University of Windsor
Scholarship at UWindsor

Electronic Theses and Dissertations | Theses, Dissertations, and Major Papers

1-1-2007

# Reconfigurable kinematics of General Stewart Platform and simulation interface.

Anqi Wang
*University of Windsor*

Follow this and additional works at: https://scholar.uwindsor.ca/etd

Recommended Citation
Wang, Anqi, "Reconfigurable kinematics of General Stewart Platform and simulation interface." (2007). *Electronic Theses and Dissertations*. 7122.
https://scholar.uwindsor.ca/etd/7122

This online database contains the full-text of PhD dissertations and Masters’ theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.

# RECONFIGURABLE KINEMATICS OF GENERAL STEWART PLATFORM AND SIMULATION INTERFACE

By

Anqi Wang

A Thesis
Submitted to the Faculty of Graduate Studies and Research
through Industrial Engineering and Manufacturing Systems
in Partial Fulfillment of the Requirements for
the Degree of Masters of Science at the
University of Windsor

Windsor, Ontario, Canada

2007 © Anqi Wang

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

*Your file Votre référence*
*ISBN: 978-0-494-42313-4*
*Our file Notre référence*
*ISBN: 978-0-494-42313-4*

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

---

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# ABSTRACT

This research introduces a new algorithm to solve the forward kinematics of the General Stewart Platform. Basically, there are at least 20 basic feasible topologies for the General Stewart Platform and many different configurations for each of them, some of which have been studied fully but most of them have not. The new algorithm can be extended to solve every single configuration of General Stewart Platform by slight change of inputs. Unlike the existing algorithm, the proposed algorithm was developed by projective geometry, which enables the extension of solution to any special configuration. In addition, extra sensors are introduced to give a set of good initial estimate in order to solve the nonlinear equations. A clear classification is given to classify all special Stewart Platforms that can be used in practice.

This research also develops a graphical robotic simulation module to model and simulate the General Stewart Platforms by creating an optimized object-oriented design module added to software designed by Ding [Ding, Z.Q., 2005]. The design approach implements all components in the Visual C++ programming language and freely distributed graphical library OpenGL, utilizing a single PC running the Windows operating system. The algorithm and the simulation module are demonstrated by two examples.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# ACKNOWLEDGEMENTS

I would like to express my sincerest and deepest appreciation to my supervisor Professor Waguih ElMaraghy for giving me the opportunity, his guidance, and for helping me throughout the course of my M.A.Sc program. I would also like to extend my thanks to my Supervisory Committee, Dr. Hoda ElMaraghy and Dr. Jonathan Wu for their comments and time in reviewing my thesis.

Within our excellent UROCA research team, I am very grateful to Ms. Zhongqing Ding, Ms. Ana M. Djuric and Dr. Yang Cao for their great work. They have helped me throughout my research and their efforts have let me not only work with my favourite research topic, but also made my work much easier.

I would like to acknowledge the great help and insight that I received from the IMS Centre directors, Professor Hoda ElMaraghy and Professor Waguih ElMaraghy, for giving me the chance to discuss the research topics with other members through the regular meetings they arranged for us. I learned a lot from these meetings. I would like to extend my thanks to all other members in IMS Centre for their suggestions and encouragement.

I would like to thank the Industrial and Manufacturing Systems Engineering Department staff: Ms. Jacquie Mummery, Mr. Ram Barakat and Ms. Zaina Batal for their support and kind assistance during my study period.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# TABLE OF CONTENTS

ABSTRACT .......... iii

ACKNOWLEDGEMENTS .......... iv

TABLE OF CONTENTS .......... v

LIST OF FIGURES .......... vii

LIST OF TABLES .......... ix

CHAPTER

I. INTRODUCTION

1.1 Kinematic Geometry of Mechanisms .......... 1
1.2 Introduction to Robotics .......... 4
1.3 Objectives and Contributions .......... 13
1.4 Thesis Overview .......... 15

II. REVIEW OF LITERATURE

2.1 Review of Kinematics of Stewart Platform .......... 16
2.2 Comparison with the Existing Approaches .......... 20
2.3 Review of Graphical Robotic Simulation Systems .......... 25

III. RECONFIGURABLE KINEMATICS OF STEWART PLATFORMS

3.1 Reconfigurable Kinematics Model .......... 31
3.2 Classification of Stewart Platforms .......... 43
3.3 Matlab-based Implementation .......... 49
3.4 Applications of Reconfigurable Kinematics Model .......... 51

IV. SIMULATION SOFTWARE DESIGN AND IMPLEMENTATION

4.1 Software background and Design requirements .......... 57
4.2 Software design and implementation .......... 58

V. APPLICATION EXAMPLES

5.1 3-3 Stewart Platforms .......... 70
5.2 5-4 Stewart Platforms (1-2-1-2) .......... 78

VI. CONCLUSION AND FUTURE WORK

6.1 Conclusion .......... 84
6.2 Future Work .......... 85

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

APPENDICES

SAMPLE MATLAB PROGRAM ........ 86
SAMPLE C++ PROGRAM ........ 89
USER MANUAL ........ 93

REFERENCES ........ 97

VITA AUCTORIS ........ 105

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## LIST OF FIGURES

**Figure 1.1:** Industrial robots doing vehicle underbody assembly........................................5

**Figure 1.2:** Cincinnati Milacron T Robot Arm........................................................6

**Figure 1.3:** An example of Stewart Platform........................................................9

**Figure 1.4:** Fully parallel Stewart-Gough platform................................................10

**Figure 1.5:** Fully parallel HEXA platform; all joints are revolute.................................10

**Figure 1.6:** Milling machine with a parallel manipulator design..................................12

**Figure 2.1:** Stewart Platform Mechanism...........................................................16

**Figure 2.2:** Stewart Platform with four extra sensors.............................................22

**Figure 3.1:** The general Stewart platform...........................................................31

**Figure 3.2:** Common Side Condition................................................................34

**Figure 3.3:** 2S Common Diagonal Condition........................................................36

**Figure 3.4:** 3S Common Diagonal Condition........................................................37

**Figure 3.5:** 0S Superposition Condition.............................................................39

**Figure 3.6:** The end-effector position and orientation of Stewart Platform.....................41

**Figure 3.7:** Applications of Stewart Platforms......................................................51

**Figure 3.8:** 3-3 Stewart Platform.....................................................................52

**Figure 3.9:** 6-3 Stewart Platform.....................................................................54

**Figure 3.10:** 6-3Equivalent Mechanism.............................................................55

**Figure 4.1:** The Menu Structure of GUI.............................................................61

**Figure 4.2:** The GSP Menu.............................................................................61

**Figure 4.3:** Graphical User Interface..................................................................62

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

**Figure 4.4:** GSP Toolbar Shortcut Buttons..........62

**Figure 4.5:** Moving Reference Coordinate System..........63

**Figure 4.6:** Overall System Structure..........64

**Figure 4.7:** Parameters Input Dialogs..........65

**Figure 4.8:** The Overall Coordinate Systems..........66

**Figure 4.9:** Pendent View Dialog..........68

**Figure 4.10:** Learn Pose Points View..........69

**Figure 5.1:** 3-3 Stewart Platform..........70

**Figure 5.2:** Kinematic Model of 3-3 Stewart Platform..........71

**Figure 5.3:** Parameters Input of 3-3 Stewart Platform..........76

**Figure 5.4:** The Inputs of Target Points(3-3SP)..........77

**Figure 5.5:** The Simulation Path(3-3SP)..........77

**Figure 5.6:** Kinematic Model of 3-3 Stewart Platform (5-4SP)..........78

**Figure 5.7:** The Simulation Path(5-4SP)..........83

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# LIST OF TABLES

**Table 2.1:** Matrix of Critical Literature Review........................................23

**Table 3.1:** Hexagon Base Topologies........................................44

**Table 3.2:** Pentagon Base Topologies........................................46

**Table 3.3:** Quadrangle Base Topologies........................................47

**Table 3.4:** Triangle Base Topologies........................................49

**Table 5.1:** The Definition of Target Points(3-3SP)........................................76

**Table 5.2:** The Definition of Target Points(5-4SP)........................................83

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER I

# INTRODUCTION

This chapter presents an introduction to the principal concepts and technologies involved throughout this research. The first section introduces kinematic geometry of mechanisms. In the second section some basic topics regarding to robotics are introduced. The third section overviews existing kinematics solving methods and graphical robotic simulation systems, with emphasis on their characteristics. Through this, the motivation for proposing a new algorithm and developing a unified robotic kinematic simulation interface is explained. The objectives of this research are also described at the same time. Finally, the fourth section presents an overview of this thesis.

## 1.1 Kinematic Geometry of Mechanisms

Basic geometry was developed by the ancient Greeks and Euclid's Elements was written as early as 300 BC. The foundations of algebra as we know it, on the other hand, were laid down much later—in the third century AD—and it was only after the development of calculus in the 17th century that the analytical study of mechanics became possible. The arguable preference of the algebraic over the geometric approach is not an issue of the past. The recent advent of the computer brought a revolution in mechanical design. While certainly the computer proved to be of great assistance to the engineer, it has also had negative effects on the readiness to seek deeper understanding of the principles of mechanical motion. This trend was quickly noticed and eloquently described by the two most famous advocates of kinematic geometry: [Joyce, D.E., 1997]

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

With a computer at his elbow an engineer is often tempted to pay little if any attention to principles, but rather plunge into a particular problem of synthesis without considering either the fundamental theory or the criteria that limit the performance of the devices he aims to produce. But more importantly the geometric principles reveal a map of a terrain, regions within which can then be explored in greater detail by analytical or graphical methods. If the map shows that there are inaccessible regions on the terrain, if it warns of hazards and dangerous frontiers, and if it can guide the explorer along safe paths by which he can reach his goal quickly with simple transport, then it should have some value [Hunt, K.H., 1978].

The digital computer demands on the part of its machine-designing users a ruthless competence in the algebraic processes needed for the manipulation of mechanical information and its numerical analysis. It is accordingly fashionable just now in the field of the theory of machines not so much to denigrate as simply to ignore the main bases in actual mechanical motion from which these algebraic processes grow. The main bases are essentially pictorial, geometrical. They arise from natural philosophy. Students in the mechanical sciences are becoming increasingly unable to contemplate a piece of ordinary reality in machinery accordingly, and to extract from that reality the geometric essence of it. It is of course true that without algebra there can be no programme, no numerical data, and no numerical result; but without an underlying geometry of the reality there can be no applicable algebra. Without a diagram we cannot write an equation. But without geometry we cannot even begin to draw [Phillips, J., 1984].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

So well have Profs. Kenneth Hunt and Jack Phillips warned against the treacherous trend of over-dependence on computer-based solutions. While the powerful programs for symbolic computations are undoubtedly helpful in design, they should be used only with complete understanding of their limitations (e.g., when dealing with trigonometric expressions). Paradoxically, it is exactly the development of the computer that has made geometry important again. As computers and automatic control algorithms have become more powerful, designs of increasingly complicated mechanisms have become practical. If prior to that, analytic methods were sufficient for the study of mechanisms, this was because these mechanisms were of outstanding simplicity. However, the complex spatial machines of nowadays can no longer be completely analysed by purely analytic or numerical methods. While most researchers were occupied developing or using computer-aided engineering tools, the two Australian professors, Kenneth Hunt and Jack Phillips, were among the few who realised the need for a revival of the geometric methods.

Kinematic geometry is the first and simplest segment of kinematics that deals exclusively with displacements. [Hunt, K.H., 1978] Time, as a variable, is usually not required to be brought into account. Indeed, the use of screw theory eliminates that need completely. Yet for convenience, velocity may sometimes be introduced in the study of the special, so-called singular, configurations of mechanisms. The main subject of this research is the position kinematics of parallel mechanisms or the geometry of two relatively moving bodies connected by a multitude of kinematic chains.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 1.2 Introduction to Robotics

The Robot Institute of America defined that "a robot is a reprogrammable multifunctional manipulator designed to move materials, parts, tools or specialized devices through variable programmed motions for the performance of a variety of tasks".

Robotics is concerned with the study of those machines that can replace human beings in the execution of a task with regards to both physical activity and decision making. Robotics is truly a multidisciplinary field that includes mechanical and electronic engineering, computer science, and mathematics.

### 1.2.1 Introduction to Industrial Robots

An industrial robot is officially defined by ISO as an automatically controlled, reprogrammable, multipurpose manipulator programmable in three or more axes. The field of industrial robotics may be more practically defined as the study, design and use of robot systems for manufacturing.

Typical applications of industrial robots include welding, painting, ironing, assembly, pick and place, palletizing, product inspection, and testing, all accomplished with high endurance, speed, and precision.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

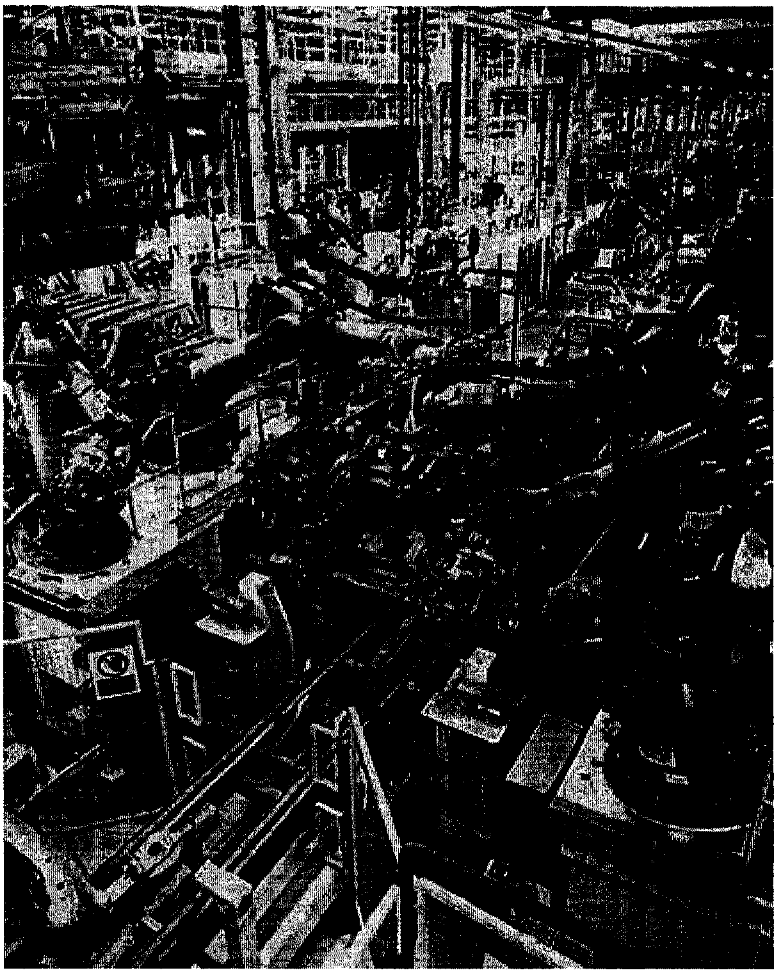


**Figure 1.1: Industrial robots doing vehicle underbody assembly (KUKA).**

**[Wikipedia, http://en.wikipedia.org/wiki/Industrial_robot]**

An industrial robot consists of:

- ✧ A manipulator
- ✧ Actuators
- ✧ Sensors
- ✧ A control system.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

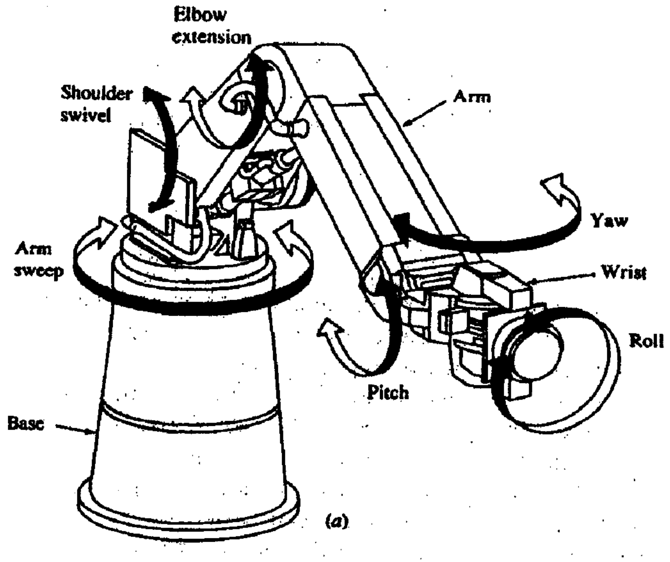


**Figure 1.2: Cincinnati Milacron T Robot Arm**

**[RISC lab, http://www1bpt.bridgeport.edu/~sobh/html/proj/sanjeev/project.html]**

A manipulator or mechanical structure consists of a sequence of rigid links connected by revolute or prismatic joints. Figure 1.2 illustrates an industrial robot manipulator. A manipulator has a supporting base, an arm that ensures mobility, a wrist that confers dexterity, and an end-effector that performs the desired task. The motion of the joints results in the relative motion of links.

### 1.2.2 Introduction to Parallel mechanism

The geometric approach used in this research has a wide application. Apart from Stewart Platforms, the approach may also be applied to the study of other parallel robots, to computer animation, and to many other fields. The field of parallel robots, despite the scarcity of specialised textbooks, is already too advanced to allow us to review it on a couple of pages. If the reader is, however, looking for a quick free overview of the field,

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

we suggest a visit to the on-line Parallel Mechanisms Information Center at http://www.parallemic.org.

An n-DOF (n-degree-of-freedom) fully-parallel mechanism is composed of n independent legs connecting the mobile platform to the base. Each of these legs is a serial kinematic chain that hosts one and only one motor which actuates, directly or indirectly, one of the joints. The variables that describe the actuated joints will be referred to as the input variables or also as the active joint variables. Other authors refer to the same variables as articular coordinates. On the other hand, the variables that describe fully the pose of the mobile platform (the end-effector) will be referred to as output variables. In other works, the same variables are referred to as generalised coordinates.

The configuration of an n-DOF parallel mechanism is not defined by its input variables. The task of finding the valid set of output variables corresponding to a set of input variables, referred to as the direct kinematic problem, has usually a multitude of solutions, referred to as assembly modes. In fact, some mechanisms allow an infinite number of solutions to their direct kinematics—a situation referred to as self motion [Karger, A., 1996]. More precisely, self motion means a finite mobility from some points of the workspace, whereas the confusingly similar term architecture singularity refers to a singularity in every point of the workspace [Ma, O., 1992]. When two, or more, of the assembly modes are coinciding, we say that there is a Type 2 singularity. The configuration of an n-DOF parallel mechanism is not even defined by both the input and output variables. Indeed, some mechanisms exist which will allow passive motion even when the motors and the mobile platform are fixed. Such particular singularities are

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

called Redundant Passive Motion (RPM) singularities [Zlatanov, D., 1994]. Most frequently, however, the user and the designer of a parallel mechanism will be interested only in the set of feasible output variables which we will refer to as the complete workspace. The complete workspace of a 6-DOF parallel manipulator is a six-dimensional highly coupled entity which is practically impossible to visualise. Therefore, the complete workspace of such mechanisms is studied only through its different subsets. Most of these are also defined for parallel mechanisms with less than six degrees of freedom. The most common subset of the complete workspace is the constant-orientation workspace which is the set of permissible positions for the centre of the mobile platform while the platform is kept at a constant orientation. Conversely, the orientation workspace is the set of permissible orientations of the mobile platform, while the platform centre is held fixed.

### 1.2.3 Introduction to Stewart-Gough Platform

A Stewart platform is a kind of manipulator using an octahedral assembly of struts. A Stewart platform has six degrees of freedom (x, y, z, pitch, roll, & yaw). There are six independently actuated legs, where the lengths of the legs are changed to position and orient the platform. The forward kinematics problem, an equation which given the leg lengths, finds the position and orientation of the platform, has 16 solutions. However, the the inverse kinematics problem (i.e. given the position and orientation of the platform, find the required leg lengths) has a unique and very simple solution.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Stewart platforms have applications in machine tool technology, crane technology, underwater research, air-to-sea rescue, flight simulation, satellite dish positioning, aircraft simulators and telescopes.

James S. Albus of the National Institute of Standards and Technology (NIST) has developed a crane, known as RoboCrane® [Albus, J.S., 1993], which uses the Stewart platform technology. Geodetic Technology trademarked "hexapod" for a Stewart platform in a machine tool context.

The Stewart platform was first reported in a paper by V. E. Gough in 1956 [Gough, V.E., 1956]. The name of Stewart was attached to this architecture because Gough's earlier work (and a photograph of his platform) was mentioned in the reviewers' remarks to a paper by D. Stewart published in 1965 [Stewart, D., 1965]; in that paper, Stewart presents another hybrid design, with three legs having two motors each.

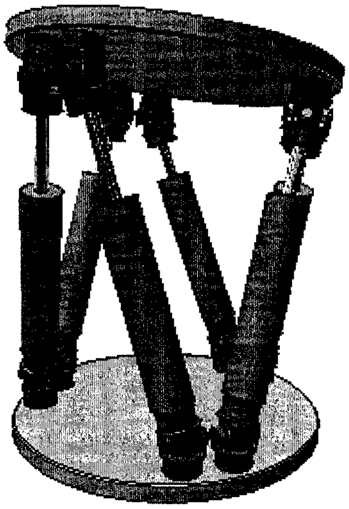


**Figure 1.3: An example of Stewart Platform**

**[Wikipedia, http://en.wikipedia.org/wiki/Stewart_platform]**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 1.2.4 Introduction to Kinematics of Parallel Robots

This section describes the kinematics of parallel robots, i.e., robots whose base and end-effector are connected by multiple serial chains in which not all joints are actuated. A fully parallel robot has six serial chains in parallel, and only one joint in each chain is actuated (Figure 1.4). Of course, all sorts of combinations of these purely serial and parallel structures are possible, and many exist in practice.

