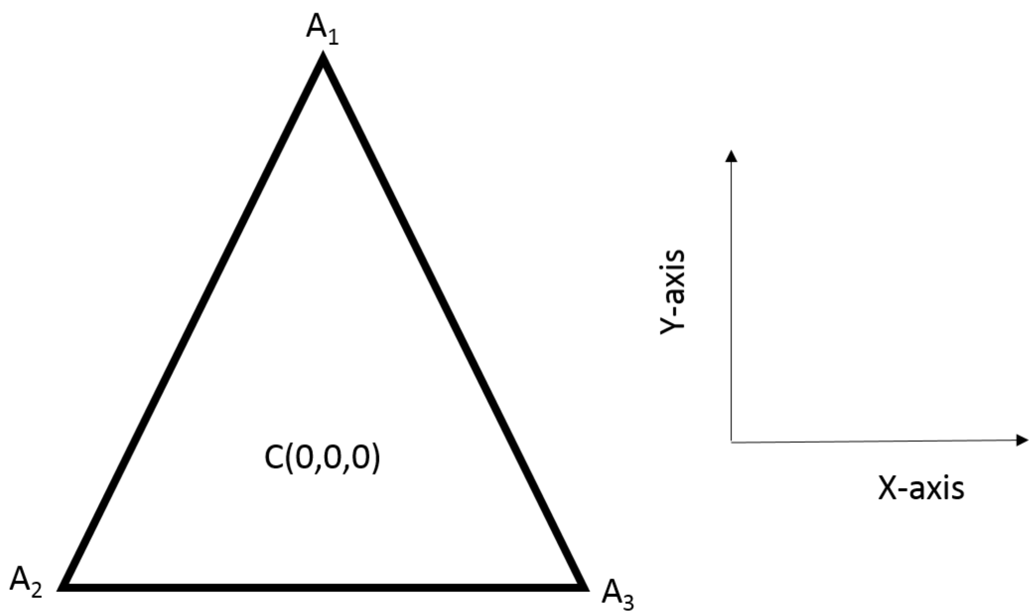



*Figure 5-1: Geometric model of the manipulator*

The base plate is an equilateral triangle. Let the length of each side of triangle is Ex units. The coordinates of the point A1 A2 and A3 are (Ex/-2, (-1)Ex*cos(30)/3), (Ex/2;(-1)Ex*cos(30)/3)and (0;2*Ex*cos(30)/3) respectively. Similarly the mobile platform is also an equilateral triangle. Let the lengths of all sides of the mobile platform is Ex1 units. The coordinates of the points B1, B2 and B3 are (Ex1/-2, (-1)Ex1*cos(30)/3), (Ex1/2;(-

1)Ex1*cos(30)/3)and (0;2*Ex1*cos(30)/3) respectively. These equations can be derived by using the elementary geometry.

Since the legs of the mechanism are prismatic joints therefore we define a lower and an upper range of the leg lengths. Maximum length of any leg at any point in time is 1 and the minimum length can be found out by using the formula $q_{min} = 1 / (1 + s_i/100)$ where $s_i$ is the maximum translation for each leg's prismatic actuator.

## 5.2. Design Parameters to be Optimized:

The two geometric parameters that control the performance and the design of the manipulator are $s_i$ and $c_i$. As stated earlier $s_i$ is the maximum translation of the leg's prismatic actuator while $c_i$ is the difference between the coordinates of vertices of the two platforms in the global reference frame.

The value of si controls the length of legs while ci defines the difference between the sizes of two plates. Hence too large a distance means that the legs will stretch to their maximum to connect them resulting in no motion or very small motion resulting in very small workspace. Manipulators with too small a distance present extra DOF (self-motion of the platform) that possibly will not be controlled from the actuation motion. In real world applications this type of designs are not tolerable.

## 5.3. Ranges of the Design Parameters:

The values of ci and si used in this work are as follows:

- ci: 0.27 – 0.645 [41]
- si: 20% - 87.5% [41].

## 5.4. Objectives for Optimization:

The factors being used for optimization are as follows:

- Workspace volume
- Inverse of condition number
- Global condition number
- Maximum singular value of inverse Jacobian
- Minimum singular value of inverse Jacobian
- Minimum singular of value of Jacobian.

## 5.5. Requirements for the Objectives for Optimization:

The requirements are as follows on these factors:

- Maximize the workspace volume
- Maximize inverse of condition number
- Maximize global condition number
- Minimize the maximum singular value of inverse Jacobian
- Maximize the minimum singular value of inverse Jacobian
- Maximize the minimum singular value of Jacobian

These requirements are conflicting to each other i.e. improving one worsens the other; hence require to be handled carefully to optimize the workspace of the mechanism in such a way that all the factors are at an optimum value. The technique used is the pareto front optimization. Pareto front is the set of points that have the optimum values of all the factors, trying to improve any of the factors worsens at least one of the other factors.

## 5.6. Brief translation:

The methodology developed is useful in the sense since it yields the results that conform to the already published works and the level of complexity involved is very low. The methodology proposed is one of a kind and the problem of multi-criteria workspace optimization is handled effectively and efficiently. The methodology is practical and can be used for the design of any parallel manipulator even though there is room for improvement in the methodology at the moment.

We have achieved a level of research to grasp an extent of accuracy to be accepted as suitable for this project. The design of this system completely satisfies the idea, objectives and scope of the project. The simplicity, efficiency, accuracy and output of the project completely match with the initial objectives of the project.

## 5.7. Iteration:

Generating the results once is not sufficient the results are generated with a number of different conditions. This is important to make sure that the methodology works precisely as intended. The iterations performed change a couple of values and then the whole cycle is repeated but the results generated again match the benchmark.

## 5.8. Impact/ longevity:

This project once adopted can benefit the industry in a lot of ways. It can provide a means of an optimization engine for the design of PKMs in the fields of:

- Tele-scopes
- Fine positioning devices
- Fast packaging
- Machine-tool
- Medical application
- Motion Simulators
- Optics
- Micro-component fabrication

The project also enhances the ability to use the small workspace of parallel manipulators to the fullest and in the most effective way possible. This also helps in designing the mechanism for a specific problem with emphasis on the factors critical to the application.
Following flow chart shows the algorithm.

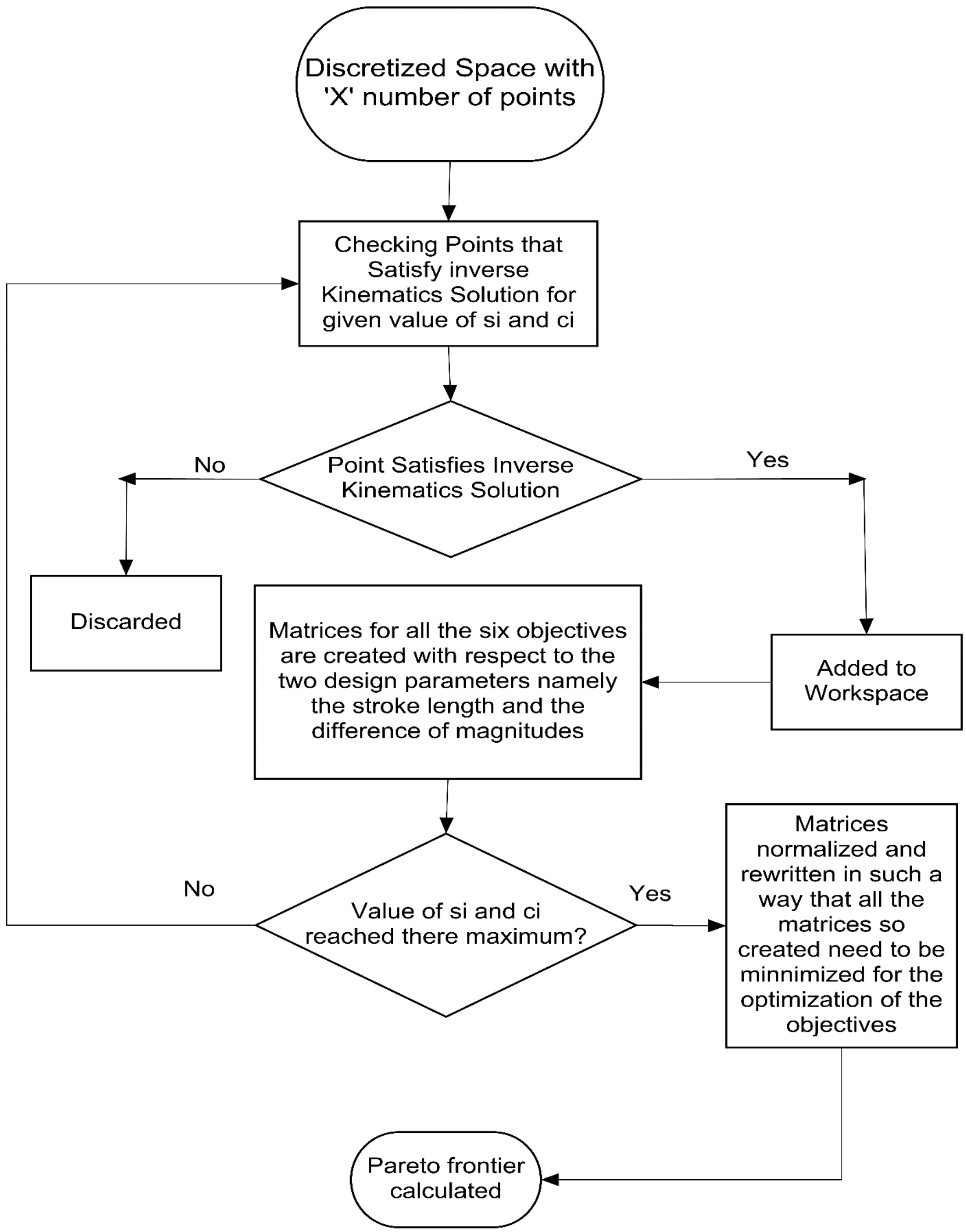



*Figure 5-2: The flow chart of the program*

Increase in the number of divisions in the 3-dimensional space results in the improvement of results. However if only one objective function is optimized at a time the resulting pareto front contains only one point i.e. the optimum value of the design parameters to achieve the optimum performance of the manipulator.