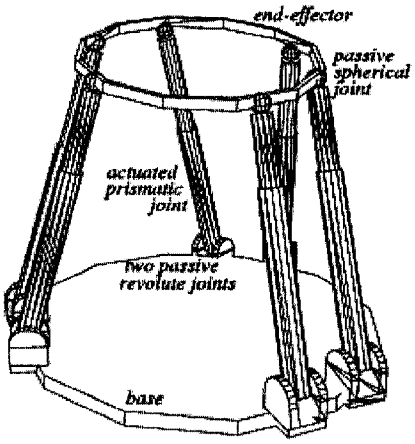


**Figure 1.4: Fully parallel Stewart-Gough platform;**

**[Bruyninckx, H., 2005, Parallel Robots, http://www.roble.info/robotics/parallel/]**

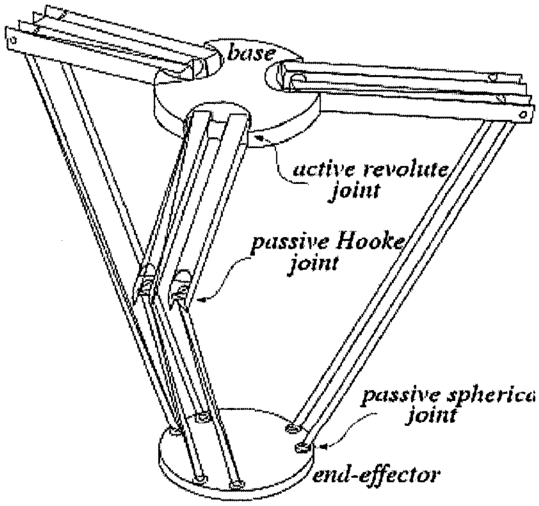


**Figure 1.5: Fully parallel HEXA platform; all joints are revolute.**

**[Bruyninckx, H., 2005, Parallel Robots, http://www.roble.info/robotics/parallel/]**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The main reasons for the overwhelming success of the serial robot design (over 99% of installed industrial robots) is that: It gives a large workspace compared to the space occupied by the robot itself; Kinematic designs exist that simplify the mathematics of the robot's geometry enormously.

The main drawback of a serial design is its low intrinsic rigidity, so that heavy links and joints must be used to obtain a reasonable effective rigidity at the end point. These pros and cons are exactly the opposites of those of parallel manipulators. The fully parallel designs of robots (Figure 1.5) have all actuators in or near the bases, which result in a very low inertia of the part of the robot that has actually to be moved. Hence, a higher bandwidth can be achieved with the same actuation power. This is why parallel structures are used for, for example, flight simulators and fast pick-and-place robots. A parallel structure supports its end-effector in multiple places, which yields a stiffer and hence more accurate manipulator for the same weight and cost, and which causes the positioning errors generated in each leg to "average out," again increasing the accuracy. This would be very advantageous for accurate milling (Figure 1.6). This pioneer was very unsuccessful in the marketplace. However, experiments with real prototypes show that parallel structures currently do not live up to these expectations: their accuracy and stiffness are about an order of magnitude worse than for classical serial machines. The reasons are:

- ✧ The compliance of the ball screws in the prismatic joints;
- ✧ The complexity of the construction with many passive joints that all have to be manufactured and assembled with strict tolerances;

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

- The complexity of kinematic calibration of this structure;
- The high forces that some passive joints have to resist.

In addition, another major disadvantage of parallel manipulators is their small workspace: legs can collide, and there are many passive joints in the structure that all introduce joint limit constraints. This is especially the case with the spherical "ball-in-socket" joints used in most implementations [Merlet, J.-P., 2006].

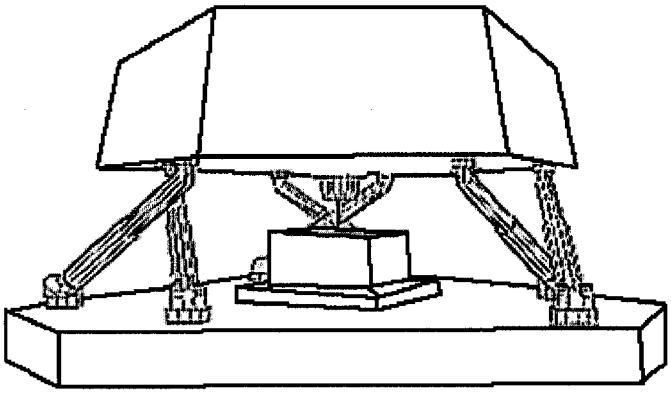


**Figure 1.6: Milling machine with a parallel manipulator design**

The definitions of forward and inverse position and velocity kinematics as defined for serial robots apply to parallel robots without change. But parallel robots have a large number of passive joints, whose only function is to provide the required number of degrees of freedom to each leg. Adding a leg between end-effector and base adds motion constraints to the end-effector, while in the case of serial robots adding a joint reduces the motion constraints (or, equivalently, adds a motion degree of freedom). This text discusses six degrees of freedom robots only, but many designs have less than six, e.g., planar [Kong, X., 2002] or spherical robots [Dafaoui, M., 1998].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 1.3 Objectives and Contributions

### Reconfigurable Kinematics of GSP

Most of researches nowadays are only focused on one specific or several similar Stewart Platforms. The algorithms they proposed are mostly Platform-Specific. An algorithm that can solve all the special cases of General Stewart Platform will make the analysis of GSP kinematics easier a lot and it also will make the software programming of GSP simulation possible. And the reconfigurable kinematics can help to pick up a best topology for a specific application as well.

### General Stewart Platform Simulation

A number of simulation software has been programmed to implement simulation of one or more particular platforms or products. If this simulation can be extended to general case it will be a great help for robot producers and researchers. So the development of a GUI, which provides 3D GSP kinematics modeling and simulation, is necessary. Our main objectives consist of the following two parts:

### Develop an algorithm to solve general Stewart Platforms

Utilizing the geometrical similarity of all different Stewart Platform, we can develop an algorithm with 6 variables (Struts' lengths) and 12 inputs (links' parameters) to solve general Stewart Platforms with slight changes of inputs.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

### Create software to simulate general Stewart Platforms

Create a programming strategy to embed the GUI of GSP simulation to our UROCA 6DOF serial robot simulation software. Design the GUI of GSP simulation based on the model developed in our first objective. Build DLLs to solve kinematics automatically and implement the GSP simulation.

## Contributions

In the past 40 years, much research and innovation has been made to parallel robots, their kinematics and simulation. Most of the improvements were focused on some special topologies, which have simpler structures or special characteristics making the real-time calculation of forward kinematics possible. The Stewart Platforms are the most widely used parallel mechanisms and they have most of the advantages of parallel mechanisms compared with serial robots and relatively simple structures.

In this research, the improvements have been made consist of two main parts, a new algorithm that can solve forward kinematics of General Stewart Platform and an upgraded software with the simulation function of General Stewart Platforms. In addition, a clear classification has been made to classify all special topologies of Stewart Platforms that can be used in practise.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 1.4 Thesis Overview

Chapter 1 provides an introduction to kinematic geometry, robotics, robotic kinematics, simulation platform, as well as the main technologies involved. Then the motivation and objectives of this research are presented.

Chapter 2 provides a systematic literature review, which consists of two parts, review of Stewart Platform Kinematics and review of Graphical Robotic Simulation System.

In Chapter 3, the details of the proposed algorithm for reconfigurable kinematics of General Stewart Platforms are introduced along with the Matlab implementation and the illustration of reconfigurable kinematics applications. Meantime, a clear classification of Stewart Platforms is proposed.

Chapter 4 introduces the software design procedure and illustrates software structure of the GUI, its main functions and an explanation of software implementation based on the theories in Chapter 1 previously.

Application examples are provided in Chapter 5 to illustrate the application of algorithm to a special topology and the simulation procedure.

The conclusion and future research are given in Chapter 6.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER II

# REVIEW OF LITERATURE

## 2.1 Review of Kinematics of Stewart Platform

The purpose of using flight simulators while training pilots is to minimize training losses and training time. A parallel mechanism placed under the simulator provides the translational and rotational movements that the pilot would be exposed to when flying with a real aircraft. Stewart was the first to bring up the idea of using parallel mechanisms in flight simulators [Stewart, D., 1965]. In the following years, parallel mechanisms have started being used commonly in various areas such as oil platforms and robotics.

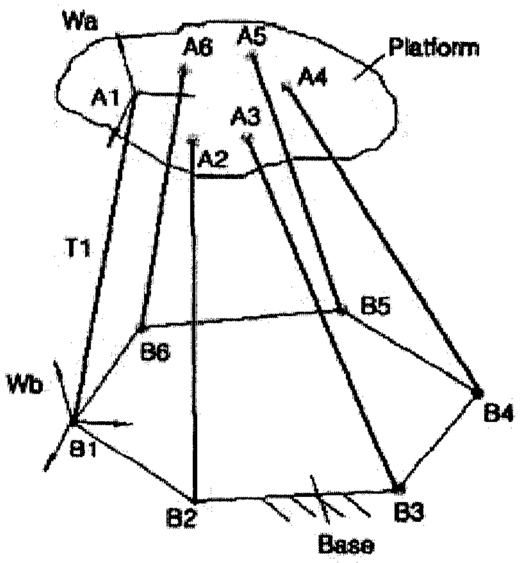


**Figure 2.1: Stewart Platform Mechanism**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Parallel mechanisms are kinematic chains with one or more closed loops and with actuators moving one or more of their links. Stewart Platform Mechanism (SPM), the most renowned parallel mechanism, consists of a fixed base platform and a movable platform, linked by 6 legs whose lengths can be changed via actuators on the base. [Figure 2.1]

The top platform has 6 degrees of freedom (DOF) of motion with respect to the base. If the legs are appointed certain fixed lengths, then the mechanism becomes a structure [Ku, D., 1999]. Traditional industrial robots are open-loop mechanisms with serial chains. Although serial chains reach further and have larger workspaces than parallel ones, parallel mechanisms have better dynamic characteristics. Serial mechanisms are not as rigid as parallel ones and have lower natural frequencies. Additionally, each link, starting from the one fixed to the base, until the tip link, must both be large enough to carry the preceding ones and provide the required accuracy. Another disadvantage of serial chains is that actuator errors add to one another, resulting in a huge error at the tip. Parallel mechanisms are preferred in applications where dynamic loading is high, speed and accuracy are important, and workspace volume is of less importance. In contrast to serial chains, not all the links are set into motion by the actuators, and moving parts are lighter, since the actuators are fixed to the base.

Due to their common application in aircraft simulators, a lot of work has been conducted on the SPM. Most often, one of the joints linking the legs to the platforms is a spherical joint while the other is a universal joint. Nomenclature of the SPM is done

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

according to its joint types and the number of links the SPM has. Even though there exist many papers in the literature on 3-3, 6-6 and 6-3 mechanisms, some researchers have come up with their own mechanism architectures in order to optimize certain criterion [Stoughton, R. S., and Arai, T., 1993].

The forward kinematics problem of the SPM is to determine the position and orientation of the top platform with respect to the base, when leg lengths are known. The inverse kinematics problem is to solve for the leg lengths that will result in a given position and orientation of the top platform. The forward kinematics problem has more than one solution, whereas the inverse kinematics problem has a single solution. Inverse kinematics has to be solved online for real-time trajectory tracking, and forward kinematics must be solved for real time control of the SPM [Innocenti, C., and Parenti-Castelli, V., 1990].

Lee and Shah [Lee, K. and Shah, D.K., 1988], who have shown that 6-dof parallel mechanisms can be composed by linking 3-dof mechanisms, have done the forward and inverse kinematic analysis of a 3-dof parallel mechanism. Nanua et al [Nanua, P., Waldron, K.J., and Murthy, V., 1990], have solved the forward kinematics problem of 6-3 SPM, reaching a 16th order polynomial, meaning the top platform can have 16 different configurations for a given set of leg lengths. Innocenti and Parenti-Castelli [Innocenti, C., and Parenti-Castelli, V., 1990], have solved the forward kinematics problem of the general SPM in closed form, repeating that the problem has 16 different solutions. Similarly, Wohlhart [Wohlhart, K., 1994] has worked on the forward kinematics of the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

spherical SPM, concluding that the problem has 16 solutions. Shi and Fenton [Shi, X. ve Fenton, R.G., 1992] have worked on the instantaeous kinematics of the SPM, resulting in 6 linear equations. Merlet [Merlet, J. P., 1993] has solved the forward kinematics of parallel mechanism in 4 different ways, and compared them on the grounds of computational time. Three of these methods are iterative methods and the fourth is the polynomial method. Although the computational time of the polynomial method is one order of magnitude greater than that of the other three, it is the only method that gives all solutions.

It must be kept in mind that not all of the solutions of the forward kinematics problem are physically feasible. When leg length constraints, joint rotation constraints and leg interference are taken into consideration, it will be seen that out of the 16 solutions of the forward kinematics problem, only 12 are feasible solutions [Yurt, S. N., 2002]. Liu et al [Liu, K., Fitzgerald, J. M., and Lewis, F. L., 1993] has put the forward kinematics problem of the SPM in the form of three nonlinear equations, which can only be solved by numerical schemes. With the method proposed in this work, the difficulty of solving a 16th order polynomial no longer exists. Sreenivasan, Husty and Innocenti [Sreenivasan, S.V., Waldron, K.J., and Nanua, P., 1994][Husty, M. L., 1996][Innocenti, C., 1998] have solved the forward kinematics problem of the 6-6 SPM using closed-form algebraic equations. Dasgupta and Mruthyunjaya [Dasgupta, B. and Mruthyunjaya, T.S., 1994] have solved the very same problem in canonic form. Ku [Ku, D., 1999] has solved the kinematics equations of the octahedral SPM that Nanua derived, using a simple and computationally efficient method based on Newton-Raphson's method. The aim of that

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

work was to reduce the computational effort that the polynomial method requires. Jakobovic and Jelenkovi [Jakobovic, D. and Jelenkovic, L., 2002] have stated that the forward kinematics problem can also be solved via optimization algorithms and that the solution converges when errors are allowed to be of the order of 10-12 times leg lengths.

In addition to the general SPM, there has been work done on mechanisms with special architectures. For example, Nanua [Nanua, P., Waldron, K.J., and Murthy, V., 1990] has solved the forward kinematics of 3-6/3-3 Stewart Platform. Tsai [Tsai, M. S., 2003] have solved the forward kinematics of the 3-PRS mechanism, while Kim and Park [Kim, J. and Park, F. C., 2001] have solved the forward kinematics of the 3-RS parallel mechanism., Di Gregorio [Di Gregorio, R., 2001] has solved the forward kinematics of the 3-URC wrist, Carretero [Carretero, J. A., 2000] has solved the forward and inverse kinematics of the 3-PRS mechanism, and Callegari and Tarantini [Callegari, M. and Tarantini, M., 2003] has solved the forward and inverse kinematics of the 3-RPC. Di Gregorio [Di Gregorio, R., 2002] has made proposals on purely translational parallel mechanisms, however they can not be used as flight simulators since they cannot simulate motion in six directions, meaning translation in three dimensions and rotation in three dimensions.

## 2.2 Comparison with the Existing Approaches

Basically, there are three approaches that are adopted to solve the forward kinematics of parallel robots, and they are univariate polynomial equation method, numerical iterative method and extra sensors method. Among these three methods, the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

univariate polynomial equation method is the only one that is able to solve the kinematics completely and get all the solutions for the forward kinematics but unfortunately there is no way to determine which solution is the current pose of the parallel robot. For numerical iterative method, there is no guarantee that the iteration is convergent. In the latest 10 years, the extra sensors method was proposed and implemented in researches and practice uses. The advantage of this method is that the forward kinematics can be obtained in short time without high performance computer required. This is the only method that can be used for real time applications. In the following three paragraphs, a brief introduction of these methods will be given [Karger, A., 1996][Merlet, J.-P., 2006].

Univariate polynomial equation method is to obtain a polynomial equation with a single variable by using some algebraic eliminating methods or geometrical eliminating methods so that the maximal number of solution can be easily told. It is worth noting that this study was started by Nanua and Waldron who determined a 24th order polynomial for the MSSM system [Nanua, P., Waldron, K.J., 1990][Husty, M. L., 1996][Innocenti, C., 1998][Wang, T. and Chen, C. C., 1993].

Numerical iterative method is to use numerical approaches to solve forward kinematics directly. The computation time is rather large and the convergence is not guaranteed; there is the problem of sorting the current pose from all the possible solutions, which has never been studied so far because of its complexity. We need a numerical method that produces the right solution in a reasonable time. All we concern are the methods efficiency and convergence. The drawback is that the obtained solution is not

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

necessarily the solution closest to the initial estimation, which will be the most likely the current pose of the end-effector [Huang, M.Z., 1996][Wang, T. and Chen, C. C., 1993].

Solving direct kinematics with extra sensors is adding extra sensors to the non-actuated joints to obtain information allowing fast calculation of the forward kinematics. What type of the extra sensors? Where are they placed? How many extra sensors the system needs? Rotation sensors are usually mounted on the base to avoid increasing moving platform weight. If three links are instrumented, which means adding six sensors, we are able to calculate the position of three points of the platform and hence solve the forward kinematics directly. Six sensors are enough to determine the pose of the moving platform by themselves [Bonev, I. A. and Ryu, J., 2000]. However it can be proved that only four extra sensors are necessary to determine the position and orientation of the end-effector. The difference here is that we still need equations to solve that unique solution according to the extra inputs from four extra sensors. The proof is illustrated by figure 2.2.

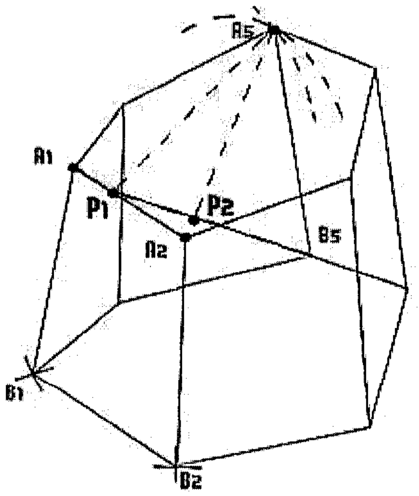


**Figure 2.2: Stewart Platform with four extra sensors**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

If two rotation sensors are instrumented on both B1 and B2 joints the position and orientation of A1 and A2 on the platform is determined. Let us pick up one of the left joints on the moving platform, for example A5. Apparently, it has to be following the trajectory of a circle C1, which is perpendicular to the fixed line A1A2 with the radius A5P1. At the same time, since the points P1 and the B5 are both fixed the point A5 should also follows the trajectory of the circle C2 perpendicular to the line P1B5 and the radius is A5P2, whose length is also fixed. Obviously the circles C1 and C2 can only has one single common point so actually the joint A5 is determined. The same theory applies to the rest joints on the moving platform. Hence the forward kinematics of Stewart Platform can be determined by four extra-sensors installed on two joints on base with 2-2 arrangement.

**Table 2.1: Matrix of Critical Literature Review**

| Ref. | Objective | Robot Type | Model Developing Methodology | Solution Technique | Fields Of Applications |
|---|---|---|---|---|---|
| Bonev, J., 2000 | Solving the Forward kinematics problem | 6-6 Stewart Platform with planar moving Platform | Three linear extra sensors on extra passive legs | Three linear extra sensors | Efficient and Suitable for real-time applications but extra structures are needed |
| Nanua, P., 1990 | Solving the Forward kinematics problem | 3-3/6-3 Stewart Platform with planar base and platform | Geometrical Method | 16th degree Univariate polynomial equation | All complex solution can be obtained but High performance computer is needed and the current pose is undecided |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

| | | | | | |
|---|---|---|---|---|---|
| Husty, M.L. 1996 | Solving the Forward kinematics problem | 6-6 Stewart Platform | Spatial kinematic mapping map three-dimensional motions into a seven-dimensional image space | 40th-degree Univariate polynomial equation | All complex solution can be obtained but High performance computer is needed and the current pose is undecided |
| Innocenti, C., 1998 | Solving the Forward kinematics problem | 6-6 Stewart Platform | Vector arithmetic | 40th-degree Univariate polynomial equation | All complex solution can be obtained but High performance computer is needed and the current pose is undecided |
| Wang, T., 1993 | Solving the Forward kinematics problem | General Parallel Robots | A cut-points method: use the depth first search Algorithm to transform the kinematic graph of the robot into a directed spanning tree and a set of cut joints. | Numerical iterative method The cyclic coordinate descent (CCD) method and Newton's method | Theoretically can be applied to all parallel robots but the algorithm is very complicated and the convergence is not guaranteed |
| Lee, T. Y. 2001 | Solving the Forward kinematics problem | 6-6 Stewart Platform | Vector arithmetic | Algebraic Elimination Method | Closed-form solution can be obtained but requires heavy computation and relative longer time. The current pose is unknown. |
| My thesis | Solving the Forward kinematics problem | General Stewart Platform | Geometrical Method | Four rotational sensors and Numerical iterative method | Theoretically can be applied to all Stewart Platforms, suitable for real-time application but only the current pose can be obtained and extra sensors are required |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The comparison Matrix [Table 2.1] below illustrates the advantages and disadvantages of different methods adopted to solve the forward kinematics of Stewart Platform. Most of research are focused on the platforms with six joints and three joints on the base so that most of Stewart Platforms with five and four joints on the base are not discussed before. I mark all these unsolved topologies with "No solution in published papers" in the category tables[3.1-3.4], including 6-4(2), 5-4(2), 5-3, 5-2, 4-4(2), 4-4(3), 4-3 and 4-2.