Points in the pareto front varies with the discretization of the space but does not follow any specific trend. The pareto front thus calculated contains all the points where all the objectives have the optimum value and improving anyone results in the deterioration of at least one of the objectives.

The pareto front optimization is one of the most commonly used multi-objective optimization algorithm used. It yields a set of points that yield optimum values of the objective functions. The point to be used is application specific one may use the point that optimizes the required objective, the objective whose optimum behavior is the most critical for the application.

## 5.9. Results and Discussion:

The results generated by this methodology are in accordance with the benchmarked research. The values of objectives differ with that of the benchmark but the trends remain the same. This is due to the fact that different parameters that were unknown for the benchmark had to be assumed these include the volume of the cube which encloses the workspace and the sizes of the plates themselves. Normalizing the objective matrices solves this problem as well.

The following figure shows the plot of workspace volume created in the benchmark:

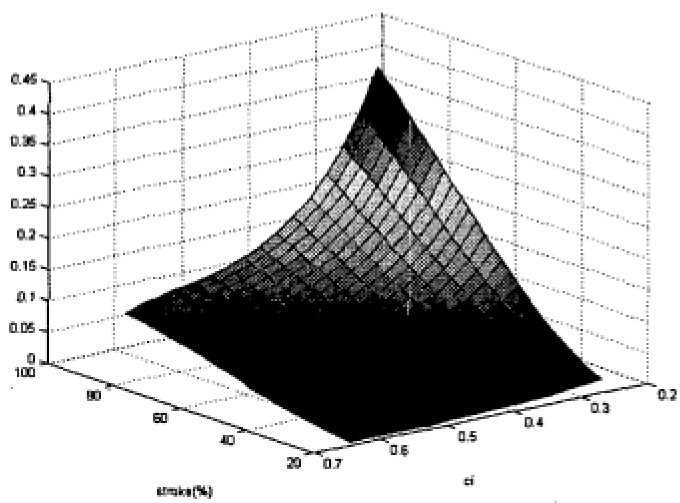



*Figure 5-3: The plot of workspace volume created in the benchmark*

The following figure shows the same plot created using the proposed methodology:

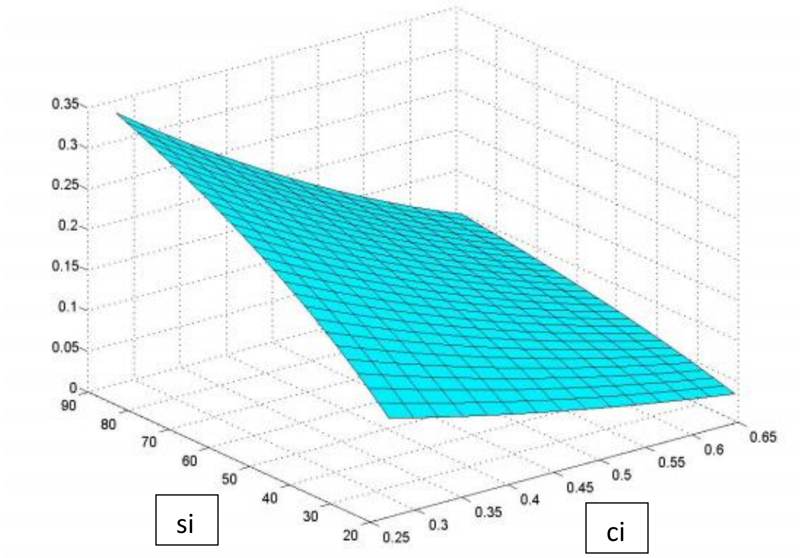



*Figure 5-4: Plot of workspace volume*

The optimization engine is designed to calculate the pareto front by minimizing all the objectives hence the objectives that need to be optimized are first inverted and then sent to the optimizer program to calculate pareto frontier. The plots of these objectives are shown in the following figures:

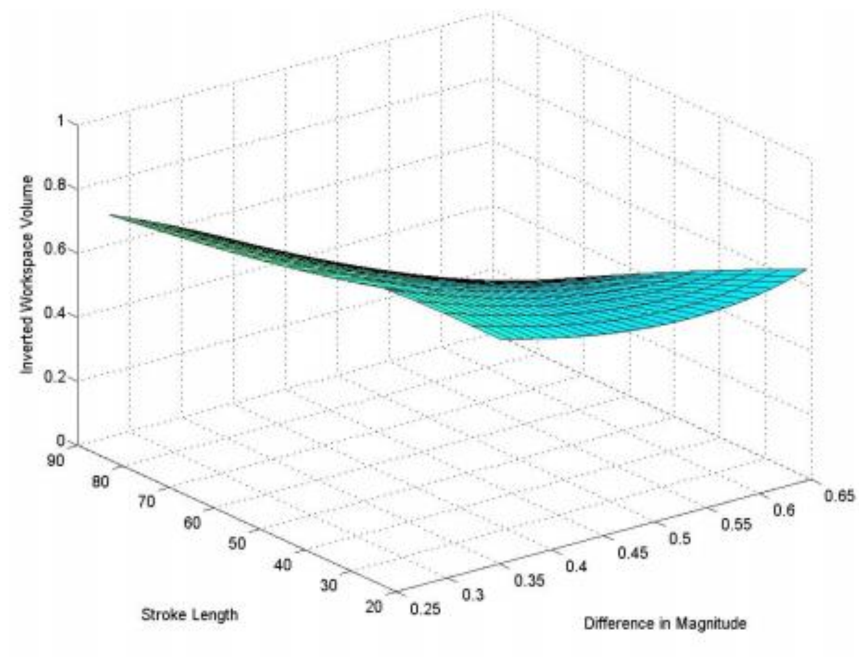



*Figure 5-5: Plot of inverted workspace volume*

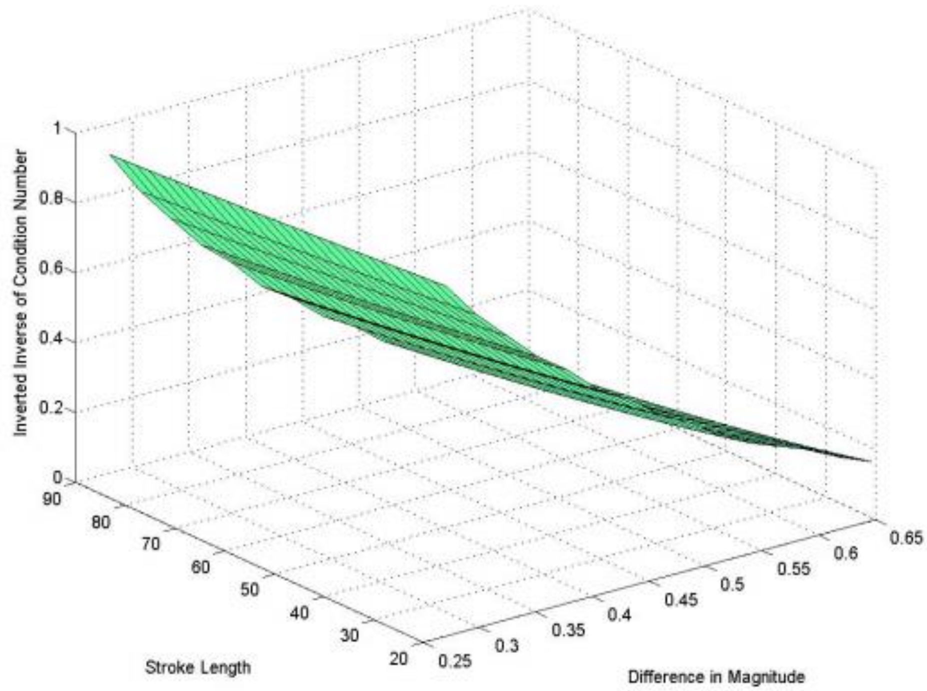



*Figure 5-6: Inverted inverse of condition number*

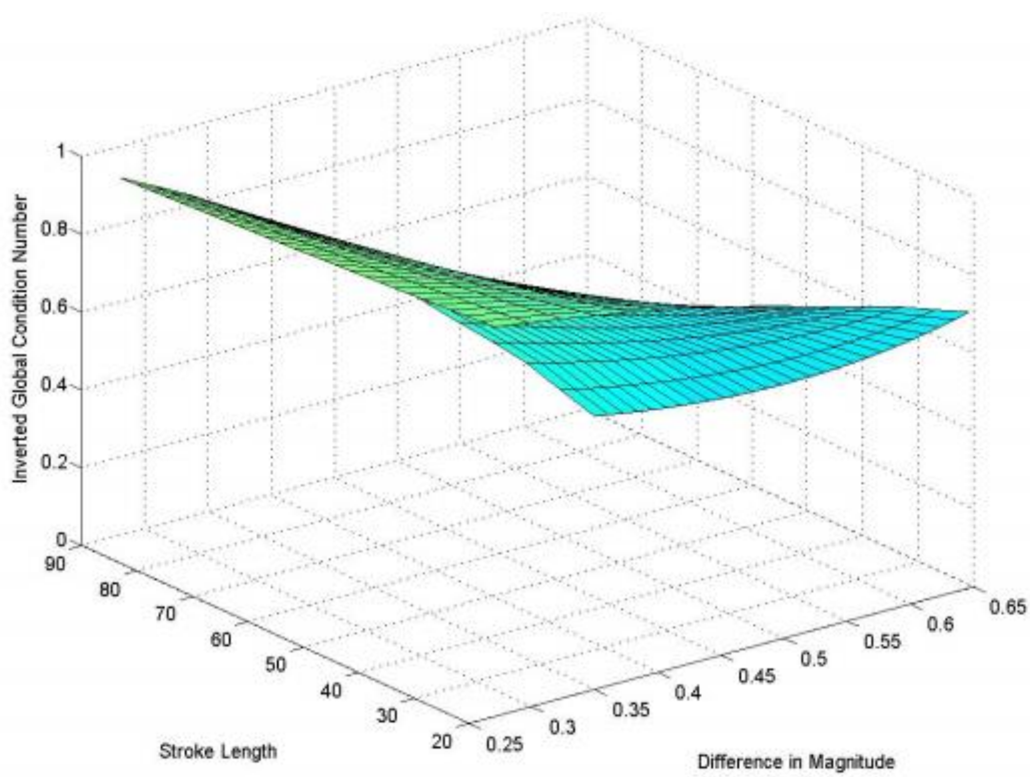



*Figure 5-7: Inverted global condition number*

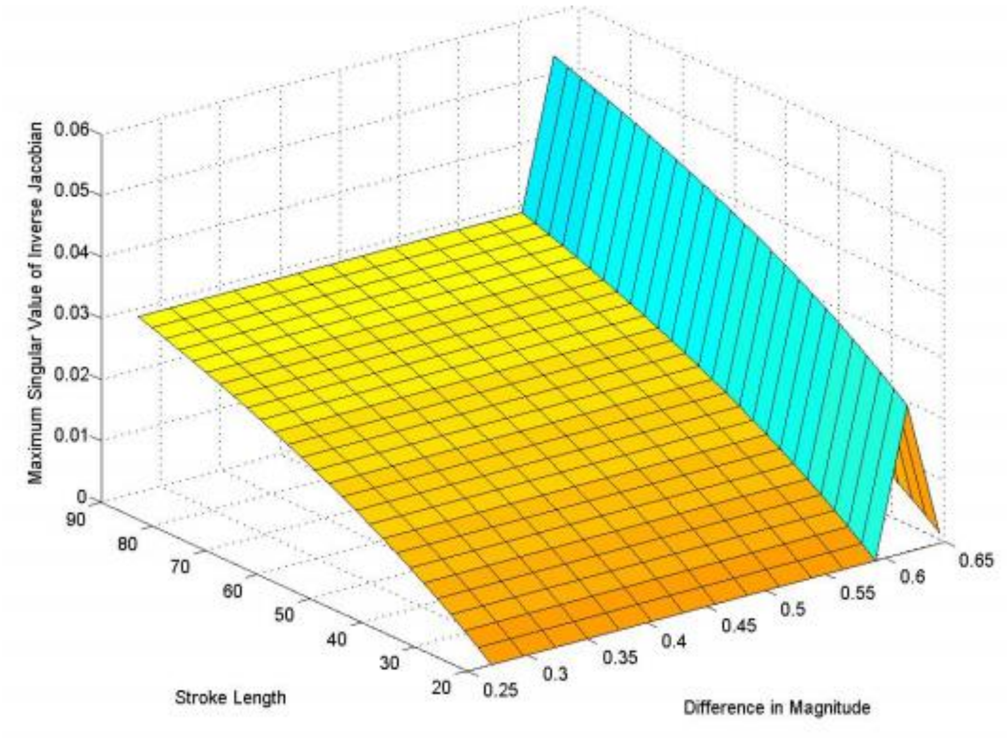



*Figure 5-8: Maximum singular value of inverse Jacobian*

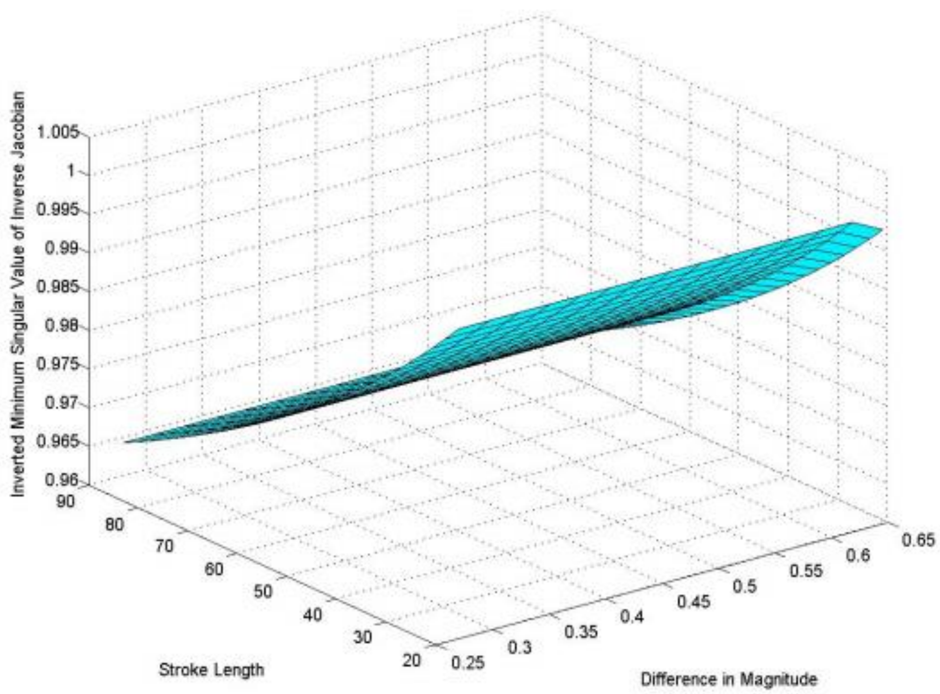



*Figure 5-9: Inverted minimum singular value of inverse Jacobian*

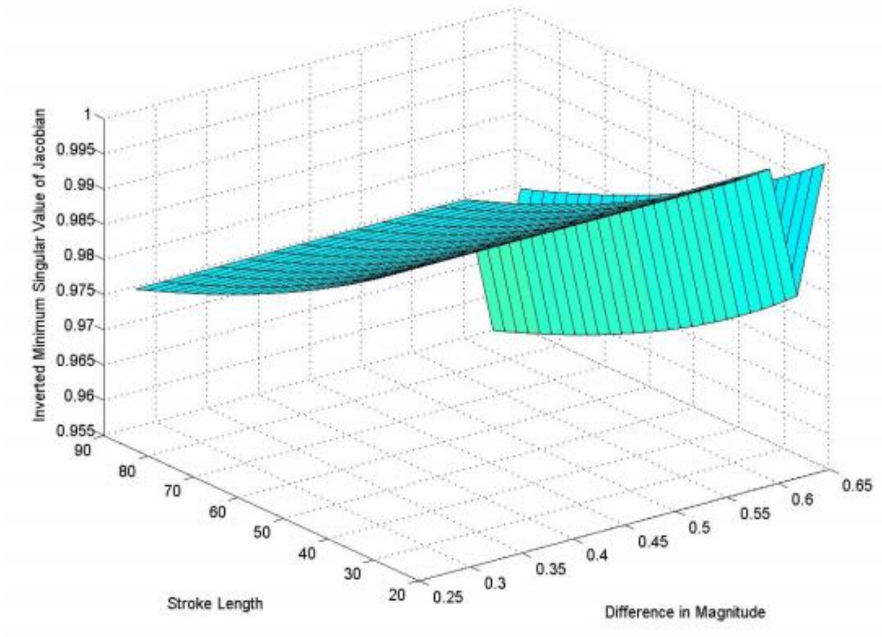



*Figure 5-10: Inverted minimum singular value of Jacobian*

The pareto frontier of these plots is calculated and the following points are found to be pareto optimal:

| **Stroke Length** | **Difference in Magnitude** |
|---|---|
| 20 | 0.59142 |
| 20 | 0.618205 |
| 20 | 0.64499 |
| 22.59615 | 0.59142 |
| 22.59615 | 0.618205 |
| 22.59615 | 0.64499 |
| 25.1923 | 0.59142 |
| 25.1923 | 0.618205 |
| 25.1923 | 0.64499 |
| 27.78845 | 0.59142 |
| 27.78845 | 0.618205 |
| 27.78845 | 0.64499 |
| 30.3846 | 0.59142 |
| 30.3846 | 0.618205 |
| 30.3846 | 0.64499 |
| 32.98075 | 0.59142 |
| 32.98075 | 0.618205 |
| 32.98075 | 0.64499 |

| | |
|---|---|
| 35.5769 | 0.59142 |
| 35.5769 | 0.618205 |
| 35.5769 | 0.64499 |
| 38.17305 | 0.59142 |
| 38.17305 | 0.618205 |
| 38.17305 | 0.64499 |
| 40.7692 | 0.59142 |
| 40.7692 | 0.618205 |
| 40.7692 | 0.64499 |
| 43.36535 | 0.59142 |
| 43.36535 | 0.618205 |
| 43.36535 | 0.64499 |
| 45.9615 | 0.618205 |
| 45.9615 | 0.64499 |
| 48.55765 | 0.59142 |
| 48.55765 | 0.618205 |
| 48.55765 | 0.64499 |
| 51.1538 | 0.59142 |
| 51.1538 | 0.618205 |
| 51.1538 | 0.64499 |
| 53.74995 | 0.618205 |
| 53.74995 | 0.64499 |
| 56.3461 | 0.59142 |
| 56.3461 | 0.618205 |
| 56.3461 | 0.64499 |
| 58.94225 | 0.59142 |
| 58.94225 | 0.618205 |
| 58.94225 | 0.64499 |
| 61.5384 | 0.618205 |
| 61.5384 | 0.64499 |
| 64.13455 | 0.59142 |
| 64.13455 | 0.618205 |
| 64.13455 | 0.64499 |
| 66.7307 | 0.59142 |
| 66.7307 | 0.618205 |
| 66.7307 | 0.64499 |
| 69.32685 | 0.59142 |
| 69.32685 | 0.618205 |
| 69.32685 | 0.64499 |
| 71.923 | 0.618205 |
| 71.923 | 0.64499 |

| | |
|---|---|
| 74.51915 | 0.59142 |
| 74.51915 | 0.618205 |
| 74.51915 | 0.64499 |
| 77.1153 | 0.59142 |
| 77.1153 | 0.618205 |
| 77.1153 | 0.64499 |
| 79.71145 | 0.59142 |
| 79.71145 | 0.618205 |
| 79.71145 | 0.64499 |
| 82.3076 | 0.59142 |
| 82.3076 | 0.618205 |
| 82.3076 | 0.64499 |
| 84.90375 | 0.59142 |
| 84.90375 | 0.618205 |
| 84.90375 | 0.64499 |
| 87.4999 | 0.59142 |
| 87.4999 | 0.618205 |
| 87.4999 | 0.64499 |

*Table 5-1: Pareto front calculated for the six objectives*

When the same optimization code is run for the three optimization parameters of the benchmark it yields the following results:

| **Stroke Length** | **Difference in Magnitude** |
|---|---|
| 87.4999 | 0.64499 |

*Table 5-2: Pareto front for the three objectives in the benchmark*

This however is not the validation of our results since these values are obtained by simultaneously optimizing the three objectives whereas in the benchmark research the objectives have been optimized one at a time.

When the proposed methodology is run for the workspace optimization subject to the global conditioning index it yields the following results:

| **Stroke Length** | **Difference in Magnitude** |
|---|---|
| 85 | 0.27 |

*Table 5-3: Pareto front for global conditioning index as calculated in the benchmark*

These are the values that were obtained in the benchmark research. Now that our results match with the bench marked research it can be stated that the proposed methodology works well on simultaneously optimizing the objectives since it gives the same results as the benchmark when used with a single objective.

# Chapter 6: Conclusion

This research aims at proposing a methodology for the workspace optimization of PKMs; using multiple criteria simultaneously. Another important factor that was not foreseen at the beginning of the research is the optimization scheme. So far no such research has been carried out where multiple objectives are simultaneously used to optimize the workspace of a PKMs.

A methodology is proposed that performs the multi criteria workspace optimization of PKMs. The optimization scheme used is pareto front optimization. Another novelty of the research is that a new method to implement the pareto front optimization is proposed and implemented.

The validation of proposed methodology was a challenge in itself since no work of such sort is available in the literature. Validation is performed by generating the results of single objective optimization using the new methodology. The proposed methodology produces results in accordance with the benchmarked research when a single optimization objective is selected. It is therefore expected that the results produced for the simultaneous optimization of all the six objectives are correct. Hence a methodology is proposed and the results validated for the "Multi-criteria Workspace Optimization of PKMs".

## 6.1. Future Works:

Even though the methodology is validated there are still some points where it can be improved. Firstly there is no consideration for the collision of the mechanism links with each other. In real applications all such poses must be avoided, hence this needs to be added into the methodology. Serial programming was used in the code for this research; this proved to be a major drawback since the computational power needed for such computations was not available. For this reason the discretization used wherever required was very small affecting the precision of the solution. Keeping above discussion in view it is recommended that parallel programing with higher discretization should be used instead of using serial programming with lower discretization. This not only will increase the precision of the calculations but also yield a better, more realistic and optimized solution. The validation part also needs some attention it is recommended that the scheme be implemented on at least a couple of more mechanisms and the results be compared with the published research. Yet another aspect of improvement in this regard is the relative weights of the objectives. So far there is no scheme in the literature that can be used for the calculation of the relative weights. If the relative weights of the objectives could somehow be found it will highly improve the methodology since weighted sum or modified weighted sum optimization schemes can be used then. These schemes are easier to understand and implement and make the methodology even simpler.

# REFERENCES


[1]. Parallel robots 2nd edition by J. P. Merlet, INRIA, Sophia-Antipolis, France.

[2]. Gosselin C. Kinematic analysis optimization and programming of parallel robotic manipulators. Ph.D. Thesis, McGill University, Montr´eal, June, 15, 1988.

[3]. Huy Hoang Pham and I-Ming Chen, "Optimal Synthesis for Workspace and Manipulability of Parallel Flexure Mechanism". Proceeding of the 11th World Congress in Mechanism and Machine Science August 18-21, 2003, Tianjin, China.

[4]. Yoshikawa, T., Analysis and control of robot manipulators with redundancy, In: 1st Int. Symposium of Robotics Research, New Hampshire, USA, 435-446, 1984.

[5]. Yoshikawa, T., Manipulability of robotic mechanism, In: Int. J. of Robotics Research, 4(2): 439-446, 1984.

[6]. Yuan Yun, Liping Wang, Liwen Guan. Dimensional Synthesis of a 3-DOF Parallel Manipulator. 2004 IEEE International Conference on Systems, Man and Cybernetics.

[7]. Yunjiang Lou, Guanfeng Liu, Ni Chen, and Zexiang Li, "Optimal Design of Parallel Manipulators for Maximum Effective Regular Workspace".

[8]. Jeha Ryu* and Jongeun Cha, "Volumetric Error Analysis and Architecture Optimization for Accuracy of HexaSlideType Parallel Manipulators".

[9]. Chen, S.; Chang, T.; Lin, Y. Applications of equivalent components concept on the singularity analysis of TRR-XY hybrid parallel kinematic machine tools. Int. J. Adv. Manuf. Technol. 2006, 30, 778-788.

[10]. Refaat, S.; Herve, J.; Nahavandi, S. Two-mode overconstrained three-DOFs rotational translational linear motor based parallel-kinematics mechanism for machine tool applications. Robotica 2007, 25, 461-466.

[11]. Gao, Z.; Zhang, D.; Hu, X.; Ge, Y. Design, analysis and stiffness optimization of a three-degree-of-freedom parallel manipulator. Robotica 2010, 28, 349-357.

[12]. Hostens, I.; Anthonis, J.; Ramon, H. New design for a 6 dof vibration simulator with improved reliability and performance. Mech. Syst. Signal Process. 2005, 19, 105-122.

[13]. Hou, Y.; Zeng, D.; Yao, J.; Kang, K.; Lu, L.; Zhao, Y. Optimal design of a hyperstatic Stewart platform-based force/torque sensor with genetic algorithms. Mechatronics 2009, 19, 199-204.

[14]. Jin, Z.; Gao, F.; Zhang, X. Design and analysis of a novel isotropic six-component force/torque sensor. Sens. Actuators A 2003, 109, 17-20.

[15]. Tsai, L.-W., 1999, ''Robot Analysis—The Mechanics of Serial and Parallel Manipulators,'' John Wiley & Sons, New York.