## 2.3 Review of Graphical Robotic Simulation Systems

Graphical robotic simulation and off-line programming of industrial robots are today relatively mature technologies. Robotic simulation software plays an important role in robotics research in many areas such as robot design, forward and inverse kinematics analysis, dynamics, control, path planning, etc. for both commercial and educational purposes. There are a lot of graphical robotic simulation software packages available in the market. Within this chapter, first, the common functionality implemented in the graphical robotic simulation software packages is described. Second, a literature review of these packages classified according to their control systems is presented. Meanwhile, characteristic analyses are provided for each category to summarize the discussed literature. Finally, the theories and technologies for developing a graphical robotic simulation system are explained.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

**Grasp2000** [BYG Systems Ltd], invented by BYG systems Ltd, is a true 3D simulation tool, based on accurate 3D geometry, process parameters and a library of industrial robots. Grasp2000 enables the creation of accurate 3D models, and real-time interactive simulations for cell layout design, planning, optimisation, and cycle time calculation. As a tool for off-line programming, the instructions can be automatically translated into the required native robot language.

Grasp2000 can generate specific application menus for arc welding, palletising and spraying. The software will find applications in PC-based cell layout and design, analysis, offline programming and process planning throughout the full range of Toshiba SCARA robot applications.

An important factor in off-line programming is the presence of inherent inaccuracies in most robots. Grasp2000 uses in-depth mathematical calculations to calibrate both the robot and 3D model to match the real world. It only requires the demonstration of a number of robot poses, which are then read into Grasp2000 and analyzed by the calibration software without external measuring equipment. An optional module for discrete event simulation extends Grasp2000's application areas to factory simulation, warehousing, logistics and materials handling.

**CSR** [Applied Computing & Engineering Ltd], is powerful 3D simulation software which enables manufacturing engineers to quickly simulate and evaluate automation concepts to determine the cost, feasibility and performance of a proposed robotic system.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Using existing in-house CAD data and AC&E's library of commercial robots and accessories to create a detailed simulation of the proposed manufacturing system, CSR accurately simulates interactions between work cell components to optimize equipment selection, fine-tune equipment positioning, and maximize production throughput.

The system is the most comprehensive and easy-to-use robotic simulation tool available and works completely off-line, eliminating the risk of damage to equipment and freeing robots for round-the-clock production. CSR can be purchased in a modular fashion to suit all budgets. Specialized application solutions tailored to the requirements of a particular robotic task provide advanced functionality and ease of use for painting, spot welding, arc welding, polishing, assembly and press operations.

**Interactive Graphics Robot Instruction Program (IGRIP)** owned by [DELMIA Corporation], is an interactive, 3D graphic simulation tool for designing, evaluating, and off-line programming robotic work cells. Actual robotic/device geometry, motion attributes, kinematics, dynamics, and I/O logic are incorporated to produce extremely accurate simulations. IGRIP optimizes critical factors such as robot motion planning, cycle time prediction, collision detection, calibration, and multiple I/O communication.

The several specific task software modules consist of UltraArc, UltraSpot, UltraPaint, and UltraFinishing, which are designed specially for arc welding, spot welding, painting, and surface finishing work cell applications respectively. Other applications include research and development, articulated design, flexible manufacturing system simulation, nuclear/hazardous duty automation, and general-purpose simulation.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Work cell components can be created in the integral CAD package or imported from other CAD packages via IGES, DXF, and direct translators. A built-in surface modeling package provides modification and/or optimization of imported surface data.

**EASY-ROB** [Anton, S., 2001], 3D Robot Simulation Tool was written in Visual C++ under the Windows operating system. In order to create high quality and high speed rendered images, the graphical capabilities of OpenGL are used. EASY-ROB is a complex and comprehensive modeling and simulation tool. It is especially designed to fulfill requirements for several industrial robotic applications as well as for educational purposes.

The EASY-ROB Basic Model allows the planning and designing of robotic work cell layouts consisting of a robot, tool and environment. A simple 3D CAD system is provided to create basic geometric parameterized primitives like cubes, cones, cylinders, pyramids, etc. In addition, a CAD interface is available to import other 3D formats such as STL. Created and imported geometries are assigned to the robot group to active or passive joints, to the tool group or to the environment group. Using a 3 button mouse, each geometry can be translated and rotated about its axis, or the operator can enter absolute or relative Cartesian values to set the Cartesian location. A modification of the view point (pan, tilt, zoom in and zoom out) in full shaded mode allows various world views.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The robot motion can be programmed using EASY-ROB standard program commands. A special Teach Window supports the user in writing robot motion programs. The built-in motion planner is implemented for the motion types, Point to point (PTP), Linear (LIN) and Circular (CIRC). The orientation interpolation for the LIN and CIRC motion type is realized for variable, fixed, tangential and quaternion modes. Several on-line output windows allow the operator to monitor robot joint values, Cartesian TCP location as well as simulation states such as cycle time, step size, override, etc. All data is saved into documented ASCII text files.

**Workspace** [Flow software technologies], described in [Owens, J., 1994], has been developed by a team led by John Owens as the world's first industrial robot simulation software package commercially released in 1989, and continuously updated over the last decade.

In addition to a library over 140 industrial robot 3D models available to the user, the 3D CAD modeler can create 3D solid objects using Constructive Solid Geometry and surface objects such as Bspline, Parametric, and Bezier surfaces. The 3D objects also can be imported from other CAD system via SXF or IGES file formats. The movement of any mechanism may be modeled using a kinematics modeler. The mechanism may have any number of joints in any serial or tree-structure combination. Conveyors, automatic vehicle, and other independently moving objects may also be modeled. Positions and paths for the robot tool to move to may be defined in several ways such as by use of the teach pendant, by clicking the mouse on the screen, or by using geometry points.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

It can be used to create and simulate robots in the native language of the robot. For example, users of Fanuc robots may write robot programs in Karel, ABB robot users may write programs in ARLA, or Visual Basic can be used just for simulation. Therefore, there is no need for translating simulation language to robot language. It is also possible to transfer existing robot programs from the robot control to Workspace for optimization.

In addition, the simulation can be replayed in real time. Calibration and dynamics modules are also available.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER III

# RECONFIGURABLE KINEMATICS OF STEWART PLATFORMS

## 3.1 Reconfigurable Kinematics Model

With reference to figure 3.1, input data to the forward kinematics of the general Stewart platform are the coordinates of the base attachment points Bi (i=1, . . . ,6) with respect to a reference frame Wb fixed to the base, the coordinates of the platform attachment points Ai (i=1, . . . ,6) with respect to a reference frame Wa fixed to the platform, and the actuator lengths Ti (i=1, . . . ,6). Without loss of generality, the origins of reference frames Wb and Wa are chosen at points B1 and A1 respectively.

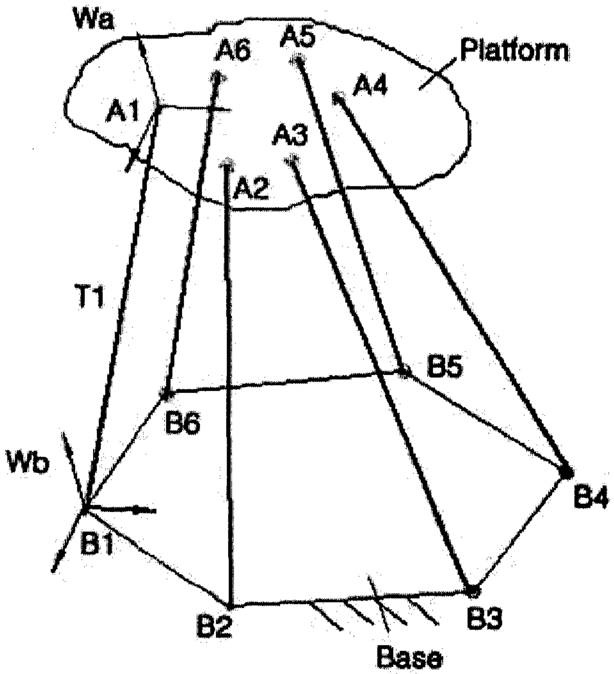


**Figure 3.1: The general Stewart platform**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

### 3.1.1 Reconfigurable Forward Kinematics

Based on the above defined inputs, we can get the follow list of inputs for the convenience of developing the kinematics model.

Inputs (i=1,2,…,6) (Link parameters)

- Base parameters: hexagon
  - Length of sides: Bi
  - Angles: $\beta$ i
- Moving platform parameters: hexagon
  - Length of sides: Ai
  - Angles: $\alpha$ i
- Length of struts: (actuated parameters): Ti

And the variables using in our model are two types of parameters which together determine the position and orientation of the platform.

Variables (i=1,2,…,6) (End effector's position and orientation);

- The length of struts' projection on base-plane $\rho$ i;
- $\gamma$ i : The angles between $\rho$i and Li on the base-plane.

The geometric-based approach we are using to solve the general Stewart Platform kinematics is projective geometrical method. Basically, the projection of each S joint

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

(vertex) on moving platform is used as an intermediate point and the distance between any two of them is used as an intermediate variable, which is one of the projections of sides and diagonals on the moving platform.

Since this is a geometrical method, we must include all the conditions that can be used to bring us to the solution and the projections we should consider consist of point projection and line projection. For line projection, there are two types of lines on the platform, they are the lines between adjacent joints (sides) and the line between non-adjacent joints (diagonals). For point projection, this only occurs when two or three joints are superposed and the point projection is the projection of coincident joint.

Based on the above analysis, the projection conditions are falling into three categories: common side (1S), common diagonal (2S and 3S) and superposition condition (0S) and the common diagonal conditions consist of two types of diagonals, which are named 2S and 3S diagonals.

### 1S Common side conditions

Simply speaking, the sides of polygon on the platform are used to be projected on the base in order to develop the relation equations between inputs and variables. The number of the generic conditions can vary from 1 to 6 in terms of the particular topology we are discussing [figure 3.2].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

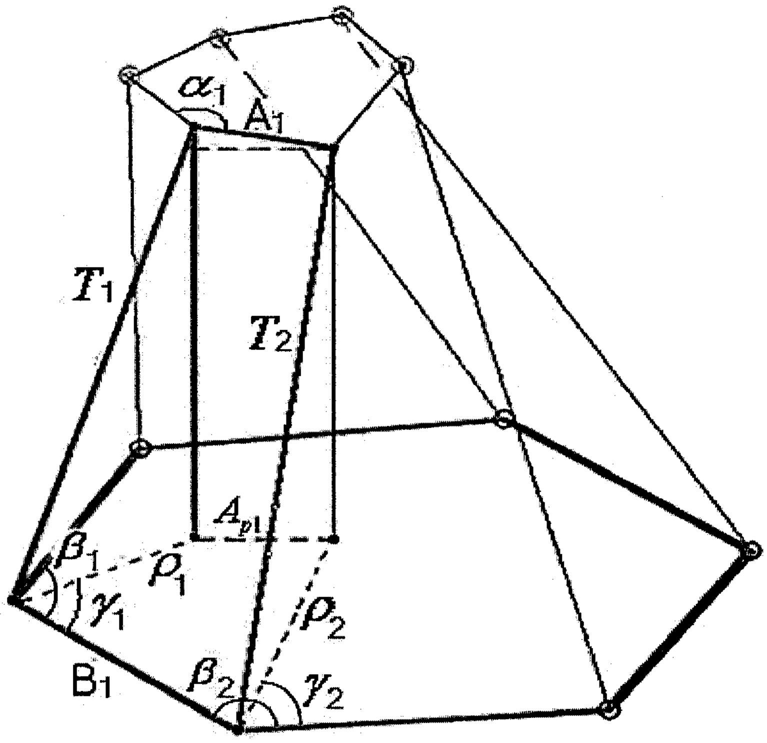


**Figure 3.2: Common Side Condition**

To develop the corresponding equations, we take a particular polyhedron out of the system to research. In the highlighted tetrahedron, A1 is the side we are using on the platform. The side (B1) on the base, the lengths of struts (T1 and T2) and the moving side (l1) are known. The variables in this polyhedron are the projections of struts (ρ1 and ρ2) and the angles between these struts' projections and the base side (γ1).

From top, one can develop:

$$l_1^2 - \left(\sqrt{T_1^2 - \rho_1^2} - \sqrt{T_2^2 - \rho_2^2}\right)^2 = l_{p1} \qquad (3.1)$$

From bottom, one can develop:

$$\left[\rho_1 \cdot \sin\gamma_1 - \rho_2 \cdot \sin\left(\beta - \gamma_2\right)\right]^2 + \left[L_1 - \rho_1 \cdot \cos\gamma_1 - \rho_2 \cdot \cos\left(\beta - \gamma_2\right)\right]^2 = l_{p1} \qquad (3.2)$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Combine and generalize:

$$l_i^2 - (\sqrt{T_i^2 - \rho_i^2} - \sqrt{T_{i+1}^2 - \rho_{i+1}^2})^2$$
$$= \left[\rho_i \cdot \sin\gamma_i - \rho_{i+1} \cdot \sin(\beta - \gamma_{i+1})\right]^2 + \left[L_i - \rho_i \cdot \cos\gamma_i - \rho_{i+1} \cdot \cos(\beta - \gamma_{i+1})\right]^2 \quad (3.3)$$

Let us set generalize the above equations as:

$$A_i^2 - (\sqrt{T_i^2 - \rho_i^2} - \sqrt{T_{i+}^2 - \rho_{i+}^2})^2$$
$$= [\rho_i \cdot \sin\phi_{i1} - \rho_{i+} \cdot \sin\phi_{i2}]^2 + [B_i - \rho_i \cdot \cos\phi_{i1} - \rho_{i+} \cdot \cos\phi_{i2}]^2 \quad (3.4)$$

Where:

$$A_i = l_i \quad (3.5)$$

$$B_i = L_i \quad (3.6)$$

$$\phi_{i1} = \gamma_i \quad (3.7)$$

$$\phi_{i2} = \beta_{i+1} - \gamma_{i+1} \quad (3.8)$$

$$i+ = i + 1 \quad (3.9)$$

This is the first set of equations that obtained by common side conditions with two types of variables: length ρ and angle γ. The number of these equations could be up to 6 in terms of the number of vertices on the platform.

### 2S Common diagonal conditions

Similarly, when we take a look at a particular highlighted tetrahedron, which has a side on platform acting as a diagonal [figure 3.3], we are going to use the 2S diagonal to develop the kinematic equations.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

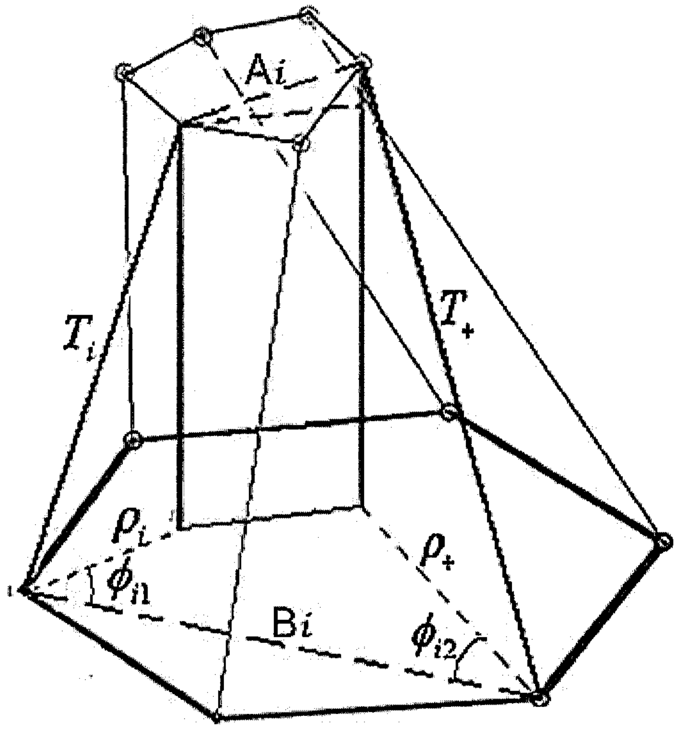


Figure 3.3: 2S Common Diagonal Condition

By the same way, the same set of equations in terms of common diagonal conditions can be developed as (3.4), however, different Ai, Bi, Φi1, Φi2 and i+ should be substituted in for two types of diagonals.

For those 2S diagonals, which are forming triangles with the corresponding 2 adjacent sides.

$$A_i = \sqrt{l_i^2 + l_{i+1}^2 - 2 \cdot l_i \cdot l_{i+1} \cdot \cos \alpha_{i+1}} \tag{3.15}$$

$$B_i = \sqrt{L_i^2 + L_{i+1}^2 - 2 \cdot L_i \cdot L_{i+1} \cdot \cos \beta_{i+1}} \tag{3.16}$$

$$\phi_{i1} = \gamma_i - \arctan\left(L_{i+1} \cdot \sin \beta_{i+1} / \left(L_i - L_{i+1} \cdot \cos \beta_{i+1}\right)\right) \tag{3.17}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$\phi_{i2} = \beta_{i+1} + \beta_{i+2} - \gamma_{i+2} + \arctan\left(L_{i+1} \cdot \sin\beta_{i+1} / \left(L_i - L_{i+1} \cdot \cos\beta_{i+1}\right)\right) - \pi \tag{3.18}$$

$$i+ = i+2 \tag{3.19}$$

### 3S Common diagonal conditions

For those 3S diagonals, which are forming quadrangles with the corresponding 3 adjacent sides [figure 3.4].

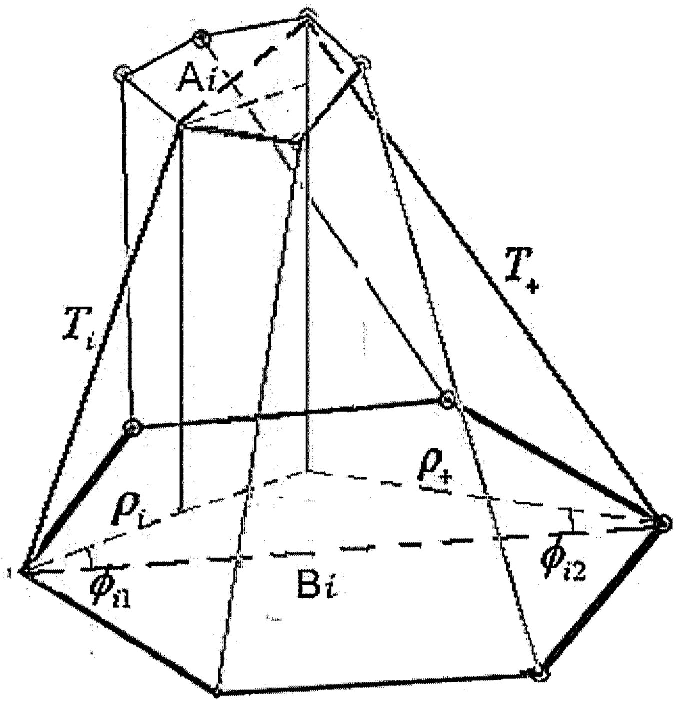


Figure 3.4: 3S Common Diagonal Condition

$$A_i = \left[l_i^2 + l_{i+1}^2 + l_{i+2}^2 + 2 \cdot l_i \cdot l_{i+2} \cdot \cos(\alpha_{i+1} + \alpha_{i+2}) - 2 \cdot l_i \cdot l_{i+1} l_{i+2} \cdot \cos\alpha_{i+1} \cdot \cos\alpha_{i+2}\right]^{1/2} \tag{3.20}$$

$$B_i = \left[L_i^2 + L_{i+1}^2 + L_{i+2}^2 + 2 \cdot L_i \cdot L_{i+2} \cdot \cos(\beta_{i+1} + \beta_{i+2}) - 2 \cdot L_i \cdot L_{i+1} L_{i+2} \cdot \cos\beta_{i+1} \cdot \cos\beta_{i+2}\right]^{1/2} \tag{3.21}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$\phi_{i1} = \left| \arcsin\left( \frac{\left(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2}\cos\beta_{i+2}\right)}{\sqrt{\left(L_{i+1} - L_i\cos\beta_{i+1} - L_{i+2}\cos\beta_{i+2}\right)^2 + \left(L_{i+2}\sin\beta_{i+2} - L_i \sin\beta_{i+1}\right)^2}} \right) - \beta_{i+1} + \pi/2 - \gamma_i \right| \tag{3.22}$$

$$\phi_{i2} = \left| \arcsin\left( \frac{\left(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2}\cos\beta_{i+2}\right)}{\sqrt{\left(L_{i+1} - L_i\cos\beta_{i+1} - L_{i+2}\cos\beta_{i+2}\right)^2 + \left(L_{i+2}\sin\beta_{i+2} - L_i \sin\beta_{i+1}\right)^2}} \right) + \beta_{i+2} + \beta_{i+3} - \gamma_{i+3} - {}^{3\pi}\!/_{2} \right| \tag{3.23}$$

$$i+ = i+3 \tag{3.24}$$

**Superposition conditions**

We can image that in some cases two or more joints may superposed together. At this time, another kind of conditions should be taken into account: superposition conditions because these are also the necessary conditions that determine the position and orientation of end-effector [Figure 3.5].

The condition equations' development is illustrated by an example here. When the first two joints on the moving platform are coincident with each other, we take the corresponding tetrahedron (with sides B1, T1 and T2) out of the system to research.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

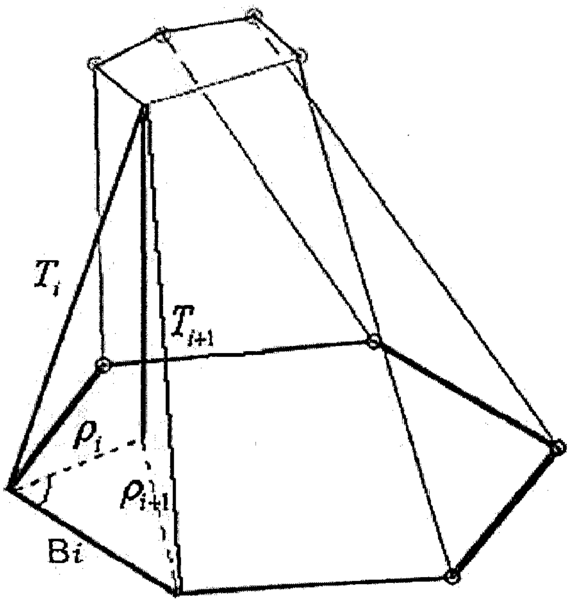


Figure 3.5: 0S Superposition Condition

This condition is simpler comparing with the former two, we use the projection line as common side the following equations can be developed:

$$T_i^2 - \rho_i^2 = T_{i+1}^2 - \rho_{i+1}^2 \tag{3.25}$$

When n joints on the moving platform are superposed, n-1 equations can be developed. Based on the above mentioned three conditions, up to 15 equations can be derived in terms of them and for those applicable topologies at least 12 equations can be derived. The nonlinear kinematics model can be derived as following:

$$\begin{aligned} &A_i^2 - (\sqrt{T_i^2 - \rho_i^2} - \sqrt{T_{i+}^2 - \rho_{i+}^2})^2 \\ &= [\rho_i \cdot \sin\phi_{i1} - \rho_{i+} \cdot \sin\phi_{i2}]^2 + [B_i - \rho_i \cdot \cos\phi_{i1} - \rho_{i+1} \cdot \cos\phi_{i2}]^2 \end{aligned} \tag{3.26}$$

$$T_i^2 - \rho_i^2 = T_{i+1}^2 - \rho_{i+1}^2 \tag{3.27}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

To solve the above equations, as the motion is continuous, at each calculation step, an initial estimate is the posture at the previous step. The convergence problem can be solved mainly by choosing a very small time step, which require higher computing capability and is not suitable for real-time applications. By using the extra sensors, we can use the redundant data input from sensors to determine uniquely the solution of the forward kinematics and to provide a good initial estimate for the iterative solving process. Another advantage of the extra sensors is that they can be used for robot self-calibration and also ensure the robot reliability in the case of a sensor failure. The choice of type and disposition of extra sensors is aimed at obtaining the unique solution to the forward kinematics with a minimal number of sensors. The sensors we are going to use are rotary sensors for measuring the leg direction and it has been proved that at least four rotary sensors are need for obtaining a unique solution, which means two of joints on base should be measured by four rotary sensors and each two sensors is assigned to get direction of the base universal joint on both axes so that the position of corresponding two joints on the moving platform are given, which are four adjacent variables for the above equations.

### 3.1.2 Reconfigurable Inverse Kinematics

For inverse kinematics, the objective is to solve the six struts' lengths with given end-effector position and orientation.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

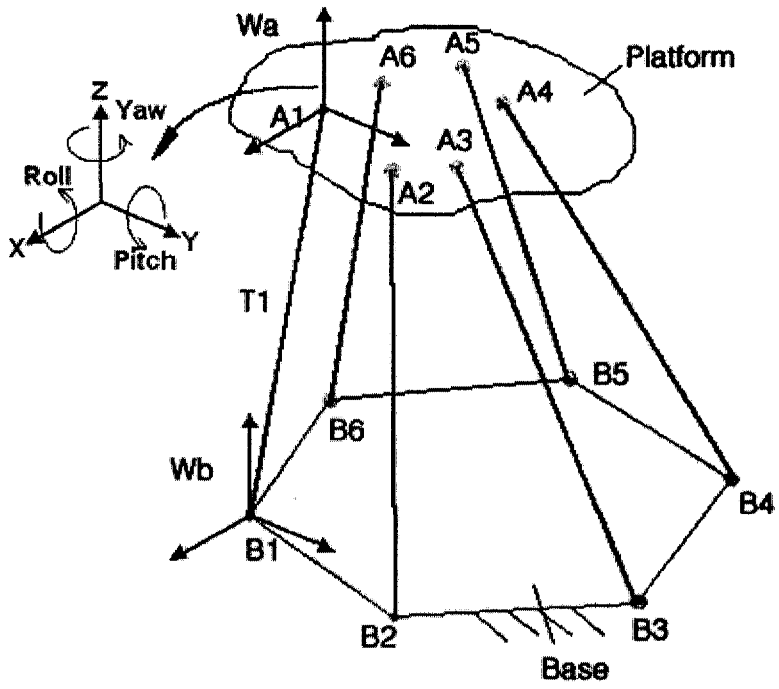


**Figure 3.6: the end-effector position and orientation of Stewart Platform**

With reference to the Figure 3.6, the given inputs are the coordinates of the end-effector and the orientation of the end-effector, which are roll, pitch and yaw. Other known parameters are the shape of moving platform and base. In order to solve the length of six struts, the coordinates of the joints on the moving platform must be solved first. With the given initial joint coordinates of the moving platform, the transformation matrix can be adopted here to obtain the coordinates of the joints on the moving platform. The translational and rotational matrices used in the process are shown below.

The translational matrix along axis x of the moving coordinate frame:

$$\text{Tx\_matrix} = \begin{bmatrix} 1 & 0 & 0 & Tx \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.28}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The translational matrix along axis y of the moving coordinate frame:

$$\text{Ty\_matrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & Ty \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.29}$$

The translational matrix along axis z of the moving coordinate frame:

$$\text{Tz\_matrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & Tz \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.30}$$

The rotational matrix about axis x of the moving coordinate frame:

$$\text{Troll\_matrix} = \begin{bmatrix} \text{Cos(Troll)} & \text{-Sin(Troll)} & 0 & 0 \\ \text{Sin(Troll)} & \text{Cos(Troll)} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.31}$$

The rotational matrix about axis y of the moving coordinate frame:

$$\text{Tpitch\_matrix} = \begin{bmatrix} \text{Cos(Tpitch)} & 0 & \text{Sin(Tpitch)} & 0 \\ 0 & 1 & 0 & 0 \\ \text{-Sin(Tpitch)} & 0 & \text{Cos(Tpitch)} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.32}$$

The rotational matrix about axis z of the moving coordinate frame:

$$\text{Tyaw\_matrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \text{Cos(Tyaw)} & \text{-Sin(Tyaw)} & 0 \\ 0 & \text{Sin(Tyaw)} & \text{Cos(Tyaw)} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.33}$$

The product of the above matrix is the transformation matrix:

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$\text{Ttrans\_matrix} = \text{Tx\_matrix} \cdot \text{Ty\_matrix} \cdot \text{Tz\_matrix} \cdot \text{Troll\_matrix} \cdot \text{Tpitch\_matrix} \cdot \text{Tyaw\_matrix} \tag{3.34}$$

By multiplying the transformation matrix with the known initial coordinates of the moving joints, we can obtain coordinates of the joints on the moving platform. Let's say the given coordinates matrix of the moving joints is movePM0, then the final coordinates matrix of the moving joints is:

$$\text{movePM2} = \text{Ttrans\_matrix} \cdot \text{movePM0} \tag{3.35}$$

Assume the given base coordinates matrix is basePM, then the struts length matrix can be solved as:

$$\text{Strutlen} = \left|\text{movePM2 - basePM}\right| \tag{3.36}$$

## 3.2 Classification of Stewart Platforms

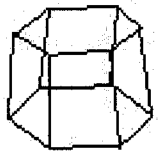
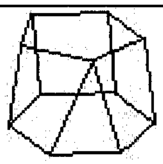
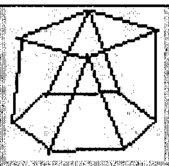
We classify different categories of Stewart platform based on the shape formed by the rotational joints on base and moving platform respectively. There are five types of geometric shapes: hexagon, pentagon, quadrangle, triangle and line. I exclude the special type of shape, point, which leads to all the rotational joints coincident together on platform or base, because this situation reduces the degree of freedom of the system to zero, which is meaningless.

For the base, first four shapes are practicable but the line shape is unusable for its un-stability. Considering the hexagon on the base [Table 3.1], all types of moving

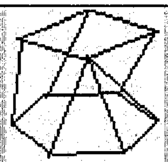
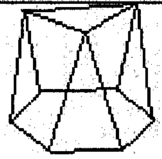
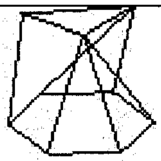
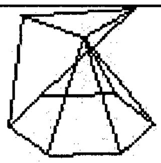
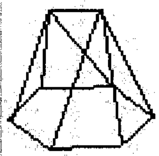
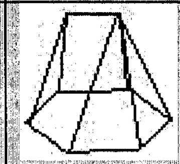
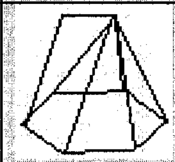
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

platform shapes can be applied to it. Considering the pentagon base [Table 3.2], hexagon moving platform has the exactly same calculating condition as the 6-6 type so it will not be mentioned here and the same for all those types with more joints on moving platform. For the quadrangle base [Table 3.3], there are five available topologies are practicable. The last category, triangle base [Table 3.4], contains only one usable topology, 3-3 SP, with every adjacent joint couple coincident together. The ones with Fs under the numbers of types are unpractical. F1 stands for the ones with more than 4 joints intersecting together; F2 is for the ones with legs interfering with each others; F3 is for the ones with unstable structures.

**Table 3.1: Hexagon Base Topologies**

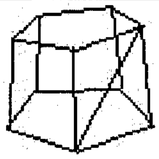
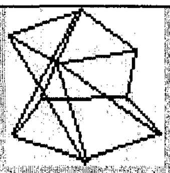
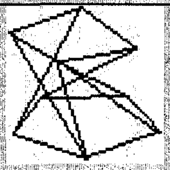
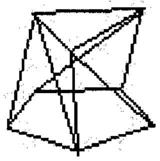
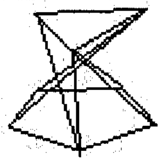
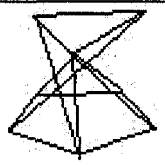
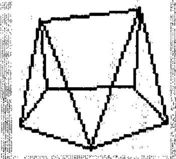
| Conf | Conditions | Demo | Applications |
|---|---|---|---|
| 6-6 | Com:[1s] 1/2, 2/3, 3/4, 4/5, 5/6, 6/1<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: None |  | COPRA® Hexapod<br>Lens Hexapod<br>Servos motion system<br>PI Hexapod |
| 6-5 | Com:[1s] 2/3, 3/4, 4/5, 5/6, 6/1<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2 |  | |
| 6-4(1) | Com:[1s] 2/3, 3/4, 5/6, 6/1<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 4/5 |  | |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

| | | | |
|---|---|---|---|
| 6-4(2) | Com:[1s] 3/4, 4/5, 5/6, 6/1<br>[2s] 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3 |  | No solution in published papers |
| 6-3(1) | Com:[1s] 2/3, 4/5, 6/1<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 3/4, 5/6 |  | Fanuc F-200iB<br>Moog E-Cue 660<br>AI Motion Bases |
| 6-3(2) | Com:[1s] 3/4, 5/6, 6/1<br>[2s] 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3, 4/5 |  | |
| 6-3(3)<br>F1 | Com:[1s] 4/5, 5/6, 6/1<br>[2s] 3/5, 4/6, 5/1, 6/2<br>[3s] 2/5, 3/6<br>Coi: 1/2, 2/3, 3/4 |  | N/A |
| 6-2(1) | Com:[1s] 3/4, 6/1<br>[2s] 2/4, 3/5, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3, 4/5, 5/6 |  | |
| 6-2(2)<br>F1 | Com:[1s] 4/5, 6/1<br>[2s] 4/6, 6/2<br>[3s] 3/6<br>Coi: 1/2, 2/3, 3/4,4/5 |  | N/A |
| 6-2(3)<br>F1 | Com:[1s] 5/6, 6/1<br>[2s] 3/5, 4/6, 5/1, 6/2<br>[3s] 2/5, 3/6<br>Coi: 1/2, 2/3, 3/4, 5/6 |  | N/A |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Table 3.2: Pentagon Base Topologies

| Conf | Conditions | Demo | Applications |
|---|---|---|---|
| 5-5 | Com:[1s] 2/3, 3/4, 4/5, 5/6<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2 |  | |
| 5-4(1) | Com:[1s] 2/3, 3/4, 5/6<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 4/5 |  | No solution in published papers |
| 5-4(2) | Com:[1s] 3/4, 4/5, 5/6<br>[2s] 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3 |  | No solution in published papers |
| 5-3(1) | Com:[1s] 2/3, 4/5<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 3/4, 5/6 |  | No solution in published papers |
| 5-3(2) | Com:[1s] 3/4, 5/6<br>[2s] 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3, 4/5 |  | No solution in published papers |
| 5-3(3)<br>F1 | Com:[1s] 4/5, 5/6<br>[2s] 3/5, 4/6, 5/1, 6/2<br>[3s] 2/5, 3/6<br>Coi: 1/2, 2/3, 3/4 |  | N/A |
| 5-2(1) | Com:[1s] 3/4<br>[2s] 2/4, 3/5, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6 |  | No solution in published papers |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

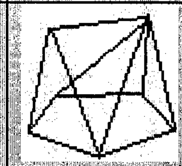
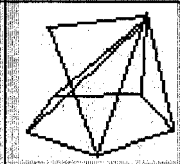
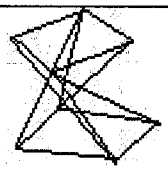
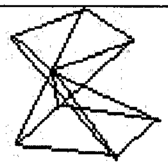
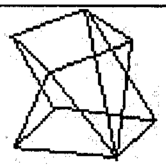
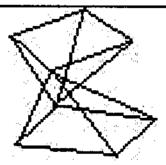
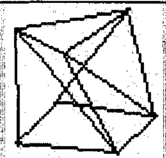
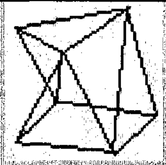
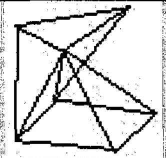
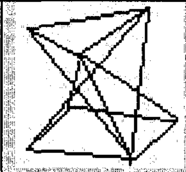
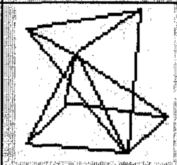
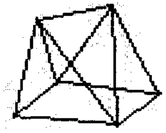
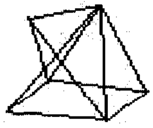
| | | | |
|---|---|---|---|
| | Coi: 1/2, 2/3, 4/5, 5/6 | | |
| 5-2(2)<br>F1 | Com:[1s] 4/5<br>[2s] 3/5, 4/6, 5/1, 6/2<br>[3s] 2/5, 3/6<br>Coi: 1/2, 2/3, 3/4, 5/6 |  | N/A |
| 5-2(3)<br>F1 | Com:[1s] 5/6<br>[2s] 4/6, 6/2<br>[3s] 3/6<br>Coi: 1/2, 2/3, 3/4,4/5 |  | N/A |

**Table 3.3: Quadrangle Base Topologies**

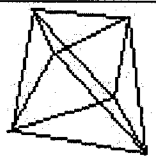
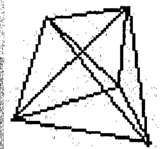
| Conf | Conditions | Demo | Applications |
|---|---|---|---|
| 4-4(1) | Com:[1s] 2/3, 5/6<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 4/5 |  | |
| 4-4(2) | Com:[1s] 4/5, 5/6<br>[2s] 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3 |  | No solution in published papers |
| 4-4(3) | Com:[1s] 5/6, 6/1<br>[2s] 1/3, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 4/5 |  | No solution in published papers |
| 4-4(4)<br>F3 | Com:[1s] 3/4, 6/1<br>[2s] 2/4, 3/5, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6 |  | N/A |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

| | | | |
|---|---|---|---|
| | Coi: 1/2, 2/3 | | |
| 4-3(1) | Com:[1s] 4/5<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 3/4, 5/6 |  | No solution in published papers |
| 4-3(2) | Com:[1s] 5/6<br>[2s] 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3, 4/5 |  | No solution in published papers |
| 4-3(3)<br>F1 | Com:[1s] 5/6<br>[2s] 3/5, 4/6, 5/1, 6/2<br>[3s] 2/5, 3/6<br>Coi: 1/2, 2/3, 3/4 |  | N/A |
| 4-3(4)<br>F2 | Com:[1s] 3/4<br>[2s] 2/4, 3/5, 4/6, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3, 4/5 |  | N/A |
| 4-3(5)<br>F1 | Com:[1s] 6/1<br>[2s] 3/5, 5/1, 6/2<br>[3s] 2/5, 3/6<br>Coi: 1/2, 2/3, 3/4 |  | N/A |
| 4-2(1)<br>F | Com:[1s] N/A<br>[2s] 2/4, 3/5, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 2/3, 4/5, 5/6 |  | N/A |
| 4-2(3)<br>F1 | Com:[1s] 4/5, 5/6<br>[2s] 2/4, 6/2<br>[3s] 1/4, 3/6<br>Coi: 1/2, 2/3 |  | N/A |

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

**Table 3.4: Triangle Base Topologies**

| Conf | Conditions | Demo | Applications |
|---|---|---|---|
| 3-3 | Use of conditions:<br>Com:[1s] N/A<br>[2s] 1/3, 2/4, 3/5, 4/6, 5/1, 6/2<br>[3s] 1/4, 2/5, 3/6<br>Coi: 1/2, 3/4, 5/6 |  | FCS E-Cue 624-1800<br>E-Cue 660-12000<br>MOOG 6DOF2000E<br>PI M-850<br>AI Hexapod |
| 3-2<br>F2 | Use of conditions:<br>Com:[1s] 1/2, 3/4, 5/6<br>[2s] None<br>[3s] 1/4, 2/5, 3/6<br>Coi: None |  | N/A |

According to the above mentioned tables, up to 20 different topologies can be used for industrial application theoretically. In the following section, the solving method will be presented for all of these topologies and for each of topology a pick-up algorithm is shown in the corresponding table row to choose those proper conditions for a specific topology.

## 3.3 Matlab-based Implementation

Matlab [Mathwork] is an interactive environment for mathematical and scientific computing. It is the standard tool for numerical computing in industry and research. Matlab stands for Matrix Laboratory. It specializes in Matrix and vector computations,

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

but includes functions for graphics, numerical integration and differentiation, solving differential equations, etc. Matlab differs from most significantly from, say, Maple, by not having a facility for abstract computation.

In addition to functions for numerical linear algebra, Matlab provides functions for the solution of a number of common problems, such as numerical integration, initial value problems in ordinary differential equations, root-finding, and optimization. In addition, optional "Toolboxes" provide a variety of such functions aimed at a particular type of computation, for example, optimization, spline approximation, signal processing, and so forth. The optional toolboxes are not the main point here but the solving nonlinear problems.

More details about the following commands may be obtained from the help command:

- fzero root-finding (single variable);
- fmin nonlinear minimization (single variable);
- fmins nonlinear minimization (several variables);
- fsolve solves systems of nonlinear equations of several variables.

Matlab is powerful math computing tool but it still has some fatal limitations, which are insufficient user interface, poor debug environment and so on. A short program was

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

coded to implement and solve the kinematic model with all necessary inputs and outputs displayed. The code example is shown in APPENDIX A.

This code example just shows how simple it could be to program a Matlab program in order to implement the mathematical model and it is not adopted for simulation because Visual C++ is another more powerful way to implement the modelling and simulation process, which is the way we are using in the coming chapters. [Mark, S., 1999]

## 3.4 Applications of Reconfigurable Kinematics Model

Stewart Platforms are the most commonly used parallel robot in industrial applications and they can be applied in many different fields, such as motion simulators, milling machines, auto test rigs, positioning systems and so on. Figure 3.7 shows some of applications existing in real industrial applications.

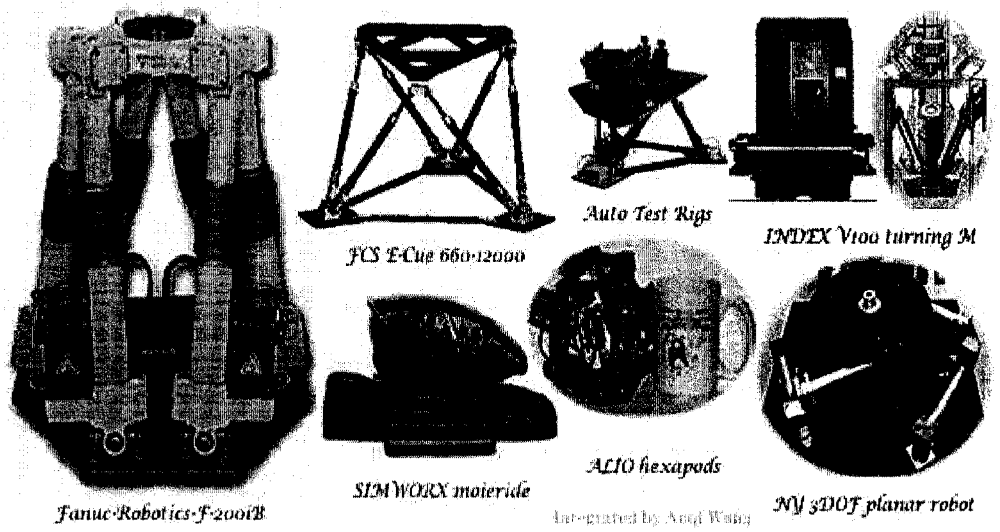


Figure 3.7: Applications of Stewart Platforms

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

As we discussed, the kinematics model is derived for general case so that this model can be applied to any special application by slightly change some inputs. In the following section, two configurations are discussed to illustrate how and why this model can be applied to special cases.