[16]. Merlet, J.-P., 2000, Parallel Robots, Kluwer, Dordrecht, Netherlands.

[17]. Hamlin, G., and Sanderson, A., 1997, ''TETROBOT: A Modular Approach to Parallel Robotics,'' IEEE Rob. & Autom. Mag., 4~1!, pp. 42–49.

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

[19]. J.-P. Merlet. An improved design algorithm based on interval analysis for spatial parallel manipulator with specified workspace. In Proceedings of IEEE International Conference on Robotics and Automation, pages 1289–1294, 2001.

[20]. E. Ottaviano and M. Ceccarelli. Optimal design of CAPAMAN (Cassino parallel manipulator) with a specific orientation workspace. Robotica, 20(2):159–166, 2002.

[21]. Y.J. Lou, G.F. Liu, J.J. Xu, and Z.X. Li. A general approach for optimal kinematic design of parallel manipulators. In Proceedings of IEEE International Conference on Robotics and Automation, pages 3659 –3664, 2004.

[22]. R. Boudreau and C. M. Gosselin, The synthesis of planar parallel manipulators with a genetic algorithm, J. Mech. Des. 121, 533–537 (1999).

[23]. M. A. Laribi, L. Romdhane and S. Zeghloul, Analysis and dimensional synthesis of the DELTA robot for a prescribed workspace, Mech. Mach. Theory 42, 859–870 (2007).

[24]. Y. Li and Q. Xu, Optimal kinematic design for a general 3-PRS spatial parallel manipulator based on dexterity and workspace, in: Proc. 11th Int. Conf. on Machine Design and Production, Antalya, pp. 571–584 (2004).

[25]. Q. Xu and Y. Li, Kinematic analysis and optimization of a new compliant parallel micromanipulator, Int. J. Adv. Robotic Syst. 3, 351–358 (2006).

[26]. H. Lipkin and J. Duffy. Hybrid twist and wrench control for a robotic manipulator. Transactions of ASME, Journal of Mechanisms, Transmissions, and Automation in Design, 110(2):138–144, 1988.

[27]. M. Tandirci, J. Angeles, and F. Ranjbaran. The characteristic point and characteristic length of robotic manipulator. In Proc. ASME 22nd Biennial Conf. Robot., Spatial Mech. Mech. Syst., Scottsdale, AZ, Sept. 13-16, pages 203–208, 1992.

[28]. O. Ma and J. Angeles. Optimum architecture design of platform manipulators. In Proceedings of the Fifth International Conference on Advanced Robotics, pages 1130 – 1135, 1991.

[29]. Yunjiang Lou, Dongjun Zhang, and Zexiang Li. Optimal Design of a Parallel Machine Based on Multiple Criteria. Proceedings of the 2005 IEEE International Conference on Robotics and Automation Barcelona, Spain, April 2005.

[30]. M.A. Laribi, A. Mlika, L. Romdhane and S. Zeghloul, "Multi criteria optimum design of 3 dof translational in parallel manipulators (3TPM)".

[31]. M. Ceccarelli, G. Carbone, E. Ottaviano ,2005, An Optimization Problem Approach For Designing Both Serial And Parallel Manipulators. The Int. Sym. on Multibody Systems and Mechatronics Uberlandia, Brazil, 6-9 March.

[32]. Stamper, R.E., Tsai, L.-W., and Walsh, G.C., "Optimization of a three DOF translational platform for well-conditioned workspace", Proc. IEEE Int. Conf. on Robot. Auto.,Albuquerque, p.3250-3255, 1997.

[33]. Zhen Gao and Dan Zhang. Workspace Representation and Optimization of a Novel Parallel Mechanism with Three-Degrees-of-Freedom. Sustainability 2011, 3, 2217-2228; doi:10.3390/su3112217.

[34]. L. J. Stocco, S. E. Salcudean, and F. Sarsani. Optimal kinematic design of a haptic pen. IEEE/ASME 'transactions on mechatronics, 6(3):210-220, Sep. 2001.

[35]. Kim, J. and Khosla, P., Dexterity measures for design and control of manipulators, In: IEEE/RSJ Int. Workshop on Robots and Systems IROS'91, 758-763, 1991.

[36]. Ilya Tyapin, "Collision-Free Workspace Design Optimisation of the 3-DOF Gantry-Tau Parallel Kinematic Machine".

[37]. Mir Amin Hosseini, Hamid-Reza M. Daniali and Hamid D. Taghirad, "Dexterous Workspace Optimization of a Tricept Parallel Manipulator.

[38]. Dan Zhang, "Global Stiffness Optimization of Parallel Robots Using Kinetostatic Performance Indices".

[39]. Raza UR-REHMAN, St´ephane CARO, Damien CHABLAT, Philippe WENGER, "Multiobjective Design Optimization of 3–PRR Planar Parallel Manipulators". Published in 20th CIRP design conference, Nantes :France (2010).

[40]. D. Chablat, Ph.Wenger, F. Majou and J.-P. Merlet, "An Interval Analysis Based Study for the Design and the Comparison of Three-Degrees-of- Freedom Parallel Kinematic Machines". In The International Journal of Robotics Research Vol. 23, No. 6, June 2004, pp. 615-624, DOI: 10.1177/0278364904044079.

[41]. Mircea Badescu and Constantinos Mavroidis, "Workspace Optimization of 3-Legged UPU and UPS Parallel Platforms With Joint Constraints". In Journal of Mechanical Design, MARCH 2004, Vol. 126 / 291.

[42]. Tsai, L.-W., and Joshi, S., 2000, ''Kinematics and Optimization of a Spatial 3-UPU Parallel Manipulator,'' ASME J. Mech. Des., 122~4!, pp. 439–446.

[43]. MirceaBadescu and ConstantinosMavroidis "Workspace Optimization of 3-Legged UPU Parallel Platforms With Joint Constraints".