### 3-3 Stewart Platform

One of the most famous Stewart Platform is the 3-3 Stewart Platform and it is applied in many industrial fields and for miscellaneous purposes, such as flight simulator FCS E-Cue 624-1800 and E-Cue 660-12000, motion base MOOG 6DOF2000E, positioning PI M-850, multiple-application AI Hexapod and so on. The configurations used for above-mentioned applications are specified in Table3.1-Table3.4.

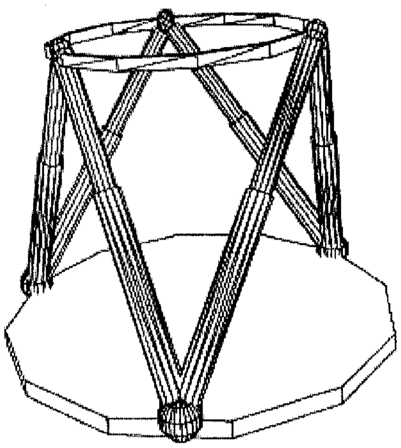


**Figure 3.8: 3-3 Stewart Platform**

**[Bruyninckx, H., 2005, Parallel Robots, http://www.roble.info/robotics/parallel/]**

As shown in figure 3.8, in 3-3 Stewart Platform three pairs of joints are superposed respectively on both base and moving platform. This configuration is widely used in

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

industry because its triangle structures make it stronger and more rigid. Another major advantage is that it is easier to be configured and controlled. Our kinematics model can be easily applied in this case by changing the inputs on base and moving platform and the detail is explained as following.

The inputs we are talking about are actually the link parameters, which determine the special configuration of the topology. These inputs include the moving platform parameters: the lengths of sides and the angles of adjacent sides, which form the moving platform, and the similar base parameters. The inputs are listed as following:

Inputs (i=1,2,⋯,6) (Link parameters)

- ✧ Base parameters: hexagon
  - Length of sides: Li
  - Angles:βi
- ✧ Moving platform parameters: hexagon
  - Length of sides: li
  - Angles:αi

When these parameters are changed a new topology is obtained. For example, when we let L1, L3 and L5 equal to zero and let l2, l4 and l6 equal to zero too, a 3-3 Stewart Platform is formed. So we can solve the 3-3 Stewart platform by changing the above parameters with no consideration of changing kinematics model.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

### 6-3/6-6 Stewart Platform

6-3/6-6 Stewart Platform is another popular used topology and many researchers have done lots of work on solving the kinematics, singularity and workspace. Ku D.M [Ku, D.M., 2000] and Akcali ID [Akcali, I.D., Mutlu, H., 2006] proposed effective and efficient approach to solve forward kinematics and provide some numerical examples. Yanwen Li [Li ,Y., Huang, Z., Chen, L., 2003] studied the singularity of the 6-3 Stewart Platform in 2003. A program 6p-3 [Merlet, J.P., 1992] was designed to compute the solutions of the forward kinematics of 6-3 Stewart Platform with a planar base [figure 3.8]. 6-3 Stewart Platforms are used in Funuc F-200iB, Moog E-Cue 660, AI Motion Bases etc. and 6-6 Stewart Platforms are used in COPRA® Hexapod, Lens Hexapod, Servos motion system, PI Hexapod and so on. The configurations used for above-mentioned applications are specified in Table3.1-Table3.4.

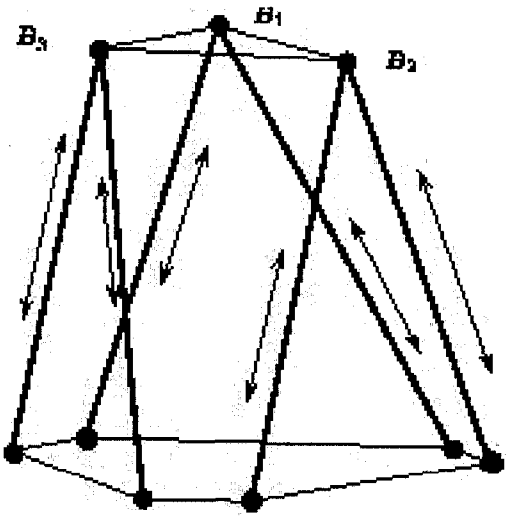


**Figure 3.9: 6-3 Stewart Platform**

**[Merlet, J-P., 1992, 6p-3, http://www-sop.inria.fr/coprin/logiciels/RP/FK/6p-3/notice-html.html]**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

For the forward kinematics the mechanism is equivalent to a 3-(RS) mechanism. The 3-(RS) mechanism is constituted of a base and a moving platform, which is a triangle. Each vertex of this triangle is connected to the ground by a fixed length link which is attached to the platform by a ball-and-socket joint. The other extremity of the link is connected to the ground through a revolute joint [figure 3.10].

The difference between 6-3 Stewart Platform and 6-6 general Stewart Platform can be simply figured out that the three pair of joints on moving platform are superposed and there are two struts are connecting to each vertex of the triangle on moving platform. So for instance if we set l1, l3 and l5 equal to zero at the same time we can have a 6-3 Stewart Platform. By the same way can we use the same reconfigurable kinematics model to solve this special topology.

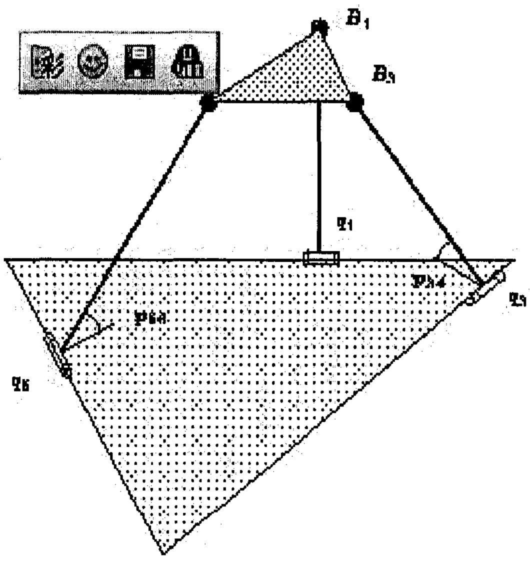


**Figure 3.10: 6-3Equivalent Mechanism**

**[Merlet, J-P., 1992, 6p-3, http://www-sop.inria.fr/coprin/logiciels/RP/FK/6p-3/notice-html.html]**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## Other applications

A 3-2-1 Stewart Platform was studied as early as 1994 by Geng, Z.[Geng, Z., and Haynes, L., 1994] and he provided an application to six degrees of freedom pose measurements. 6-4 Stewart Platforms are studied by Chen N.X. and Song S.M. [Chen N.X. and Song S.M., 1992 and 1994] and they presented a forward kinematics analysis for 6-4 Stewart Platform. Husain, M. and Herman Bruyninckx [Husain, M. and Waldron, K.J., 1994 and Herman Bruyninckx,1998] proposed an algorithm to solve the forward kinematics of 3-1-1-1 Stewart Platform. In 1992, Knapczyk J., Dzierzek S.[Knapczyk, J. and Dzierzek, S., 1992] and Nielsen J., Roth B.[Nielsen, J. and Roth, B., 1996] studied the 6-5 Stewart Platform and gave its forward kinematics solutions. In 1990, Lin W. and Duffy J.[Lin, W., Duffy, J. and Griffis M., 1990] published a paper talking about the kinematics of 4-4 Stewart Platform and two years later they gave the forward kinematics solution of 4-5 Stewart Platform[Lin, W., Crane, C.D. and Duffy, J., 1992].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER IV

# SIMULATION SOFTWARE DESIGN AND IMPLEMENTATION

## 4.1 Software background and Design requirements

Unified Kinematic Modeler and Simulator (UKMS) was designed by Zhongqing Ding in 2005 [Ding, Z.Q. and ElMaraghy, W.H., 2005]. UKMS applied the unified solution for Puma serial robots proposed by Dr. Djuric and Dr. ElMaraghy [Djuric, A.M. and ElMaraghy, W.H, 2004] and hence implement modeling and simulating of Puma type serial robots. Nowadays, Parallel robots are attracting growing attention in various fields and the star is absolutely Stewart Platform, which is widely used in industrial application, machine tools, positioning device and other miscellaneous applications. More than 100 laboratories and R&D centers around the world are focusing research on it. Most of famous robot and machine tool manufacturers including Fanuc, ABB and Adept are currently supplying some parallel robot products.

Although Stewart Platform has brought a lot of attention to researchers and manufacturers, the software platforms that can model and simulate Stewart Platforms are still too few compared with the wide spread of Stewart Platforms. SEMORS-PKM designed by I-Ming Chen is used to Simulate Modular Reconfigurable Robotic Systems based on Product-Of-Exponential formulation. It can be run on Windows desktop operating system and do the singularity analysis and kinematic simulation. RoboWork is another software tool that can be use to model parallel robot. It provides seven shapes of

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

objects (cylinder, cone, disk, annular disk, sphere, cube and wedge), five transformations (rotation, translation, scaling, transformation start and transformation stop) and several materials, group trees and on/off switch. Unfortunately, the functions of RoboWork are too simple to run kinematics simulation.

My main objectives of the proposed research is not only to provide a GUI software module to implement the functionality described in former section, but also to create a software platform, which has the features of easy of use, extensibility, portability, and reusability. And these functions of modeling and simulating Stewart Platforms are going to be integrated into our UKMS to make the software stronger and multi-functional. The theory and specification used to build the software are the same in order to make the software has solid stability.

## 4.2 Software design and implementation

### Introduction of Robot Modeling and Simulation

Geometrical modeling of robot manipulators is an expanding area of research because it can aid in the design and usage of robots in a number of ways [Mirolo, C. and Pagello, E., 1989]:

- ✧ Design and testing of manipulators: The purpose of the modeling is to study different approaches to satisfy the design specifications of the manipulator

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

- Robot action planning: The modeling environment is used to build a representation of the robots, positioners, and other mechanisms with moving joints, and the objects in the workspace for creating and validating action plans by simulating the effect of these actions in the model space.
- On-line control of robot manipulators: The simulated action plans generated in the model space are transmitted (after validation) to the attached robot manipulators for execution.
- Training and education: Robotics simulation packages provide an inexpensive and safe way to teach the theory and operation of robot manipulators.
- Tele-robotic user interface: In applications where the operator of the robot has to be at a large distance from the workcell (radiation, space, etc.) realistic graphical simulation can be used for better interaction with the manipulator.

To satisfy all or some of the above goals, a robotics modeling and simulation package has to provide the following minimal fixtures:

- A way to build models of solid objects: This requires the facilities to create a set of solid primitives ((like boxes. cylinders, cones, etc.) and some services to combine these into more complex shapes.
- A way to assemble models of robot manipulator links and other solid objects into complete models of robotic work cells.
- A way to manipulate the models. This includes methods fix accessing and animating objects, routines for forward and inverse kinematics calculations, and graphics display.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Advanced robot simulation environments can also support one or more of the following:

- Path planning services for moving the manipulator along various trajectories (straight-line, etc).
- Collision detection and collision avoidance services.
- Simulation of manipulator dynamics to obtain the forces and torques on the links of the manipulator arm and to enforce their limits.

### Manu structure and Graphical user interface

The developed user menu of the software is interactive for the construction of kinematic modeling and simulation for both industrial serial and parallel robots. The software allows the user to create kinematic models and implement the simulation of robots by using graphical user dialogs which have several sub-tab to perform different tasks. The default supervising GUI, which is loaded automatically at the startup of the system, is a Multiple Document Interface (MDI) application. The menu structure of GUI is shown in figure 4.1

The new menu added for General Stewart Platform modeling and simulation is the GSP menu, which includes three submenus: model, pendent view and simulation. The menu is shown in figure 4.2.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

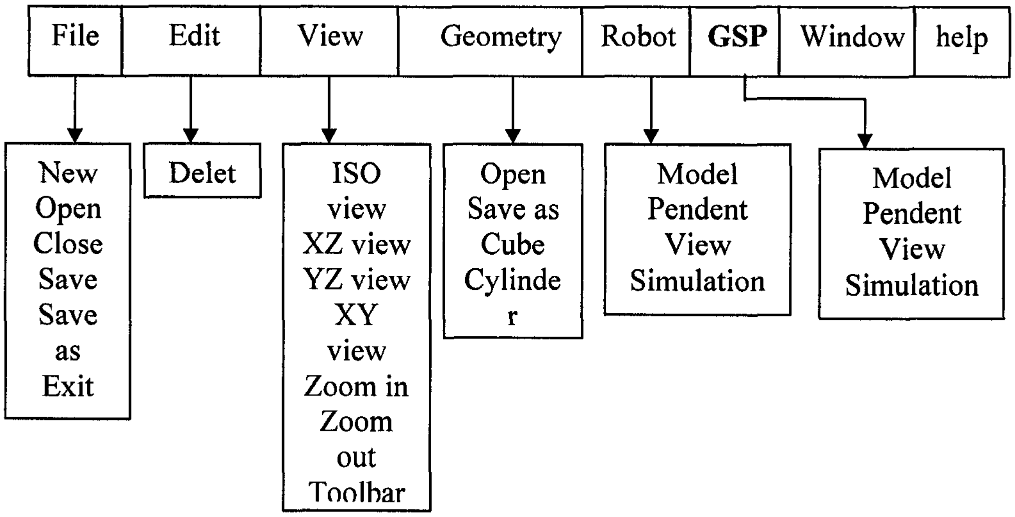


**Figure 4.1: The Menu Structure of GUI**

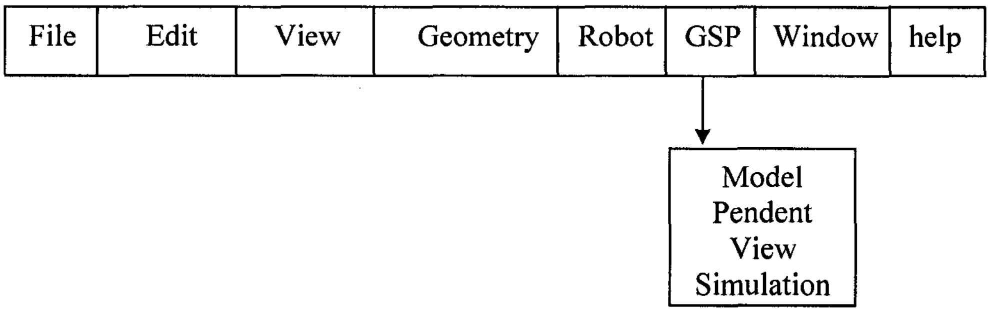


**Figure 4.2: The GSP Menu**

The default window is shown in figure 4.3. The window comprises:

- A Menu Bar that contains all command menus and options.
- A Toolbar that contains icons for the most commonly used options.
- A working area which is a 3D scene viewer showing images of the work cell objects.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

- A Status Bar that includes 5 columns for displaying the status message, the selected object name, the origin coordinate X, Y, Z values of the selected object respectively.

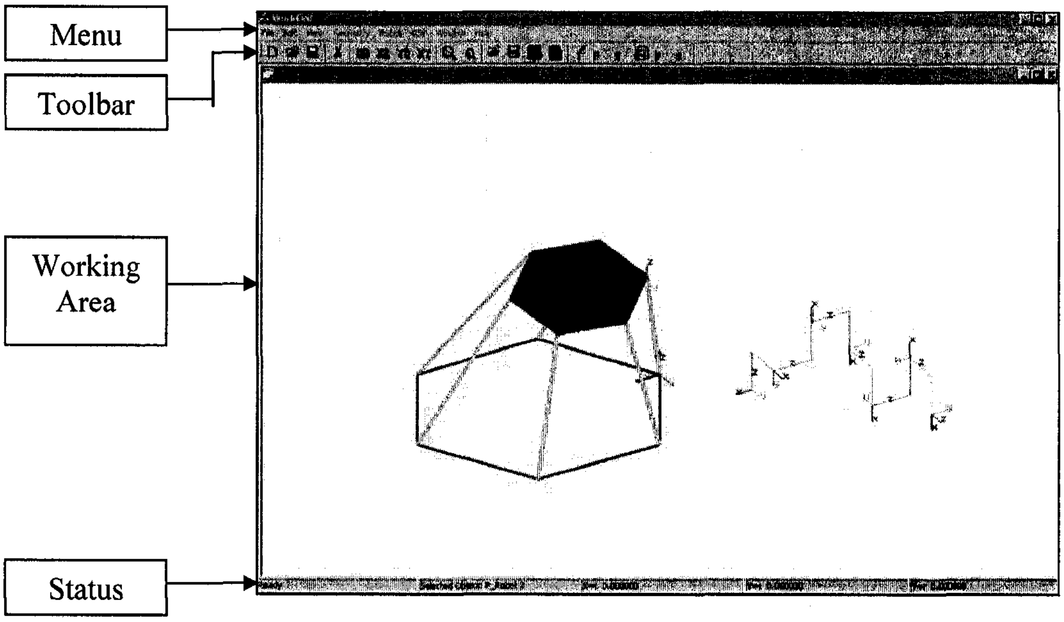


**Figure 4.3: Graphical User Interface**

The new introduced shortcut buttons on the toolbar are the three blue buttons that act as the shortcuts with the same functions with the corresponding menus. These three shortcut buttons are shown in figure 4.4.

**Figure 4.4: GSP Toolbar Shortcut Buttons**

The world coordinate system known as the world coordinate system and the platform coordinate system known as the moving reference coordinate system both adopt

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

the right-hand rectangular Cartesian coordinate system regulation which specifies the x, y, and z axes are in the same relative orientation as forefinger, second finger, and thumb respectively of a right hand shown in figure 4.5

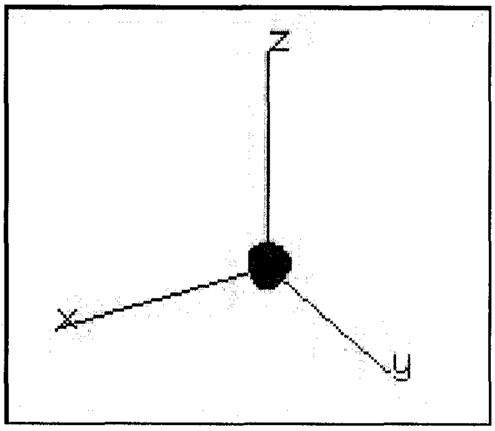


**Figure 4.5: Moving Reference Coordinate System**

### Creation of Models: the System Modeling Environment

UKMS models the three dimensional geometric objects using lists of their bounding polygons in a manner similar to other solid modeling software tools. It can model and simulate two major branches of industrial robots, which are the Puma type robots (designed by Zhongqing Ding [Ding, Z.Q. and ElMaraghy, W.H., 2005]) and Stewart Platforms. The system supports a GUI developed and run under Windows-based operating system, which makes it easy to port this system to PC-based robot control systems. The overall system structure is illustrated in Figure 4.6.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

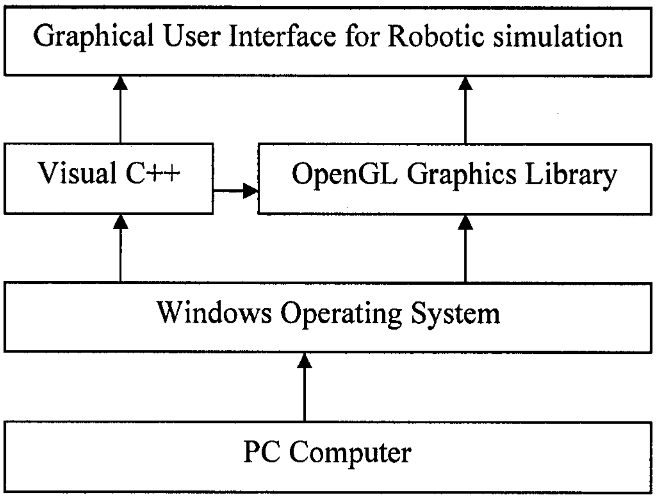


**Figure 4.6: Overall System Structure**

**[Ding, Z.Q. and ElMaraghy, W.H., 2005]**

**The procedure of GSP modeling is as following:**

By analyzing the kinematic structures of General Stewart Platform, we give the input parameters to determine the topology of the Stewart Platform. By simply clicking on the model sub-menu of the GSP menu, inputting the necessary link parameters, we are able to obtain the model view in the work area [Figure 4.7].

- Create the origin coordinate of moving platform with matrix translation. There is no rotational translation for the moving platform origin.
- Calculate the joint coordinates on moving platform and create the model.
- Change model by using different link parameters to form all different Stewart Platforms.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

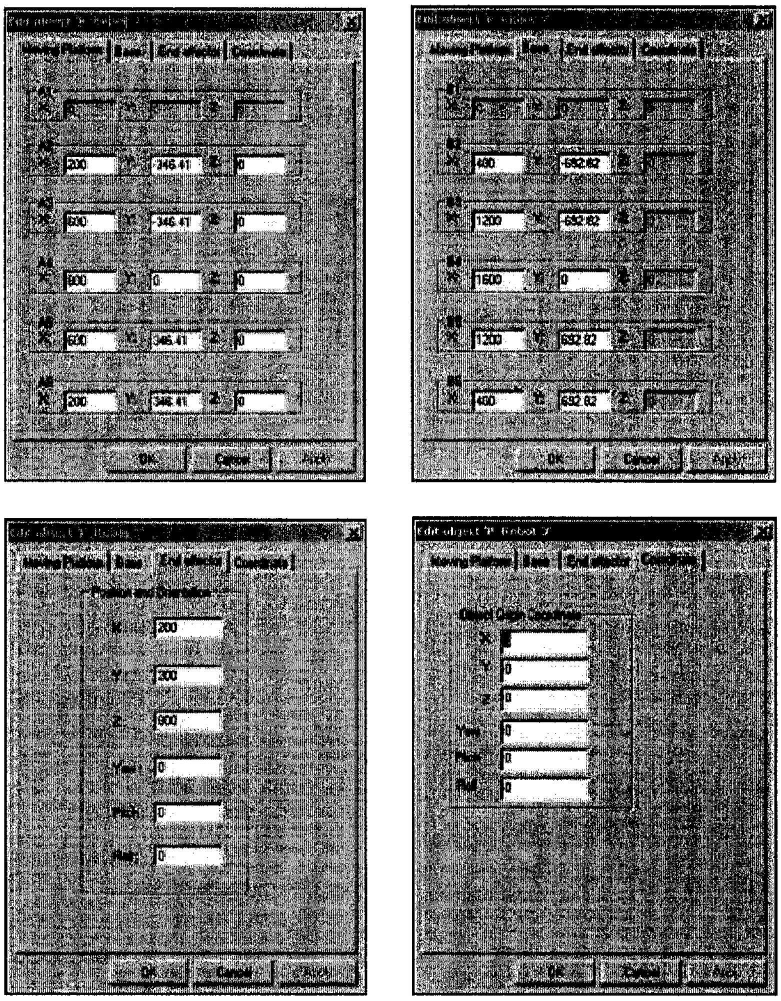


**Figure 4.7: Parameters Input Dialogs**

The coordinate systems used in the modeling system are shown in figure 4.8. There are two coordinate systems are assigned to the objects, the base coordinate system which is fixed and the moving platform coordinate system which is moving. The origin of base coordinate system is fixed on the first joint of the base. The origin of moving platform is fixed on the first joint of the moving platform and moving with the platform. The absolute coordinates of moving platform coordinate frame is the position of the first joint of moving platform in base coordinate system [Figure 4.8].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

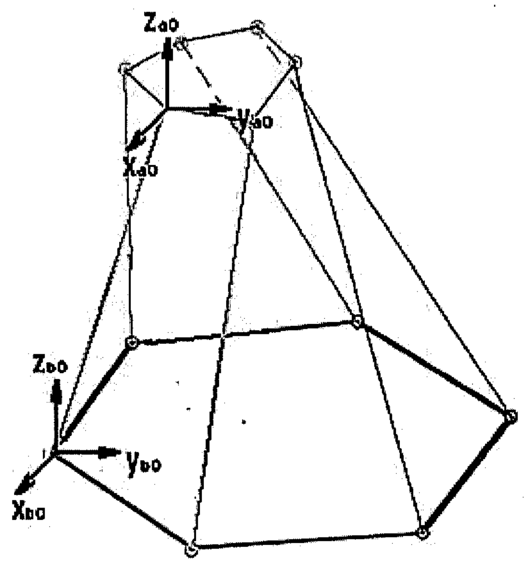


Figure 4.8: The Overall Coordinate Systems

## Operating the Models: the System Simulation Environment

The required robot path is given by a set of points. Those points are target locations of the robot's end-effector and are called Target Points (TPs). Each point is defined with its position and orientation. We need to calculate robot joint values for each point depending on the position and orientation of the point.

A path is a list of all the TPs that a robot follows during a sequence of motions. We can generate a TP using the Learn TP button on the pendent view. After the user has created the path, the simulation function can be used to simulate the activity of the robot following the path.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The pendent view is divided into three sections. The top section is labeled "end-effector" and displays the absolute position and orientation of the tool frame of the robot with regard to the base coordinate frame. The second section is labeled "Strut lengths" and shows the lengths of the six extendable struts. The third section includes two operating button. The Learn TP button learns and saves the current data as a Target Point (TP). The Home button moves the robot to the initial position. The robot will be moving to the location of solution when it is selected.

If the user changes the position and orientation of end-effector in the first section, the robot will immediately move according to the given values. Meanwhile, the inverse kinematic problem will be solved, and the corresponding struts' lengths will be shown in the second section.

The UKMS provides an interactive simulation environment where the configuration inputs entered by the user are immediately executed and the results are displayed on a graphics screen. From this interactive environment users can change the simulation scenario and operate the robot manipulator models in the system. The procedure of simulation can be described as follows:

Select the object that we want to simulate and click the pendent view sub-menu of GSP menu. The pendent view dialog will be popping up [Figure 4.9].

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Click on the up/down arrow in the end-effector area or just type the value to give the position and orientation of the end-effector. Once a parking point is selected we click on the "learn TP" button on the bottom of pendent view dialog to let the system learn the robot pose at this point [Figure 4.9].

After all necessary points on the trajectory are given, we can click the OK button to finish the learning section and proceed to simulation section by choosing the simulate menu in the GSP main menu and then the system will exactly follow the trajectory that you taught to show the simulation of the platform moving [Figure 4.10].

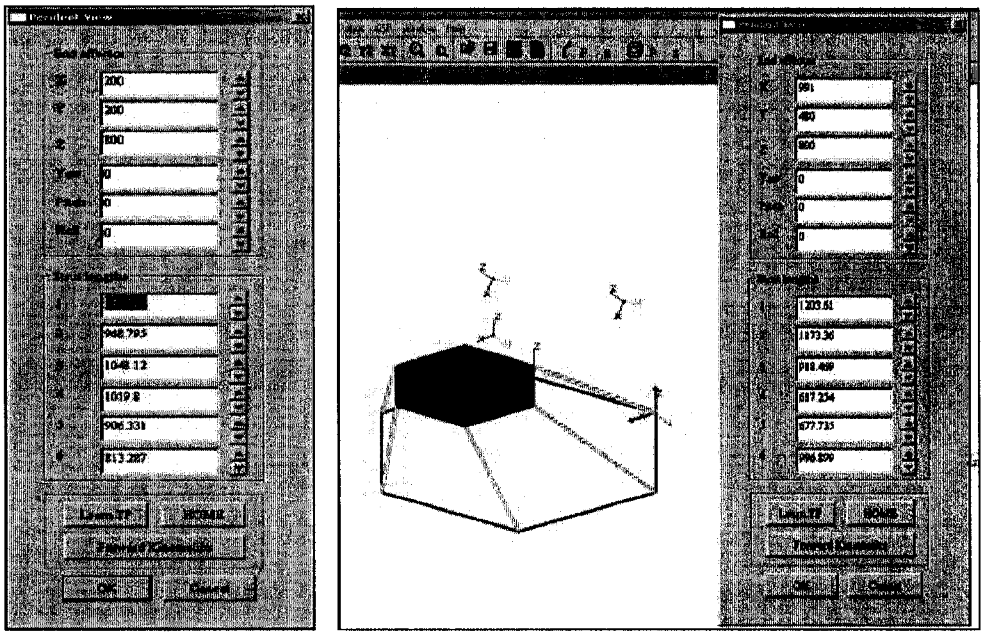


**Figure 4.9: Pendent View Dialog**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

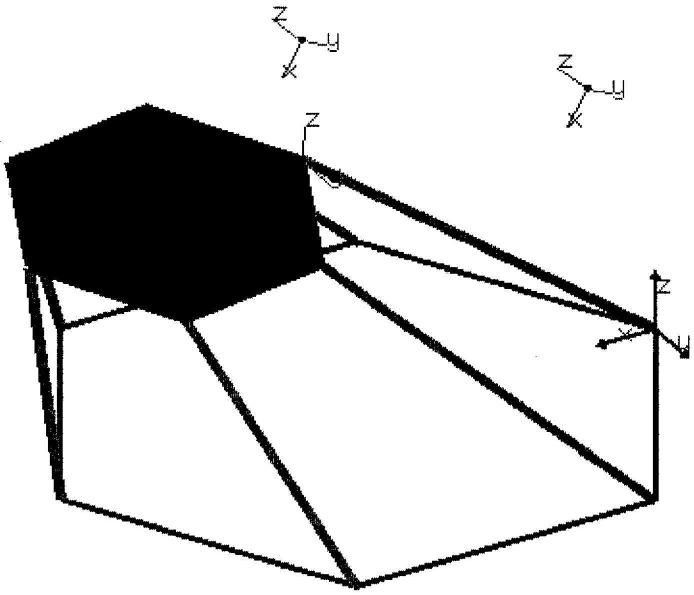


**Figure 4.10: Learn Pose Points View**

In order to make the software easy to install and use for the first time user, we provide a software user manual for reference, which is listed in Appendix C.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER V

# APPLICATION EXAMPLES

Thanks to the rigid, precise and agility of Stewart Platform, it is ideal solution for motion simulation, positioning device, Nano technology and assembly line. In this chapter two application examples are given by both using the reconfigurable model to produce numerical kinematic model and using the GUI to create graphical model and simulation. We are using a 3-3 Stewart Platform, which is one of the most popular configurations using in flight simulators, and a 5-4 Stewart Platform, which has never been studied and has no solved solution in published papers.

## 5.1 3-3 Stewart Platforms

### 5.1.1 Problem Description

The special case we consider here is 3-3 Stewart Platform shown in figure 5.1.

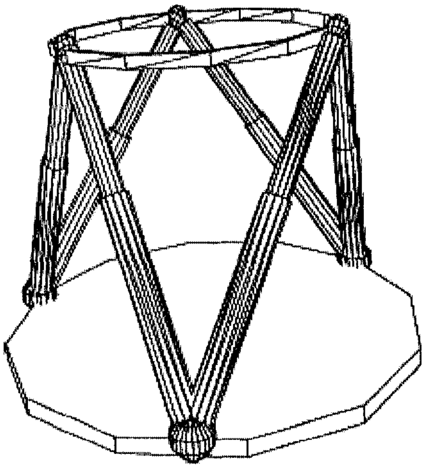


**Figure 5.1: 3-3 Stewart Platform**

**[Bruyninckx, H., 2005, Parallel Robots, http://www.roble.info/robotics/parallel/]**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

3-3 Stewart Platform is a parallel mechanism, which has two rigid bodies connected by six extensible struts. The six extensible struts are forming three concurrent pairs both on base and moving platform, hence six triangles are formed with six struts and two rigid bodies. The joints used to connect struts and the base are universal joints and the joints between the struts and moving platform are spherical joints or ball-and-socket joints. The only actuated joints are those prismatic joints on the struts.

### 5.1.2 kinematic Model

As we discussed in chapter 3, there are three conditions that can be used for developing the kinematic model. For any special topology, we have to analyze these three conditions one by one. One important rule to decide which conditions can be used in developing procedure is that any coincident joint can not be used as a condition, in other words, any zero-length side or diagonal can not be used as a condition [Figure 5.2].

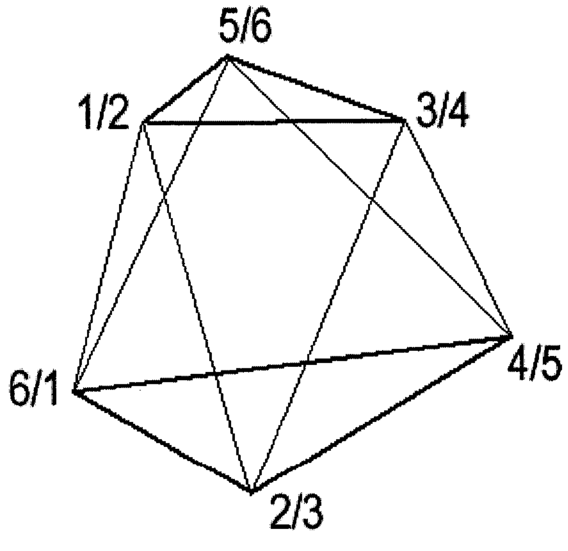


**Figure 5.2: Kinematic Model of 3-3 Stewart Platform**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

### 1S Common side conditions

Refer to the figure 5.2, three non-zero sides on the moving platform are sides A2-3, A4-5 and A6-1, but the corresponding sides B2-3, B4-5 and B6-1 on the base are all zero. Based on the above analysis, there is no common side condition that can be used for developing.

### 2S and 3S Common diagonal conditions

With reference to the figure 5.2, all the diagonals on the moving platform are non-zero, so are the corresponding diagonals on the base. All nine common diagonal conditions are available.

Consider the common diagonal condition equations (5.1) we developed in chapter 3

$$\begin{aligned} & A_i^2-(\sqrt{T_i^2-\rho_i^2}-\sqrt{T_{i+}^2-\rho_{i+}^2})^2 \\ & =\left[\rho_i \cdot \sin\phi_{i1}-\rho_{i+} \cdot \sin\phi_{i2}\right]^2+\left[B_i-\rho_i \cdot \cos\phi_{i1}-\rho_{i+} \cdot \cos\phi_{i2}\right]^2 \end{aligned} \tag{5.1}$$

For those 2S diagonals, which are forming triangles with the corresponding 2 adjacent sides.

$$\begin{aligned} & A_i=\sqrt{l_i^2+l_{i+1}^2-2 \cdot l_i \cdot l_{i+1} \cdot \cos\alpha_{i+1}} \\ & B_i=\sqrt{L_i^2+L_{i+1}^2-2 \cdot L_i \cdot L_{i+1} \cdot \cos\beta_{i+1}} \\ & \phi_{i1}=\gamma_i-\arctan\left(L_{i+1} \cdot \sin\beta_{i+1}/\left(L_i-L_{i+1} \cdot \cos\beta_{i+1}\right)\right) \\ & \phi_{i2}=\beta_{i+1}+\beta_{i+2}-\gamma_{i+2}+\arctan\left(L_{i+1} \cdot \sin\beta_{i+1}/\left(L_i-L_{i+1} \cdot \cos\beta_{i+1}\right)\right)-\pi \\ & i+=i+2 \end{aligned} \tag{5.2}$$

i =1, 2, 3, 4, 5, 6.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

For those 3S diagonals, which are forming quadrangles with the corresponding 3 adjacent sides.

$$A_i = \left[ l_i^2 + l_{i+1}{}^2 + l_{i+2}{}^2 + 2 \cdot l_i \cdot l_{i+2} \cdot \cos(\alpha_{i+1} + \alpha_{i+2}) - 2 \cdot l_i \cdot l_{i+1} l_{i+2} \cdot \cos\alpha_{i+1} \cdot \cos\alpha_{i+2} \right]^{1/2}$$

$$B_i = \left[ L_i^2 + L_{i+1}{}^2 + L_{i+2}{}^2 + 2 \cdot L_i \cdot L_{i+2} \cdot \cos(\beta_{i+1} + \beta_{i+2}) - 2 \cdot L_i \cdot L_{i+1} L_{i+2} \cdot \cos\beta_{i+1} \cdot \cos\beta_{i+2} \right]^{1/2}$$

$$\phi_{i1} =$$

$$\left| \arcsin\left( \frac{(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2} \cos\beta_{i+2})}{\sqrt{(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2} \cos\beta_{i+2})^2 + (L_{i+2} \sin\beta_{i+2} - L_i \sin\beta_{i+1})^2}} \right) - \beta_{i+1} + \pi/2 - \gamma_i \right|$$

$$\phi_{i2} =$$

$$\left| \arcsin\left( \frac{(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2} \cos\beta_{i+2})}{\sqrt{(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2} \cos\beta_{i+2})^2 + (L_{i+2} \sin\beta_{i+2} - L_i \sin\beta_{i+1})^2}} \right) + \beta i + 2 + \beta i + 3 - \gamma i + 3 - {}^{3\pi}\!/_{2} \right|$$

$$i+ = i + 3 \qquad (5.3)$$

i =1, 2, 3.

### 0S Superposition conditions

With reference to the figure 5.2, three pairs of joints are coincident with each other on the moving platform, they are joints 1/2, joints 3/4 and joints 5/6 and the corresponding sides on the base are non-zero. The superposition conditions can be applied to these three pairs of joints. The superposition condition equations can be developed as following.

$$T_i^2 - \rho_i^2 = T_{i+1}{}^2 - \rho_{i+1}{}^2 \qquad (5.4)$$

i =1, 3, 5.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## Kinematic Model of 3-3 Stewart Platform

The characteristic parameters for 3-3 Stewart Platform are six zero sides, they are three sides on the moving platform l1, l3, l5 and three sides on the base L1, L3, L5. The kinematic model of 3-3 Stewart Platform can be obtained as following.

$$\left\{\begin{array}{l} A1_i^{\ 2}-(\sqrt{T_i^{\ 2}-\rho_i^{\ 2}}-\sqrt{T_{i+2}^{2}-\rho_{i+2}^{2}})^2 \\ =\left[\rho_i\cdot\sin\phi1_{i1}-\rho_{i+2}\cdot\sin\phi1_{i2}\right]^2+\left[B1_i-\rho_i\cdot\cos\phi1_{i1}-\rho_{i+2}\cdot\cos\phi1_{i2}\right]^2 \\ \\ A2_j^{\ 2}-(\sqrt{T_j^2-\rho_j^2}-\sqrt{T_{j+3}^2-\rho_{j+3}^2})^2 \\ =\left[\rho_j\cdot\sin\phi2_{j1}-\rho_{j+3}\cdot\sin\phi2_{j2}\right]^2+\left[B2_j-\rho_j\cdot\cos\phi2_{j1}-\rho_{j+3}\cdot\cos\phi2_{j2}\right]^2 \\ \\ T_k^{\ 2}-\rho_k^{\ 2}=T_{k+1}^{\ \ 2}-\rho_{k+1}^{\ \ 2} \\ i=1,2,3,4,5,6 \\ j=1,2,3 \\ k=1,3,5 \end{array}\right. \tag{5.5}$$

Where:

$$A1_i=\sqrt{l_i^{\ 2}+l_{i+1}^2-2\cdot l_i\cdot l_{i+1}\cdot\cos\alpha_{i+1}}$$

$$B1_i=\sqrt{L_i^{\ 2}+L_{i+1}^2-2\cdot L_i\cdot L_{i+1}\cdot\cos\beta_{i+1}}$$

$$\phi1_{i1}=\gamma_i-\arctan\left(L_{i+1}\cdot\sin\beta_{i+1}/(L_i-L_{i+1}\cdot\cos\beta_{i+1})\right)$$

$$\phi1_{i2}=\beta_{i+1}+\beta_{i+2}-\gamma_{i+2}+\arctan\left(L_{i+1}\cdot\sin\beta_{i+1}/(L_i-L_{i+1}\cdot\cos\beta_{i+1})\right)-\pi$$

$$A2_j=\left[l_j^{\ 2}+l_{j+1}^{\ \ 2}+l_{j+2}^{\ \ 2}+2\cdot l_j\cdot l_{j+2}\cdot\cos(\alpha_{j+1}+\alpha_{j+2})-2\cdot l_j\cdot l_{j+1}l_{j+2}\cdot\cos\alpha_{j+1}\cdot\cos\alpha_{j+2}\right]^{1/2}$$

$$B2_j=\left[L_j^{\ 2}+L_{j+1}^{\ \ 2}+L_{j+2}^{\ \ 2}+2\cdot L_j\cdot L_{j+2}\cdot\cos(\beta_{j+1}+\beta_{j+2})-2\cdot L_j\cdot L_{j+1}L_{j+2}\cdot\cos\beta_{j+1}\cdot\cos\beta_{j+2}\right]^{1/2}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$\phi 2_{j1} =$$

$$\left| \arcsin\left( \frac{\left(L_{j+1} - L_j \cos\beta_{j+1} - L_{j+2}\cos\beta_{j+2}\right)}{\sqrt{\left(L_{j+1} - L_j \cos\beta_{j+1} - L_{j+2}\cos\beta_{j+2}\right)^2 + \left(L_{j+2}\sin\beta_{j+2} - L_j \sin\beta_{j+1}\right)^2}} \right) - \beta_{j+1} + \pi/2 - \gamma_j \right|$$

$$\phi 2_{j2} =$$

$$\left| \arcsin\left( \frac{\left(L_{j+1} - L_j \cos\beta_{j+1} - L_{j+2}\cos\beta_{j+2}\right)}{\sqrt{\left(L_{j+1} - L_j \cos\beta_{j+1} - L_{j+2}\cos\beta_{j+2}\right)^2 + \left(L_{j+2}\sin\beta_{j+2} - L_j \sin\beta_{j+1}\right)^2}} \right) + \beta_j + 2 + \beta_{j+3} - \gamma_{j+3} - {}^{3\pi}\!/_{2} \right|$$

### 5.1.3 Graphical Model and Simulation

This section gives the illustration of a given 3-3 Stewart Platform model and simulation with the following data:

The base coordinates:

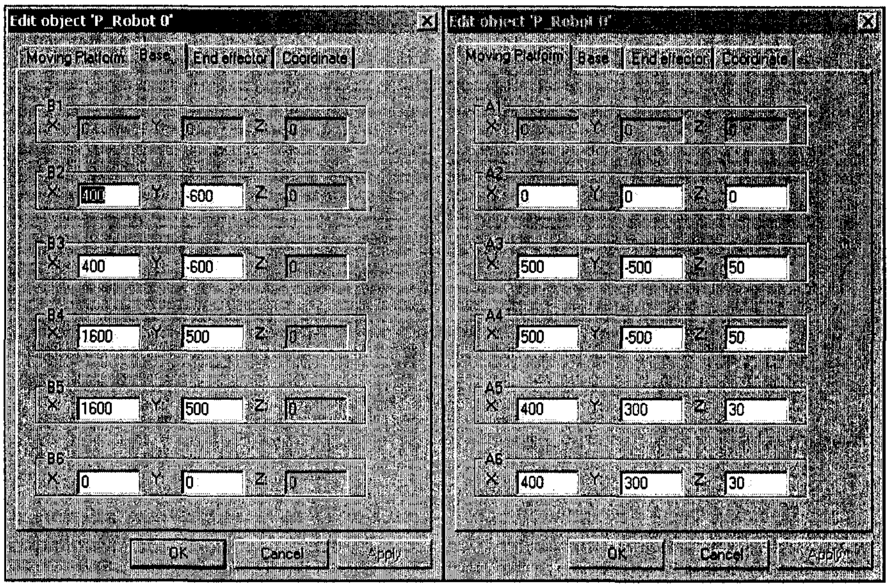
$$B = \begin{bmatrix} 0 & 0 & 0 \\ 400 & -600 & 0 \\ 400 & -600 & 0 \\ 1600 & 500 & 0 \\ 1600 & 500 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5.6}$$

The platform coordinates:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 500 & -500 & 50 \\ 500 & -500 & 50 \\ 400 & 300 & 30 \\ 4000 & 300 & 0 \end{bmatrix} \tag{5.7}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Input the above given base and platform coordinates as shown in figure 5.3.