## APPENDIX 'A'

## The Code:

```
clc
clear all
close all

% This is the programme for the benchmarked paper
% In the name of ALLAH I begin
% It is 09:34 A.M 25th April 2012
% A1 stands for the point where leg 1 is connected to the base
% B1 """""""""""""""""""""""""""""""""""""""""" moving platform
% and so on so forth
% The length of each side of the triangular plate is assumed to be 0.5 units
% The length of each side of the base plate is assumed to be 0.75 units
% The maximum length of the legs is assumed to be 1
% The minimum length of a leg is given by formula qmin = 1 / ( 1 + si/100)
% Matrices Px, Py, Pz contain the cordinates of the points that are to be
% checked for bieng the candidate points.
% The coordinates of the points Bi are calculated by subtractung the local
% frame center from the coordinates of the points Bi in the center local
% frame

ex = 0.75;
ex1 = 0.5;
NOP = 30;
A1 = [ex/-2;ex*cosd(30)/-3];
A2 = [ex/2;ex*cosd(30)/-3];
A3 = [0;2*ex*cosd(30)/3];
Stroke_Length = 1;
Workspacex = [];
Workspacey = [];
Workspacez = [];
UC_Workspacex = [];
UC_Workspacey = [];
UC_Workspacez = [];
xbounds = zeros(1,6);
ybounds = zeros(1,6);

flag = 0;
for si = 17.40385:2.59615:87.5
    Diff_Mag = 1;
    qmin(Stroke_Length) = 1 / ( 1 + si/100);
    zmin = (sqrt(qmin(Stroke_Length)^2 - ((ex - ex1) / 2)^2));
    zmax = (sqrt(1 - ((ex - ex1) / 2)^2));
```

```
   %     Pz = sqrt(qmin(Stroke_Length)^2 - ((ex - ex1) / 2)^2):(sqrt(1 - ((ex - ex1) / 2)^2) - 
sqrt(qmin(Stroke_Length)^2 - ((ex - ex1) / 2)^2)) / (NOP - 1) :sqrt(1 - ((ex - ex1) / 2)^2);

   Pz = zmin : (zmax - zmin)/ (NOP - 1) : zmax;
   for ci = 0.27:0.026785:0.645
      SOIOCN = 0;
      xbounds(1) = (ex1/ -2) - A1(1) + ci;
      xbounds(2) = (ex1/ 2) - A2(1) + ci;
      xbounds(3) = - A3(1) + ci;
      xbounds(4) = (ex1/ -2) - A1(1) - ci;
      xbounds(5) = (ex1/ 2) - A2(1) - ci;
      xbounds(6) = - A3(1) - ci;
      ybounds(1) = (ex1*cosd(30)/-3) - A1(2) + ci;
      ybounds(2) = (ex1*cosd(30)/3) - A2(2) + ci;
      ybounds(3) = (2*ex1*cosd(30)/3) - A3(2) + ci;
      ybounds(4) = (ex1*cosd(30)/-3) - A1(2) - ci;
      ybounds(5) = (ex1*cosd(30)/3) - A2(2) - ci;
      ybounds(6) = (2*ex1*cosd(30)/3) - A3(2) - ci;
      xmin = min(xbounds);
      xmax = max(xbounds);
      ymin = min(ybounds);
      ymax = max(ybounds);
      Px = xmin : ((xmax - xmin)/(NOP - 1)) : xmax;
      Py = ymin : ((ymax - ymin)/(NOP - 1)) : ymax;

      %         Px = min(xbounds):(max(xbounds) - min(xbounds)) / (NOP - 1) :max(xbounds);
      %         Py = min(ybounds):(max(ybounds) - min(ybounds)) / (NOP - 1) :max(ybounds);
      %         Vol_Factor = (((max(xbounds) - min(xbounds)) * (max(ybounds) - min(ybounds)) * 
(Pz(NOP) - Pz(1))) / ((length(Px)) * (length(Py)) * (length(Pz))));

      Vol_Factor = ((xmax - xmin) * (ymax - ymin) * (Pz(NOP) - Pz(1))) / ((length(Px)) * 
(length(Py)) * (length(Pz)));
      z = 1;
      ucw = 1;
      max_s_v = [];
      for i = 1:1:NOP
         l = 1;
         for j = 1:1:NOP
            m = 1;
            for k = 1:1:NOP
               A1 = [ex/-2;ex*cosd(30)/-3];
               A2 = [ex/2;ex*cosd(30)/-3];
               A3 = [0;2*ex*cosd(30)/3];
               B1 = [(ex1/-2) - Px(i) ;(ex1*cosd(30)/-3) - Py(j)];
               B2 = [(ex1/2) - Px(i); (ex1*cosd(30)/-3) - Py(j)];
               B3 = [0 - Px(i); (2*ex1*cosd(30)/3) - Py(j)];
```

```
            C1 = A1 - B1;
            C2 = A2 - B2;
            C3 = A3 - B3;
            magC1 = sqrt(C1(1)^2 + C1(2)^2);
            magC2 = sqrt(C2(1)^2 + C2(2)^2);
            magC3 = sqrt(C3(1)^2 + C3(2)^2);
            Q(1) = sqrt((Px(i) - C1(1))^2 + (Py(j) - C1(2))^2 + Pz(k)^2);
            Q(2) = sqrt((Px(i) - C2(1))^2 + (Py(j) - C2(2))^2 + Pz(k)^2);
            Q(3) = sqrt((Px(i) - C3(1))^2 + (Py(j) - C3(2))^2 + Pz(k)^2);
            if min(Q) >= qmin(Stroke_Length) && max(Q) <= 1        %&& magC1 == ci
              if ci <= magC1 <= ci + 0.00000026785 && ci <= magC2 <= ci + 0.00000026785
&& ci <= magC3 <= ci + 0.00000026785
                Workspacex(z) = Px(i);
                Workspacey(z) = Py(j);
                Workspacez(z) = Pz(k);
                z = z + 1;
                UC_Workspacex(ucw) = Px(i);
                UC_Workspacey(ucw) = Py(j);
                UC_Workspacez(ucw) = Pz(k);
                b1 = [(ex1/-2) - UC_Workspacex(ucw) ;(ex1*cosd(30)/-3) -
UC_Workspacey(ucw); UC_Workspacez(ucw)];
                b2 = [(ex1/2) - UC_Workspacex(ucw); (ex1*cosd(30)/-3) -
UC_Workspacey(ucw); UC_Workspacez(ucw)];
                b3 = [0 - UC_Workspacex(ucw); (2*ex1*cosd(30)/3) - UC_Workspacey(ucw);
UC_Workspacez(ucw)];
                a1 = [ex/-2;ex*cosd(30)/-3; 0];
                a2 = [ex/2;ex*cosd(30)/-3; 0];
                a3 = [0;2*ex*cosd(30)/3; 0];
                u1 = b1 - a1;
                u2 = b2 - a2;
                u3 = b3 - a3;
                ub1 = u1/sqrt(u1(1)^2 + u1(2)^2 + u1(3)^2);
                ub2 = u2/sqrt(u2(1)^2 + u2(2)^2 + u2(3)^2);
                ub3 = u3/sqrt(u3(1)^2 + u3(2)^2 + u3(3)^2);
                Jaco = [ub1(1) ub2(1) ub3(1);ub1(2) ub2(2) ub3(2); ub1(3) ub2(3) ub3(3)];
                Inversejaco = inv(Jaco);
                SOIOCN = SOIOCN +(1/ (norm(Jaco) * norm (Inversejaco)));
                ucw = ucw + 1;
                mx_s_v(ucw) = svds(Inversejaco,1);
                mn_s_v(ucw) = svds(Inversejaco,1,0);
                f_mx_s_v (ucw) = svds (Jaco,1);
                f_mn_s_v (ucw) = svds (Jaco,1,0);

                %                       if mn_s_v(ucw) == 0
                %                           mn_s_v(ucw) = 10000;
                %                       end
```

```
            end
          elseif 0 <= min (Q) <= qmin(Stroke_Length)
            %                if ci <= magC1 <= ci + 0.026785 || ci <= magC2 <= ci +
0.026785 || ci <= magC3 <= ci + 0.026785

            UC_Workspacex(ucw) = Px(i);
            UC_Workspacey(ucw) = Py(j);
            UC_Workspacez(ucw) = Pz(k);
            ucw = ucw + 1;
            b1 = [(ex1/-2) - UC_Workspacex(ucw) ;(ex1*cosd(30)/-3) -
UC_Workspacey(ucw); UC_Workspacez(ucw)];
            b2 = [(ex1/2) - UC_Workspacex(ucw); (ex1*cosd(30)/-3) -
UC_Workspacey(ucw); UC_Workspacez(ucw)];
            b3 = [0 - UC_Workspacex(ucw); (2*ex1*cosd(30)/3) - UC_Workspacey(ucw);
UC_Workspacez(ucw)];
            a1 = [ex/-2;ex*cosd(30)/-3; 0];
            a2 = [ex/2;ex*cosd(30)/-3; 0];
            a3 = [0;2*ex*cosd(30)/3; 0];
            u1 = b1 - a1;
            u2 = b2 - a2;
            u3 = b3 - a3;
            ub1 = u1/sqrt(u1(1)^2 + u1(2)^2 + u1(3)^2);
            ub2 = u2/sqrt(u2(1)^2 + u2(2)^2 + u2(3)^2);
            ub3 = u3/sqrt(u3(1)^2 + u3(2)^2 + u3(3)^2);
            Jaco = [ub1(1) ub2(1) ub3(1);ub1(2) ub2(2) ub3(2); ub1(3) ub2(3) ub3(3)];
            Inversejaco = inv(Jaco);
            SOIOCN = SOIOCN +(1/ (norm(Jaco) * norm (Inversejaco)));
            mx_s_v(ucw) = svds(Inversejaco,1);
            mn_s_v(ucw) = svds(Inversejaco,1,0);
            f_mx_s_v (ucw) = svds (Jaco,1);
            f_mn_s_v (ucw) = svds (Jaco,1,0);

            %                  if mn_s_v(ucw) == 0
            %                      mn_s_v(ucw) = 10000;
            %                  end
            %                end

          end
        end
      end

    end
    Volume(Stroke_Length,Diff_Mag) = Vol_Factor * length(Workspacex);
    ICN(Stroke_Length,Diff_Mag) = SOIOCN/length(UC_Workspacez);
```

```
        GCN(Stroke_Length,Diff_Mag) = SOIOCN * (((max(xbounds) - min(xbounds)) *
(max(ybounds) - min(ybounds)) * (Pz(NOP) - Pz(1)))/length(UC_Workspacex));
        max_sing_value (Stroke_Length,Diff_Mag) = max(mx_s_v);
        for q = 1:1:size(mn_s_v)
            if mn_s_v(q) == 0
                mn_s_v(q) = 1000;
            end
        end
        min_sing_value (Stroke_Length,Diff_Mag) = min(mn_s_v);
        f_max_sing_value (Stroke_Length, Diff_Mag) = max(f_mx_s_v);
        for q = 1:1:size(f_mn_s_v)
            if f_mn_s_v(q) == 0
                f_mn_s_v(q) = 1000;
            end
        end
        f_min_sing_value (Stroke_Length,Diff_Mag) = min(f_mn_s_v);
        Diff_Mag = Diff_Mag + 1;
    end
    Stroke_Length = Stroke_Length + 1;
    flag = flag + 1
end

% % % % % % % % % % % % % % % % % % [p q] = size(Volume);
% % % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % % for i =1:1:p-1
% % % % % % % % % % % % % % % % % %     for j = 1:1:q
% % % % % % % % % % % % % % % % % %         Volume1 (i,j) = Volume (i+1,j);
% % % % % % % % % % % % % % % % % %         ICN1 (i,j) = ICN (i+1,j);
% % % % % % % % % % % % % % % % % %         GCN1 (i,j) = GCN (i+1,j);
% % % % % % % % % % % % % % % % % %         max_sing_value1 (i,j) =
max_sing_value(i+1,j);
% % % % % % % % % % % % % % % % % %         min_sing_value1 (i,j) =
min_sing_value(i+1,j);
% % % % % % % % % % % % % % % % % %         f_max_sing_value1 (i,j) =
f_max_sing_value(i+1,j);
% % % % % % % % % % % % % % % % % %         f_min_sing_value1 (i,j) =
f_min_sing_value(i+1,j);
% % % % % % % % % % % % % % % % % %     end
% % % % % % % % % % % % % % % % % % end
% % % % % % % % % % % % % % % % % % ci = 0.27:0.026785:0.645;
% % % % % % % % % % % % % % % % % % si = 20:2.59615:87.5;
% % % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % % surfl (ci,si,Volume1)
% % % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % % surfl (ci,si,ICN1)
% % % % % % % % % % % % % % % % % % figure
```

```
% % % % % % % % % % % % % % % % % % surfl (ci,si,GCN1)
% % % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % % surfl(ci,si,max_sing_value1)
% % % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % % surfl(ci,si,min_sing_value1)
% % % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % % surfl(ci,si,f_max_sing_value1)
% % % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % % surfl(ci,si,f_min_sing_value1)


% To increase the accuracy of the manipulator we need to minimize the
% maximum singular value of the inverse Jacobian matrix.


% To increase the velocity transmission factor we need to maximize the
% minimum singular value of the inverse Jacobian matrix.


% To increase the force transmission factor we need to maximize the minimum
% singular value of the Jacobian matrix.

for i = 1:1:Stroke_Length - 1

   for j = 1:1:Diff_Mag - 1

        WorkspaceVolume (i,j) = ( Volume(i,j) - min(min(Volume)))/(max(max(Volume)) -
min(min(Volume)));

        inverted_workspace_volume = 1 - WorkspaceVolume;

        % minimize inverted_workspace_volume

        InverseOfConditionNumber (i,j) = ( ICN(i,j) - min(min(ICN)))/(max(max(ICN)) -
min(min(ICN)));

        inverted_inverse_of_condition_number = 1 - InverseOfConditionNumber;

        % minimize inverted_inverse_of_condition_number

        GlobalConditionNumber (i,j) = ( GCN(i,j) - min(min(GCN)))/(max(max(GCN)) -
min(min(GCN)));

        inverted_global_condition_number = 1 - GlobalConditionNumber;

        % minimize  inverted_global_condition_number
```

```
        MaximumSingularValueOfInverseJacobian (i,j) = ( max(max(max_sing_value)) -
max_sing_value(i,j))/(max(max(max_sing_value)) - min(min(max_sing_value)));

        % minimize  MaximumSingularValueOfInverseJacobian

        MinimumSingularValueOfInverseJacobian (i,j) = (min_sing_value(i,j) -
min(min(min_sing_value)))/(max(max(min_sing_value)) - min(min(min_sing_value)));

        inverted_minimum_singular_value_of_inverse_Jacobian = 1 -
MinimumSingularValueOfInverseJacobian;

        % minimize inverted_minimum_singular_value_of_inverse_Jacobian

        MinimumSingularValueOfJacobian (i,j) = (f_min_sing_value(i,j) -
min(min(f_min_sing_value)))/(max(max(f_min_sing_value)) - min(min(f_min_sing_value)));

        inverted_minimum_singular_value_of_Jacobian = 1 - MinimumSingularValueOfJacobian;

        % minimize  inverted_minimum_singular_value_of_Jacobian

    end

end


[p q] = size(Volume);

for i =1:1:p-1
    for j = 1:1:q
        inverted_workspace_volume1 (i,j) = inverted_workspace_volume (i+1,j);
        inverted_inverse_of_condition_number1 (i,j) = inverted_inverse_of_condition_number
(i+1,j);
        inverted_global_condition_number1 (i,j) = inverted_global_condition_number (i+1,j);
        MaximumSingularValueOfInverseJacobian1 (i,j) =
MaximumSingularValueOfInverseJacobian (i+1,j);
        inverted_minimum_singular_value_of_inverse_Jacobian1 (i,j) =
inverted_minimum_singular_value_of_inverse_Jacobian (i+1,j);

%         f_max_sing_value1 (i,j) = f_max_sing_value(i+1,j);

        inverted_minimum_singular_value_of_Jacobian1 (i,j) =
inverted_minimum_singular_value_of_Jacobian (i+1,j);
    end
end
ci = 0.27:0.026785:0.645;
```

```
si = 20:2.59615:87.5;

surfl (ci,si,inverted_workspace_volume1)
title ('Inverted Workspace Volume');
xlabel ('Difference in Magnitude');
ylabel ('Stroke Length');
zlabel ('Inverted Workspace Volume');
figure
surfl (ci,si,inverted_inverse_of_condition_number1)
title ('Inverted Inverse of Condition Number');
xlabel ('Difference in Magnitude');
ylabel ('Stroke Length');
zlabel ('Inverted Inverse of Condition Number');
figure
surfl (ci,si,inverted_global_condition_number1)
title ('Inverted Global Condition Number');
xlabel ('Difference in Magnitude');
ylabel ('Stroke Length');
zlabel ('Inverted Global Condition Number');
figure
surfl(ci,si,MaximumSingularValueOfInverseJacobian1)
title ('Maximum Singular Value of Inverse Jacobian');
xlabel ('Difference in Magnitude');
ylabel ('Stroke Length');
zlabel ('Maximum Singular Value of Inverse Jacobian');
figure
surfl(ci,si,inverted_minimum_singular_value_of_inverse_Jacobian1)
title ('Inverted Minimum Singular Value of Inverse Jacobian');
xlabel ('Difference in Magnitude');
ylabel ('Stroke Length');
zlabel ('Inverted Minimum Singular Value of Inverse Jacobian');

% figure
% surfl(ci,si,f_max_sing_value1)

figure
surfl(ci,si,inverted_minimum_singular_value_of_Jacobian1)
title ('Inverted Minimum Singular Value of Jacobian');
xlabel ('Difference in Magnitude');
ylabel ('Stroke Length');
zlabel ('Inverted Minimum Singular Value of Jacobian');



% For time being we define the objective function as the sum of all the
```