B1 X 0 Y 0 Z 0
B2 X 400 Y -600 Z 0
B3 X 400 Y -600 Z 0
B4 X 1600 Y 500 Z 0
B5 X 1600 Y 500 Z 0
B6 X 0 Y 0 Z 0

A1 X 0 Y 0 Z 0
A2 X 0 Y 0 Z 0
A3 X 500 Y -500 Z 50
A4 X 500 Y -500 Z 50
A5 X 400 Y 300 Z 30
A6 X 400 Y 300 Z 30

Figure 5.3: Parameters Input of 3-3 Stewart Platform

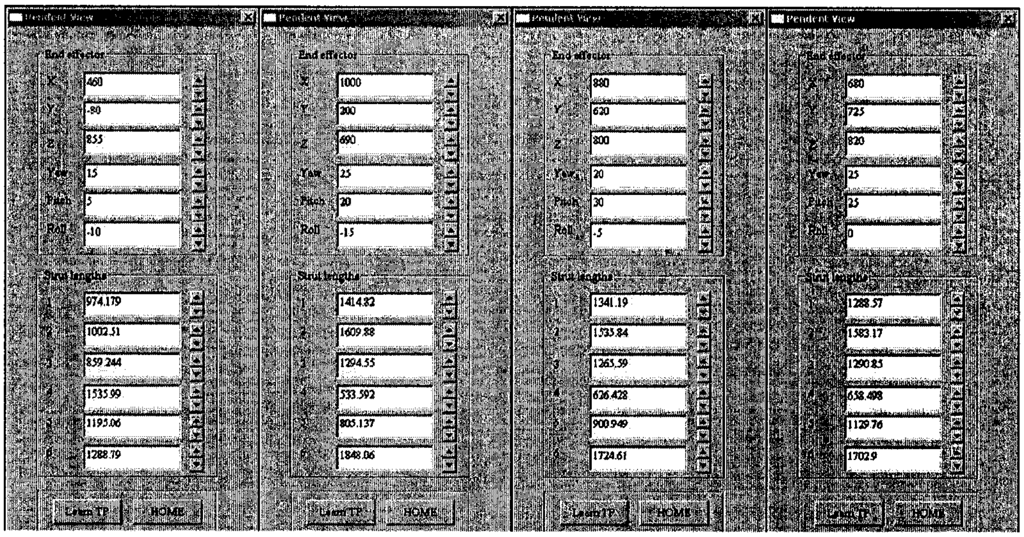
The simulation target points are listed in table 5.1.

Table 5.1: the Definition of Target Points

| Points | $P_X$ | $P_Y$ | $P_Z$ | Yaw | Pitch | Roll |
|---|---|---|---|---|---|---|
| 1 | 460 | -80 | 855 | 15 | 5 | -10 |
| 2 | 1000 | 200 | 690 | 25 | 20 | -15 |
| 3 | 880 | 620 | 800 | 20 | 30 | -5 |
| 4 | 680 | 725 | 820 | 25 | 25 | 0 |

Input target points in pendent view dialog as shown in figure 5.4.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.



End effector
X 460
Y -80
Z 855
Yaw 15
Pitch 5
Roll -10

Strut lengths
1 974.179
2 1002.51
3 859.244
4 1535.99
5 1195.06
6 1288.79

End effector
X 1000
Y 200
Z 690
Yaw 25
Pitch 20
Roll -15

Strut lengths
1 1414.82
2 1609.88
3 1294.55
4 533.592
5 805.137
6 1848.06

End effector
X 880
Y 620
Z 800
Yaw 20
Pitch 30
Roll -5

Strut lengths
1 1341.19
2 1535.84
3 1265.59
4 626.428
5 900.949
6 1724.61

End effector
X 680
Y 725
Z 820
Yaw 25
Pitch 25
Roll 0

Strut lengths
1 1288.57
2 1583.17
3 1290.85
4 658.498
5 1129.76
6 1702.9

**Figure 5.4: the Inputs of Target Points**

Click the "learn TP" button for each target point and then click "OK". The target simulation path is shown in the view window marked by four target points [Figure 5.5].

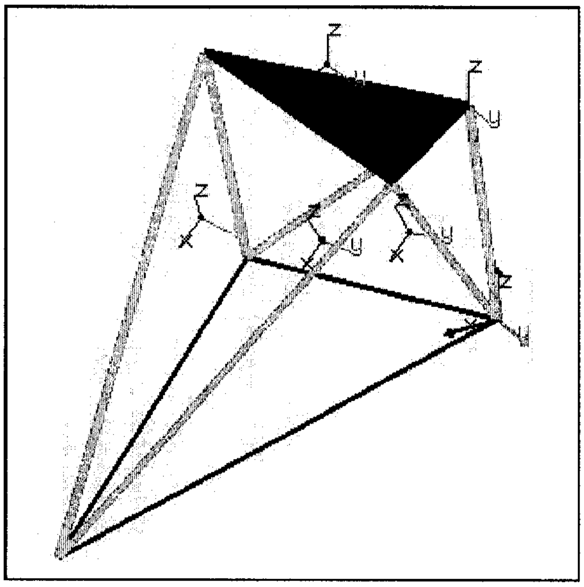


**Figure 5.5: the Simulation Path**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

By choosing the simulation menu in the GSP menu, the simulation animation is played in the view window following the target point path we defined.

## 5.2 5-4 Stewart Platforms (1-2-1-2)

### 5.2.1 Problem Description

The special case we consider here is 5-4 Stewart Platform with 1-2-1-2 joints arrangement on the moving platform. 5-4 Stewart Platform is also a parallel mechanism, which has two rigid bodies connected by six extensible struts. The six extensible struts are connecting the five joints on the base and the four joints on the moving platform, hence there are two pairs of joints on the moving platform coincident with each other and one pair of joints on the base are superposed. The joints used to connect struts and the base are universal joints and the joints between the struts and moving platform are spherical joints or ball-and-socket joints. The only actuated joints are those prismatic joints on the struts.

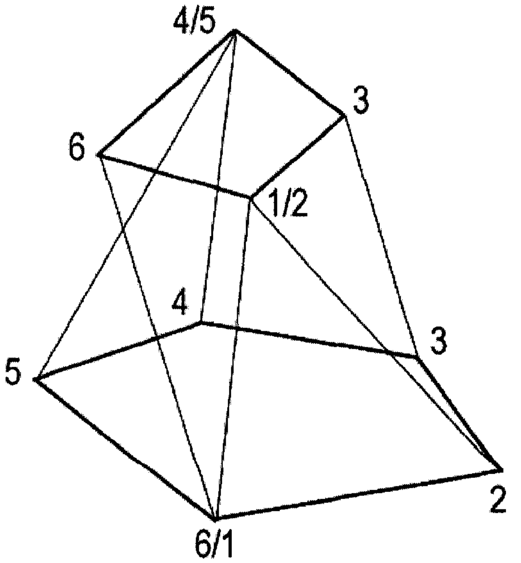


Figure 5.6: Kinematic Model of 5-4 Stewart Platform

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

### 5.2.2 kinematic Model

By the same way, three conditions can be used to develop the kinematic model as follows and the kinematics model of the 5-4 Stewart Platform is shown in [Figure 5.6].

**1S Common side conditions**

Refer to the figure 5.6, two pairs of joints (1/2 and 4/5) on the moving platform are coincident with each other and so do the joint 1 and joint 6 on the base, hence only the sides 2, 3, 5 can be used for the 1S condition.

Consider the common side condition equations (5.8) we developed in chapter 3

$$A_i^2-(\sqrt{T_i^2-\rho_i^2}-\sqrt{T_{i+}^2-\rho_{i+}^2})^2$$
$$=\left[\rho_i\cdot\sin\phi_{i1}-\rho_{i+}\cdot\sin\phi_{i2}\right]^2+\left[B_i-\rho_i\cdot\cos\phi_{i1}-\rho_{i+1}\cdot\cos\phi_{i2}\right]^2 \tag{5.8}$$

For the 1S condition, for which we have 3 of them, the followings are fiven:

$$A_i=l_i$$
$$B_i=L_i$$
$$\phi_{i1}=\gamma_i$$
$$\phi_{i2}=\beta_{i+1}-\gamma_{i+1}$$
$$i+=i+1 \tag{5.9}$$

where i =2, 3, 5.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

### 2S and 3S Common diagonal conditions

With reference to the figure 5.6, all the diagonals on the moving platform are non-zero, so are the corresponding diagonals on the base. All nine common diagonal conditions are available.

For those 2S diagonals:

$$A_i = \sqrt{l_i^2 + l_{i+1}^2 - 2 \cdot l_i \cdot l_{i+1} \cdot \cos\alpha_{i+1}}$$

$$B_i = \sqrt{L_i^2 + L_{i+1}^2 - 2 \cdot L_i \cdot L_{i+1} \cdot \cos\beta_{i+1}}$$

$$\phi_{i1} = \gamma_i - \arctan\left(L_{i+1} \cdot \sin\beta_{i+1} / \left(L_i - L_{i+1} \cdot \cos\beta_{i+1}\right)\right)$$

$$\phi_{i2} = \beta_{i+1} + \beta_{i+2} - \gamma_{i+2} + \arctan\left(L_{i+1} \cdot \sin\beta_{i+1} / \left(L_i - L_{i+1} \cdot \cos\beta_{i+1}\right)\right) - \pi$$

$$i+ = i+2 \tag{5.10}$$

i =1, 2, 3, 4, 5, 6.

For those 3S diagonals:

$$A_i = \left[l_i^2 + l_{i+1}{}^2 + l_{i+2}{}^2 + 2 \cdot l_i \cdot l_{i+2} \cdot \cos(\alpha_{i+1} + \alpha_{i+2}) - 2 \cdot l_i \cdot l_{i+1} l_{i+2} \cdot \cos\alpha_{i+1} \cdot \cos\alpha_{i+2}\right]^{1/2}$$

$$B_i = \left[L_i^2 + L_{i+1}{}^2 + L_{i+2}{}^2 + 2 \cdot L_i \cdot L_{i+2} \cdot \cos(\beta_{i+1} + \beta_{i+2}) - 2 \cdot L_i \cdot L_{i+1} L_{i+2} \cdot \cos\beta_{i+1} \cdot \cos\beta_{i+2}\right]^{1/2}$$

$$\phi_{i1} =$$

$$\left|\arcsin\left(\frac{\left(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2} \cos\beta_{i+2}\right)}{\sqrt{\left(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2} \cos\beta_{i+2}\right)^2 + \left(L_{i+2} \sin\beta_{i+2} - L_i \sin\beta_{i+1}\right)^2}}\right) - \beta_{i+1} + \pi/2 - \gamma_i\right|$$

$$\phi_{i2} =$$

$$\left|\arcsin\left(\frac{\left(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2} \cos\beta_{i+2}\right)}{\sqrt{\left(L_{i+1} - L_i \cos\beta_{i+1} - L_{i+2} \cos\beta_{i+2}\right)^2 + \left(L_{i+2} \sin\beta_{i+2} - L_i \sin\beta_{i+1}\right)^2}}\right) + \beta i + 2 + \beta i + 3 - \gamma i + 3 - {}^{3\pi}\!/_{2}\right|$$

$$i+ = i+3 \tag{5.11}$$

i =1, 2, 3.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## Kinematic Model of 5-4 Stewart Platform

The characteristic parameters for 5-4 Stewart Platform are three zero sides, they are two sides on the moving platform l1, l4 and one side on the base L6. The kinematic model of 5-4 Stewart Platform can be obtained as following.

$$\left\{\begin{array}{l} A1_i{}^2-(\sqrt{T_i^2-\rho_i^2}-\sqrt{T_{i+2}^2-\rho_{i+2}^2})^2 \\ =\left[\rho_i\cdot\sin\phi1_{i1}-\rho_{i+2}\cdot\sin\phi1_{i2}\right]^2+\left[B1_i-\rho_i\cdot\cos\phi1_{i1}-\rho_{i+2}\cdot\cos\phi1_{i2}\right]^2 \\ A2_j{}^2-(\sqrt{T_j^2-\rho_j^2}-\sqrt{T_{j+3}^2-\rho_{j+3}^2})^2 \\ =\left[\rho_j\cdot\sin\phi2_{j1}-\rho_{j+3}\cdot\sin\phi2_{j2}\right]^2+\left[B2_j-\rho_j\cdot\cos\phi2_{j1}-\rho_{j+3}\cdot\cos\phi2_{j2}\right]^2 \\ A3_k{}^2-(\sqrt{T_k^2-\rho_k^2}-\sqrt{T_{k+1}^2-\rho_{k+1}^2})^2 \\ =\left[\rho_k\cdot\sin\phi3_{k1}-\rho_{k+1}\cdot\sin\phi3_{k2}\right]^2+\left[B3_k-\rho_k\cdot\cos\phi3_{k1}-\rho_{k+1}\cdot\cos\phi3_{k2}\right]^2 \\ i=1,2,3,4,5,6 \\ j=1,2,3 \\ k=2,3,5 \end{array}\right. \tag{5.12}$$

Where:

$$A1_i=\sqrt{l_i^2+l_{i+1}^2-2\cdot l_i\cdot l_{i+1}\cdot\cos\alpha_{i+1}}$$

$$B1_i=\sqrt{L_i^2+L_{i+1}^2-2\cdot L_i\cdot L_{i+1}\cdot\cos\beta_{i+1}}$$

$$\phi1_{i1}=\gamma_i-\arctan\left(L_{i+1}\cdot\sin\beta_{i+1}/\left(L_i-L_{i+1}\cdot\cos\beta_{i+1}\right)\right)$$

$$\phi1_{i2}=\beta_{i+1}+\beta_{i+2}-\gamma_{i+2}+\arctan\left(L_{i+1}\cdot\sin\beta_{i+1}/\left(L_i-L_{i+1}\cdot\cos\beta_{i+1}\right)\right)-\pi$$

$$A2_j=\left[l_j{}^2+l_{j+1}{}^2+l_{j+2}{}^2+2\cdot l_j\cdot l_{j+2}\cdot\cos(\alpha_{j+1}+\alpha_{j+2})-2\cdot l_j\cdot l_{j+1}l_{j+2}\cdot\cos\alpha_{j+1}\cdot\cos\alpha_{j+2}\right]^{1/2}$$

$$B2_j=\left[L_j{}^2+L_{j+1}{}^2+L_{j+2}{}^2+2\cdot L_j\cdot L_{j+2}\cdot\cos(\beta_{j+1}+\beta_{j+2})-2\cdot L_j\cdot L_{j+1}L_{j+2}\cdot\cos\beta_{j+1}\cdot\cos\beta_{j+2}\right]^{1/2}$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

$$\phi 2_{j1} =$$

$$\left| \arcsin\left( \frac{\left(L_{j+1} - L_j \cos\beta_{j+1} - L_{j+2}\cos\beta_{j+2}\right)}{\sqrt{\left(L_{j+1} - L_j \cos\beta_{j+1} - L_{j+2}\cos\beta_{j+2}\right)^2 + \left(L_{j+2}\sin\beta_{j+2} - L_j \sin\beta_{j+1}\right)^2}} \right) - \beta_{j+1} + \pi/2 - \gamma_j \right|$$

$$\phi 2_{j2} =$$

$$\left| \arcsin\left( \frac{\left(L_{j+1} - L_j \cos\beta_{j+1} - L_{j+2}\cos\beta_{j+2}\right)}{\sqrt{\left(L_{j+1} - L_j \cos\beta_{j+1} - L_{j+2}\cos\beta_{j+2}\right)^2 + \left(L_{j+2}\sin\beta_{j+2} - L_j \sin\beta_{j+1}\right)^2}} \right) + \beta_j + 2 + \beta_{j+3} - \gamma_{j+3} - {}^{3\pi}\!/_{2} \right|$$

$$A3_k = l_k$$

$$B3_k = L_k$$

$$\phi 3_{k1} = \gamma_k$$

$$\phi 3_{k2} = \beta_{k+1} - \gamma_{k+1}$$

### 5.2.3 Graphical Model and Simulation

This section gives the illustration of a given 5-4 Stewart Platform model and simulation with the following data:

The base coordinates:

$$B = \begin{bmatrix} 0 & 0 & 0 \\ 400 & -600 & 0 \\ 900 & -800 & 0 \\ 1200 & 100 & 0 \\ 800 & 700 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad (5.13)$$

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The platform coordinates:

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 600 & -300 & 100 \\ 800 & 0 & -50 \\ 800 & 0 & -50 \\ 300 & 400 & 50 \end{bmatrix} \tag{5.14}$$

Input the above given base and platform coordinates. The simulation target points are listed in table 5.2. Input target points in pendent view dialog and learn TP. The target simulation path is shown in the view window marked by four target points [Figure 5.7].

**Table 5.2: the Definition of Target Points**

| Points | $P_x$ | $P_y$ | $P_z$ | Yaw | Pitch | Roll |
|---|---|---|---|---|---|---|
| 1 | 400 | 200 | 800 | 0 | 0 | 0 |
| 2 | 400 | 500 | 800 | 0 | 20 | 0 |
| 3 | 400 | 500 | 1000 | -20 | 20 | 10 |
| 4 | 900 | 500 | 800 | 30 | 20 | 20 |

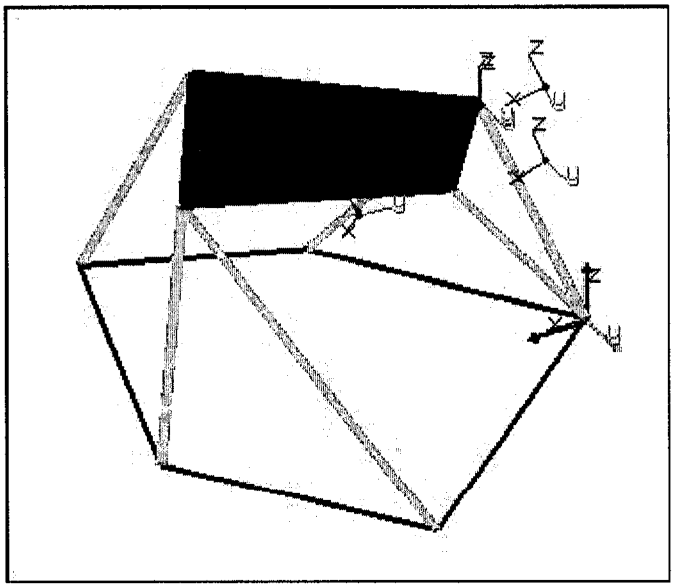


**Figure 5.7: the Simulation Path**

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# CHAPTER VI

# CONCLUSION AND FUTURE WORK

## 6.1 Conclusion

In this research, we present a new geometry-based algorithm, which is reconfigurable and unified model for solving forward kinematics of General Stewart Platform. As we proved that this new reconfigurable kinematic model can be applied to any special case of General Stewart Platforms and it is very useful for digging a new topology for special application and verifying the existing method as well.

There are seven topologies that have been studied in published papers and in this thesis three of them have been studied, which are 6-6, 3-3 and 6-3 Stewart Platform [Table 3.1- 3.4]. Basically, at least one configuration for each type of applicable base configuration has been studied. More examples should be done for the rest of the configurations in the future. The cases studied in published papers are: 6-6, 6-3, 3-3, 6-5, 6-4, 6-2, 5-5 and the first three cases have been verified in this thesis. The cases that have not been studied in published papers include 5-4, 5-3, 5-2, 4-4, 4-3, 4-2 and the 5-4 Stewart Platform has been studied in this thesis.

In addition, an upgraded version of UKMS simulation system has been developed with add-in feature of model and simulation of General Stewart Platform. The upgraded software can be used for work cell planning and simulating.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

## 6.2 Future Work

There are still a number of unsolved research and development issues related to the kinematic model and 3D graphical user interface as well.