*% above factors that have been calculated in the nested for loop just*
*% above.*

*% % % % % % % % % % % % % % % % BestSoFar = 0;*
*% % % % % % % % % % % % % % % % BestWeightsSoFar.WrokspaceVolumeWeight = 0;*
*% % % % % % % % % % % % % % % %*
*BestWeightsSoFar.InverseOfConditionNumberWeight = 0;*
*% % % % % % % % % % % % % % % % BestWeightsSoFar.GlobalConditionNumberWeight*
*= 0;*
*% % % % % % % % % % % % % % % %*
*BestWeightsSoFar.MaximumSingularValueOfInverseJacobianWeight = 0;*
*% % % % % % % % % % % % % % % %*
*BestWeightsSoFar.MinimumSingularValueOfInverseJacobianWeight = 0;*
*% % % % % % % % % % % % % % % %*
*BestWeightsSoFar.MinimumSingularValueOfJacobianWeight = 0;*
*% % % % % % % % % % % % % % % %*
*% % % % % % % % % % % % % % % % for j =1:1:1000*
*% % % % % % % % % % % % % % % %     for i=1:1:1000*
*% % % % % % % % % % % % % % % %         weights(i).WrokspaceVolumeWeight =*
*rand(1);*
*% % % % % % % % % % % % % % % %         weights(i).InverseOfConditionNumberWeight*
*= rand(1);*
*% % % % % % % % % % % % % % % %         weights(i).GlobalConditionNumberWeight =*
*rand(1);*
*% % % % % % % % % % % % % % % %*
*weights(i).MaximumSingularValueOfInverseJacobianWeight = rand(1);*
*% % % % % % % % % % % % % % % %*
*weights(i).MinimumSingularValueOfInverseJacobianWeight = rand(1);*
*% % % % % % % % % % % % % % % %*
*weights(i).MinimumSingularValueOfJacobianWeight = rand(1);*
*% % % % % % % % % % % % % % % %         sum = weights(i).WrokspaceVolumeWeight +*
*weights(i).InverseOfConditionNumberWeight + weights(i).GlobalConditionNumberWeight +*
*weights(i).MaximumSingularValueOfInverseJacobianWeight +*
*weights(i).MinimumSingularValueOfInverseJacobianWeight +*
*weights(i).MinimumSingularValueOfJacobianWeight;*
*% % % % % % % % % % % % % % % %         weights(i).WrokspaceVolumeWeight =*
*weights(i).WrokspaceVolumeWeight/sum;*
*% % % % % % % % % % % % % % % %         weights(i).InverseOfConditionNumberWeight*
*= weights(i).InverseOfConditionNumberWeight/sum;*
*% % % % % % % % % % % % % % % %         weights(i).GlobalConditionNumberWeight =*
*weights(i).GlobalConditionNumberWeight/sum;*
*% % % % % % % % % % % % % % % %*
*weights(i).MaximumSingularValueOfInverseJacobianWeight =*
*weights(i).MaximumSingularValueOfInverseJacobianWeight/sum;*

```
% % % % % % % % % % % % % % % % %
weights(i).MinimumSingularValueOfInverseJacobianWeight =
weights(i).MinimumSingularValueOfInverseJacobianWeight/sum;
% % % % % % % % % % % % % % % % %
weights(i).MinimumSingularValueOfJacobianWeight =
weights(i).MinimumSingularValueOfJacobianWeight/sum;
% % % % % % % % % % % % % % % % %     end
% % % % % % % % % % % % % % % % %     for i=1:1:1000
% % % % % % % % % % % % % % % % %         ObjectiveFunction =
weights(i).WrokspaceVolumeWeight * WorkspaceVolume +
weights(i).InverseOfConditionNumberWeight * InverseOfConditionNumber +
weights(i).GlobalConditionNumberWeight * GlobalConditionNumber +
weights(i).MaximumSingularValueOfInverseJacobianWeight *
MaximumSingularValueOfInverseJacobian +
weights(i).MinimumSingularValueOfInverseJacobianWeight *
MinimumSingularValueOfInverseJacobian +
weights(i).MinimumSingularValueOfJacobianWeight * MinimumSingularValueOfJacobian;
% % % % % % % % % % % % % % % % %         if max(max(ObjectiveFunction)) >=
BestSoFar
% % % % % % % % % % % % % % % % %             BestSoFar =
max(max(ObjectiveFunction));
% % % % % % % % % % % % % % % % %
BestWeightsSoFar.WrokspaceVolumeWeight = weights(i).WrokspaceVolumeWeight;
% % % % % % % % % % % % % % % % %
BestWeightsSoFar.InverseOfConditionNumberWeight =
weights(i).InverseOfConditionNumberWeight;
% % % % % % % % % % % % % % % % %
BestWeightsSoFar.GlobalConditionNumberWeight =
weights(i).GlobalConditionNumberWeight;
% % % % % % % % % % % % % % % % %
BestWeightsSoFar.MaximumSingularValueOfInverseJacobianWeight =
weights(i).MaximumSingularValueOfInverseJacobianWeight;
% % % % % % % % % % % % % % % % %
BestWeightsSoFar.MinimumSingularValueOfInverseJacobianWeight =
weights(i).MinimumSingularValueOfInverseJacobianWeight;
% % % % % % % % % % % % % % % % %
BestWeightsSoFar.MinimumSingularValueOfJacobianWeight =
weights(i).MinimumSingularValueOfJacobianWeight;
% % % % % % % % % % % % % % % % %         end
% % % % % % % % % % % % % % % % %     end
% % % % % % % % % % % % % % % % % end
% % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % ObjectiveFunction =
BestWeightsSoFar.WrokspaceVolumeWeight * WorkspaceVolume +
BestWeightsSoFar.InverseOfConditionNumberWeight * InverseOfConditionNumber +
BestWeightsSoFar.GlobalConditionNumberWeight * GlobalConditionNumber +
```

```
BestWeightsSoFar.MaximumSingularValueOfInverseJacobianWeight *
MaximumSingularValueOfInverseJacobian +
BestWeightsSoFar.MinimumSingularValueOfInverseJacobianWeight *
MinimumSingularValueOfInverseJacobian +
BestWeightsSoFar.MinimumSingularValueOfJacobianWeight *
MinimumSingularValueOfJacobian;
% % % % % % % % % % % % % % % % % [p q] = size(ObjectiveFunction);
% % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % for i =1:1:p-1
% % % % % % % % % % % % % % % % %     for j = 1:1:q
% % % % % % % % % % % % % % % % %         OF2(i,j) = ObjectiveFunction(i+1,j);
% % % % % % % % % % % % % % % % %         WV2(i,j) =  WorkspaceVolume (i+1,j);
% % % % % % % % % % % % % % % % %         IOCN2(i,j) = InverseOfConditionNumber
(i+1,j);
% % % % % % % % % % % % % % % % %         GCN2(i,j) = GlobalConditionNumber (i+1,j);
% % % % % % % % % % % % % % % % %         MaxSVOIJ2(i,j) =
MaximumSingularValueOfInverseJacobian (i+1,j);
% % % % % % % % % % % % % % % % %         MinSVOIJ2(i,j) =
MinimumSingularValueOfInverseJacobian (i+1,j);
% % % % % % % % % % % % % % % % %         MSVOJ2(i,j) =
MinimumSingularValueOfJacobian (i+1,j);
% % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % %     end
% % % % % % % % % % % % % % % % % end
% % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % si = 20:2.59615:87.5;
% % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % surfl(ci,si,OF2)
% % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % surfl(ci,si,WV2)
% % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % surfl(ci,si,IOCN2)
% % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % surfl(ci,si,GCN2)
% % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % surfl(ci,si,MaxSVOIJ2)
% % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % surfl(ci,si,MinSVOIJ2)
% % % % % % % % % % % % % % % % % figure
% % % % % % % % % % % % % % % % % surfl(ci,si,MSVOJ2)
% % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % maxer = max(max(OF2));
% % % % % % % % % % % % % % % % % [p q] = size(OF2);
% % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % for i = 1:1:p
```

```
% % % % % % % % % % % % % % % % %     for j = 1:1:q
% % % % % % % % % % % % % % % % %         if OF2(i,j) == maxer
% % % % % % % % % % % % % % % % %             strokeflag = i;
% % % % % % % % % % % % % % % % %             diffmagflag = j;
% % % % % % % % % % % % % % % % %         end
% % % % % % % % % % % % % % % % %     end
% % % % % % % % % % % % % % % % % end
% % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % % % stroke_length = ((strokeflag - 1) * 2.59615) + 20
% % % % % % % % % % % % % % % % % Diff_magnitude = ((diffmagflag - 1) * 0.026785) +
0.27
% % % % % % % % % % % % % % % % % workspacevolume =
Volume(strokeflag,diffmagflag)
% % % % % % % % % % % % % % % % % inverseconditionnumber =
ICN(strokeflag,diffmagflag)
% % % % % % % % % % % % % % % % % globalconditionnumber =
GCN(strokeflag,diffmagflag)
% % % % % % % % % % % % % % % % % maximumsingularvalueofinverseJacobian =
max_sing_value(strokeflag,diffmagflag)
% % % % % % % % % % % % % % % % % minimumSingularValueOfInverseJacobian =
min_sing_value(strokeflag,diffmagflag)
% % % % % % % % % % % % % % % % % minimumSingularValueOfJacobian =
f_min_sing_value(strokeflag,diffmagflag)


% Creating an objective matrix that will be used to find the Pareto front
% of the system. The front matrix is logical matrix where '1' represents a
% point that is part of the pareto front.

Temp = 1;


for i = 1:1:Stroke_Length - 2

    for j = 1:1:Diff_Mag - 1

        ObjectiveMatrix (Temp, 1) =  inverted_workspace_volume1 (i, j);

        ObjectiveMatrix (Temp, 2) =  inverted_inverse_of_condition_number1 (i, j);

        ObjectiveMatrix (Temp, 3) =  inverted_global_condition_number1 (i, j);

        %ObjectiveMatrix (Temp, 4) =  MaximumSingularValueOfInverseJacobian1 (i, j);

        %ObjectiveMatrix (Temp, 5) =  inverted_minimum_singular_value_of_inverse_Jacobian1
(i, j);
```

```
        %ObjectiveMatrix (Temp, 6) =  inverted_minimum_singular_value_of_Jacobian1 (i, j);

        Temp = Temp + 1;

    end

end

front = paretoGroup (ObjectiveMatrix);

% To increase the accuracy of the manipulator we need to minimize the
% maximum singular value of the inverse Jacobian matrix.


% To increase the velocity transmission factor we need to maximize the
% minimum singular value of the inverse Jacobian matrix.


 % To increase the force transmission factor we need to maximize the minimum
% singular value of the Jacobian matrix.

% Pareto_Front = struct('Stroke_Length',{}, 'Difference_Magnitude', {}, 'Workspace_Volume',
{},'Maximum_Singular_Value_of_Inverse_Jacobian',{},'Minimum_Singular_Value_of_Inverse_J
acobian',{},'Minimum_Singular_Value_of_Jacobian',{},'Global_Condition_Number',{},'Inverse_
of_Condition_Number',{});


% Once the pareto front is calculated in terms of points we again find the
% different parameters associated with that point by using the already done
% calculations

Temp = 1;

ci = 0.27:0.026785:0.645;
si = 20:2.59615:87.5;
temp2 = 0;
for i = 1:1:Stroke_Length - 2

    for j = 1:1:Diff_Mag - 1

        if front(Temp) == 1
            temp2 = temp2 + 1;
            Pareto_Front(temp2).Stroke_Length = si(i);
            Pareto_Front(temp2).Difference_Magnitude = ci(j);
            Pareto_Front(temp2).Workspace_Volume = Volume(i,j);
```

```
        %Pareto_Front(temp2).Maximum_Singular_Value_of_Inverse_Jacobian =
         max_sing_value (i,j);
        %Pareto_Front(temp2).Minimum_Singular_Value_of_Inverse_Jacobian =
         min_sing_value (i,j);
        %Pareto_Front(temp2).Minimum_Singular_Value_of_Jacobian = f_min_sing_value
(i,j);
        Pareto_Front(temp2).Global_Condition_Number = GCN(i,j);
        Pareto_Front(temp2).Inverse_of_Condition_Number = ICN(i,j);

      end
        Temp = Temp + 1;
   end
end
```