- The method to use for solving the reconfigurable kinematic model is still a challenge. The current extra-sensor method can be applied for real-time application but it can not obtain all solution and the accuracy is probably a problem because of the error of sensors. Numerical iterative methods sometimes requires heavy computation that makes the calculation time too long, besides, it still requires a high performance computer for real-time use. A univariate polynomial is the best solution for kinematic model but as we know it is very hard to find out the solution. In order to get the univariate polynomial the substructure approach and compatible equations might be an efficient way.
- Carry out more case studies as discussed in section 6.1.
- For the simulation software, the manufacturing model of industrial robots should be introduced for better visualization and dynamic analysis and control design.
- An improvement of the GUI should be made for a better visualization and more convenient operation, such as some embedded link bodies should be available to pick up for better view of robotic model.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# APPENDICES

## APPENDIX A

## SAMPLE MATLAB PROGRAM

```
    options = optimset('MaxFunEvals',10000);
    options = optimset('MaxIter',10000);
    n=fsolve('fungsp',
[10,9.6,10.5,10.2,9.8,10,1.045,1.048,1.044,1.05,1.052,1.042],options)
    fungsp.m
    function m = fungsp(n)
    % initial conditions --begin-- %
    a=[0,0,0;  5,-8.5,0;  15,-8.5,0;  20,0,0;  15,8.5,0;  5,8.5,0];
    b=[0,0,0;  10,-17,0;  30,-17,0;  40,0,0;  30,17,0;  10,17,0];
    T=[40,30,35,25,28,33];
    for i=7:10
        T(i)=T(i-6);
    end
    % initial conditions --end-- %
    % calculating sides, diagonals and angles on the base----begin----%
    for i=1:6
      i1=i;
      i2=i;
      i3=i;
      if(i==4)i3=-2;
      elseif(i==5) i2=-1;i3=-1;
      elseif(i==6) i1=0;i2=0;i3=0;
      end
      %-----------calculate length of sides or diagonals-------begin-------%
      l(i)=sqrt((a(i,1)-a(i1+1,1))^2+(a(i,2)-a(i1+1,2))^2+(a(i,3)-a(i1+1,3))^2);
      l1(i)=sqrt((a(i,1)-a(i2+2,1))^2+(a(i,2)-a(i2+2,2))^2+(a(i,3)-a(i2+2,3))^2);
      l2(i)=sqrt((a(i,1)-a(i3+3,1))^2+(a(i,2)-a(i3+3,2))^2+(a(i,3)-a(i3+3,3))^2);
      L(i)=sqrt((b(i,1)-b(i1+1,1))^2+(b(i,2)-b(i1+1,2))^2+(b(i,3)-b(i1+1,3))^2);
      L1(i)=sqrt((b(i,1)-b(i2+2,1))^2+(b(i,2)-b(i2+2,2))^2+(b(i,3)-b(i2+2,3))^2);
      L2(i)=sqrt((b(i,1)-b(i3+3,1))^2+(b(i,2)-b(i3+3,2))^2+(b(i,3)-b(i3+3,3))^2);
      %-----------calculate length of sides or diagonals-------end-------%
    end
    for i=1:6
      for z=1:5
        ii(z)=i+z;
        if (ii(z)>6) ii(z)=ii(z)-6;
        end
      end
      %-----------calculate angles(betas) on the base-------begin-------%
      if((L(i)==0)&(L(i+5)~=0))
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
        beta(i)=acos((L(ii(1))^2+L(ii(5))^2-L2(ii(2))^2)/(2*L(ii(1))*L(ii(5))));
        beta(ii(1))=beta(i);
      elseif((L(i)~=0)&(L(ii(5))==0))
        beta(i)=acos((L(i)^2+L(ii(4))^2-L2(ii(1))^2)/(2*L(i)*L(ii(4))));
        beta(ii(5))=beta(i);
      elseif ((L(i)==0)&(L(ii(5))==0))
        beta(i)=acos((L(ii(1))^2+L(ii(4))^2-L1(ii(2))^2)/(2*L(ii(1))*L(ii(4))));
        beta(ii(1))=beta(i);
        beta(ii(5))=beta(i);
      else
        beta(i)=acos((L(i)^2+L(ii(5))^2-L1(ii(5))^2)/(2*L(i)*L(ii(5))));
      end
      % calculate angles(betas) on the base-------end-------%
    end
    % calculating sides, diagonals and angles on the base----begin----%
    % define rous and gamas------begin-------%
    q=1;
    for p=1:10
      if(p>6)
        q=p-6;
      end
      rou(p)=n(q);
      gama(p)=n(q+6);
    q=p+1;
    end
    % define rous and gamas------end-------%
    % Calculate A(i), B(i), fi1(i) and fi2(i)--------begin----------%
    k=1;
    for z=1:5
      kk(z)=k+z;
      if (kk(z)>6) kk(z)=kk(z)-6;
      end
    end
    for k1=1:6
      k=k1;
      A(k)=l(k);
      B(k)=L(k);
      fi1(k)=gama(k);
      fi2(k)=beta(kk(1))-gama(kk(1));
    end
    for k2=7:12
      k=k2-6;
      A(k2)=l1(k);
      B(k2)=L1(k);
      fi1(k2)=gama(k)-atan(L(kk(1))*sin(beta(kk(1)))/(L(k)-
L(kk(1))*cos(beta(kk(1)))));
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
      fi2(k2)=beta(kk(1))+beta(kk(2))-
gama(kk(2))+atan(L(kk(1))*sin(beta(kk(1)))/(L(k)-L(kk(1))*cos(beta(kk(1)))))-pi;
    end
    for k3=13:15
      k=k3-12;
      A(k3)=l2(k);
      B(k3)=L2(k);
      fi1(k3)=abs(asin((L(kk(1))-L(k)*cos(beta(kk(1)))-
L(kk(2))*cos(beta(kk(2))))/sqrt((L(kk(1))-L(k)*cos(beta(kk(1)))-
L(kk(2))*cos(beta(kk(2))))^2+(L(kk(2))*sin(beta(kk(2)))-L(k)*sin(beta(kk(1))))^2))-
beta(kk(1))-gama(k)+0.5*pi);
      fi2(k3)=abs(asin((L(kk(1))-L(k)*cos(beta(kk(1)))-
L(kk(2))*cos(beta(kk(2))))/sqrt((L(kk(1))-L(k)*cos(beta(kk(1)))-
L(kk(2))*cos(beta(kk(2))))^2+(L(kk(2))*sin(beta(kk(2)))-
L(k)*sin(beta(kk(1))))^2))+beta(kk(2))+beta(kk(3))-gama(kk(3))-1.5*pi);
    end
    % Calculate A(i), B(i), fi1(i) and fi2(i)--------end---------%
    % Forming simultaneous equations-----------begin-----------%
    num=1;
    for k=1:15
      if(k<7) kk=1;
      elseif(k>12) kk=3;k=k-12;
      else kk=2;k=k-6;
      end
      if((l(k)~=0) & (L(k)~=0))
      % Common side conditions %
      m(num)=A(k)^2-(sqrt(T(k)^2-rou(k)^2)-sqrt(T(k+kk)^2-rou(k+kk)^2))^2-
(rou(k)*sin(fi1(k))-rou(k+kk)*sin(fi2(k)))^2-(B(k)-rou(k)*cos(fi1(k))-
rou(k+kk)*cos(fi2(k)))^2;
        num=num+1;
      end
    end
    for t=1:6
      if (l(t)==0)
      % Coincident conditions %
      m(num)=T(t)^2-rou(t)^2-T(t+1)^2+rou(t+1)^2;
        num=num+1;
      end
    end
    % Forming simultaneous equations-----------end-----------%
```

# APPENDIX B

## SAMPLE C++ PROGRAM

```
glNewList(listJointA0, GL_COMPILE);
CCoordinate *pCoordinateA0= new CCoordinate(100.0,100.0,100.0,'0');
pCoordinateA0->Render();
glEndList ();

glNewList(listJointB0, GL_COMPILE);
CCoordinate *pCoordinateB0= new CCoordinate(100.0,100.0,100.0,'1');
pCoordinateB0->Render();
glEndList ();

if (m_Create)
{
GSPInverse(EE0, basePM0, movePM0, movePM2, Strutlen0);

glPushMatrix();         // Start Drawing Robot

glCallList(listJointB0);

// DRAW BASE
glColor4f(0.0f,0.0f,1.0f,1.0f);
glLineWidth(3);

glBegin(GL_LINE_LOOP);
glVertex3f( m_xBASE1, m_yBASE1, m_zBASE1);
glVertex3f( m_xBASE2, m_yBASE2, m_zBASE2);
glVertex3f( m_xBASE3, m_yBASE3, m_zBASE3);
glVertex3f( m_xBASE4, m_yBASE4, m_zBASE4);
glVertex3f( m_xBASE5, m_yBASE5, m_zBASE5);
glVertex3f( m_xBASE6, m_yBASE6, m_zBASE6);
glEnd();

// DRAW STRUTS
glPushMatrix();
glColor4f(0.0f,1.0f,0.0f,1.0f);
glLineWidth(6);

glBegin(GL_LINES);
glVertex3f( m_xBASE1, m_yBASE1, m_zBASE1);
glVertex3f( movePM2[0][0], movePM2[1][0], movePM2[2][0]);

glVertex3f( m_xBASE2, m_yBASE2, m_zBASE2);
glVertex3f( movePM2[0][1], movePM2[1][1], movePM2[2][1]);
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
glVertex3f( m_xBASE3, m_yBASE3, m_zBASE3);
glVertex3f( movePM2[0][2], movePM2[1][2], movePM2[2][2]);

glVertex3f( m_xBASE4, m_yBASE4, m_zBASE4);
glVertex3f( movePM2[0][3], movePM2[1][3], movePM2[2][3]);
glVertex3f( m_xBASE5, m_yBASE5, m_zBASE5);
glVertex3f( movePM2[0][4], movePM2[1][4], movePM2[2][4]);

glVertex3f( m_xBASE6, m_yBASE6, m_zBASE6);
glVertex3f( movePM2[0][5], movePM2[1][5], movePM2[2][5]);
glEnd();

// DRAW MOVING PLATFORM
glPushMatrix();
glColor4f(0.0f,0.0f,1.0f,1.0f);
glLineWidth(4);

glBegin(GL_TRIANGLE_FAN);
glVertex3f( movePM2[0][0], movePM2[1][0], movePM2[2][0]);
glVertex3f( movePM2[0][1], movePM2[1][1], movePM2[2][1]);
glVertex3f( movePM2[0][2], movePM2[1][2], movePM2[2][2]);
glVertex3f( movePM2[0][3], movePM2[1][3], movePM2[2][3]);
glVertex3f( movePM2[0][4], movePM2[1][4], movePM2[2][4]);
glVertex3f( movePM2[0][5], movePM2[1][5], movePM2[2][5]);
glEnd();

glBegin(GL_LINE_LOOP);
glVertex3f( movePM2[0][0], movePM2[1][0], movePM2[2][0]);
glVertex3f( movePM2[0][1], movePM2[1][1], movePM2[2][1]);
glVertex3f( movePM2[0][2], movePM2[1][2], movePM2[2][2]);
glVertex3f( movePM2[0][3], movePM2[1][3], movePM2[2][3]);
glVertex3f( movePM2[0][4], movePM2[1][4], movePM2[2][4]);
glVertex3f( movePM2[0][5], movePM2[1][5], movePM2[2][5]);
glEnd();

glLineWidth(1);

///draw endeffector coodinates
glPushMatrix();
glTranslated(m_iGSPX,m_iGSPY,m_iGSPZ);

glRotated(m_iGSPyaw, 1.0, 0.0, 0.0);
glRotated(m_iGSPpitch, 0.0, 1.0, 0.0);
glRotated(m_iGSProll, 0.0, 0.0, 1.0);
glCallList(listJointA0);
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
glPopMatrix();//End
glPopMatrix();//End
glPopMatrix();//End
glPopMatrix();//End
}

else
{
GSPInverse(EE0, basePM0, movePM0, movePM2, Strutlen0);

glPushMatrix();          // Start Drawing Robot

        glCallList(listJointB0);

// DRAW BASE
glColor4f(0.0f,0.0f,1.0f,1.0f);
glLineWidth(3);

glBegin(GL_LINE_LOOP);
glVertex3f( m_xBASE1, m_yBASE1, m_zBASE1);
glVertex3f( m_xBASE2, m_yBASE2, m_zBASE2);
glVertex3f( m_xBASE3, m_yBASE3, m_zBASE3);
glVertex3f( m_xBASE4, m_yBASE4, m_zBASE4);
glVertex3f( m_xBASE5, m_yBASE5, m_zBASE5);
glVertex3f( m_xBASE6, m_yBASE6, m_zBASE6);
glEnd();

// DRAW STRUTS
glPushMatrix();
glColor4f(0.0f,1.0f,0.0f,1.0f);
glLineWidth(6);

glBegin(GL_LINES);
glVertex3f( m_xBASE1, m_yBASE1, m_zBASE1);
glVertex3f( movePM2[0][0], movePM2[1][0], movePM2[2][0]);

glVertex3f( m_xBASE2, m_yBASE2, m_zBASE2);
glVertex3f( movePM2[0][1], movePM2[1][1], movePM2[2][1]);

glVertex3f( m_xBASE3, m_yBASE3, m_zBASE3);
glVertex3f( movePM2[0][2], movePM2[1][2], movePM2[2][2]);

glVertex3f( m_xBASE4, m_yBASE4, m_zBASE4);
glVertex3f( movePM2[0][3], movePM2[1][3], movePM2[2][3]);
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

```
glVertex3f( m_xBASE5, m_yBASE5, m_zBASE5);
glVertex3f( movePM2[0][4], movePM2[1][4], movePM2[2][4]);

glVertex3f( m_xBASE6, m_yBASE6, m_zBASE6);
glVertex3f( movePM2[0][5], movePM2[1][5], movePM2[2][5]);
glEnd();

// DRAW MOVING PLATFORM
glPushMatrix();
glColor4f(0.0f,0.0f,1.0f,1.0f);
glLineWidth(4);

glBegin(GL_TRIANGLE_FAN);
glVertex3f( movePM2[0][0], movePM2[1][0], movePM2[2][0]);
glVertex3f( movePM2[0][1], movePM2[1][1], movePM2[2][1]);
glVertex3f( movePM2[0][2], movePM2[1][2], movePM2[2][2]);
glVertex3f( movePM2[0][3], movePM2[1][3], movePM2[2][3]);
glVertex3f( movePM2[0][4], movePM2[1][4], movePM2[2][4]);
glVertex3f( movePM2[0][5], movePM2[1][5], movePM2[2][5]);
glEnd();
glBegin(GL_LINE_LOOP);
glVertex3f( movePM2[0][0], movePM2[1][0], movePM2[2][0]);
glVertex3f( movePM2[0][1], movePM2[1][1], movePM2[2][1]);
glVertex3f( movePM2[0][2], movePM2[1][2], movePM2[2][2]);
glVertex3f( movePM2[0][3], movePM2[1][3], movePM2[2][3]);
glVertex3f( movePM2[0][4], movePM2[1][4], movePM2[2][4]);
glVertex3f( movePM2[0][5], movePM2[1][5], movePM2[2][5]);
glEnd();

glLineWidth(1);

///draw endeffector coodinates
glPushMatrix();
glTranslated(m_iGSPX,m_iGSPY,m_iGSPZ);

glRotated(m_iGSPyaw, 1.0, 0.0, 0.0);
glRotated(m_iGSPpitch, 0.0, 1.0, 0.0);
glRotated(m_iGSProll, 0.0, 0.0, 1.0);
glCallList(listJointA0);

glPopMatrix();//End
glPopMatrix();//End
glPopMatrix();//End
glPopMatrix();//End
}
```

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# APPENDIX C

# USER MANUAL

### 1.1 What UKMS can do

UKMS is a modeling and simulation system designed for the Puma-Fanuc serial robots and Stewart Platforms. It is a 32-bit Microsoft Windows application and uses OpenGL kernel to create good quality, high speed, and precise 3D images.

With UKMS you can:

Model work cells using the UKMS modeling functional menu.
Record robot paths based on the given target points.
Simulate robot work motion
Generate the system parameters and the results.

Note: The system supports 6R Puma and Fanuc type serial robots and Stewart Platforms (Parallel robots). The software designed by Ms. Zhongqing Ding and Anqi Wang.

### 1.2 System Requirements (recommendation)

Processor: 500MHz minimum, 1.0GHz or higher recommended.
Memory: 64MHz minimum, 256MHz or higher recommended.
Graphics: 4MHz RAM minimum, 16MHz or higher RAM recommended.
Mouse type: 2 button minimum, 3 button recommended.
Disk space: 10M free disk space for the software occupation.
Operating system: Windows 98, Windows 2000 or Windows XP.
Screen resolution: 1024*768 or higher recommended.
CD-ROM: required for installing UKMS.

### 2.1 Installation

1 Exist from all other applications and ensure that the system you are using is Windows based operation system.

2 Insert the UKMS 1.0 installation CD-ROM into your CD-ROM drive or DVD-ROM drive.

3 Double click on the Setup.exe file.

4 When the installation dialog is launched, click on the "next" botton.

5 Type the program display name that you want use to display in the system program list of the system or just keep the default program name "UKMS", then click on the button "next".

6 Type the program folder name that you want use to copy program files to or just keep the default folder name "UKMS".

7 Select the destination disk in which you want the program installed, then click on the button "next".

8 The installation program will automatically install all necessary files into the desired folder and click on the button "finish" to end.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

9 At this time, you will see a shortcut key on the desktop and a program folder in the "Start" menu.

### 2.2 Uninstall

Use the Windows programs removing platform to remove UKMS and all its files from the system. The procedure is as following:

1. Open control panel from Start menu > setting > control panel.
2. Double click on the icon "Add and Remove Programs".
3. Scroll down to select the UKMS 1.0 in the list.
4. Click on the "Change/Remove" button at the bottom-right corner.

### 3.1 Serial Robots Modeling

Through the Robot->Model menu, the kinematic structures of a serial robot can be modeled inside the program. The robot (Puma or Fanuc type) data can be saved for future use by the Geometry->Save as menu like other objects. To represent a robot's kinematic model in this system, we need to define all the joint coordinate frames with their positions and orientations. The procedure that creates a kinematic model is as following:

1. Select the joint coordinate frame with its orientations with arbitrary link length and offset under the "6R robot joints" tab.
2. Modify the robot arm lengths, offsets and the robot name with exact values under the "links definition" tab.
3. You can also move the robot position by changing the coordinates under the "Coordinates" tab. The robot can be translated and rotated about the world coordinate system.

### 3.2 Serial Robots Pendent View

After the kinematic model has been created, the direct kinematics can be visualized by the pendent view. The pendent view allows the user to manipulate a robot by changes the joint variable values. Meanwhile the robot moves to the new location. The pendent view is an accurate way for the user to create a Target Point (TP).

The pendent view is divided into three sections. The top section is labeled "Joint values" and show the numbers of the six joint variables θi. The second section is labeled "End-effector" and displays the absolute position and orientation of the tool frame of the robot with regard to the base coordinate frame. The third section has no label. A Home button moves the robot to the home position. After the position and orientation of the end-effector have been input, the Inverse button calculates and pop ups the 8 solutions of six joint variables θi. If any one is selected, the robot will move to that location. The Learn TP button saves the current data as a Target Point (TP).

If the user changes the six joint variables θi in the first section, the robot will immediately move according to the θi values. Meanwhile, the direct kinematic problem

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

will be calculated, and the absolute position and orientation of the tool frame of the robot will show in second section. If the user inputs the values in second section and presses the inverse button, the inverse kinematic problem will be solved. If one of the 8 solutions is selected, the values of the six joint variables will be shown in the top section.

### 3.3 Serial Robots Motion Simulation

The required robot path is given by a set of points, which is set in pendent view section. Those points are target locations of the robot's end-effector and are called Target Points (TPs). Each point is defined with its position and orientation. We need to calculate robot joint values for each point depending on the position and orientation of the point.

A path is a list of all the TPs that a robot follows during a sequence of motions. We can generate a TP using the Learn TP button on the pendent view. After the user has created the path, the simulation function can be used to simulate the activity of the robot following the path by simply clicking on the simulation menu or the simulation button on the toolbar.

### 4.1 Stewart Platforms Modeling

Through the GSP->Model menu, the kinematic structures of a Stewart Platform can be modeled inside the program. To represent a Stewart Platform's kinematic model in this system, we need to define all the joint coordinate frames with their positions and orientations for the moving platform and base respectively. The procedure that creates a kinematic model is as following:

1. Input the joint coordinates for the moving platform under the "Moving Platform" tab.
2. Input the joint coordinates for the base platform under the "Base" tab.
3. You can also move the robot position by changing the system origin coordinates under the "Coordinate" tab. The Stewart Platform can be translated and rotated about the world coordinate system.
4. You can specify an original position and orientation of the end-effector if you like.

### 4.2 Stewart Platforms Pendent View

After the kinematic model has been created, the inverse kinematics can be visualized by the pendent view. The pendent view allows users to manipulate a robot by changing the end-effector position and orientation. Meanwhile the system gives the corresponding struts' length. The pendent view provides a vivid way for users to evaluate the pose of the Stewart Platform corresponding to a specific Target Point (TP).

The pendent view is also divided into three sections. The top section is labeled "End effector" and shows the position and orientation of the end-effector. The second section is labeled "Strut lengths" and displays the struts' lengths corresponding to a given end-effector position and orientation. In the third section, a "Home" button can move the

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

robot to the home position. Once a target point is chosen, clicking the "Learn TP" button saves the current data as a Target Point (TP), which is also a preparation for motion simulation.

### 4.3 Stewart Platforms Motion Simulation

The target points (TPs) are given in the pendent view section and they give the required motion path as a set of points, which are defined with their position and orientation. Then the struts' lengths are calculated automatically for each target point depending on the position and orientation of the point.

A path is a list of all the TPs that the moving platform follows during a sequence of motions. The TPs are generated by clicking on the "Learn TP" button on the pendent view section. After the user has created the path, the simulation function can be used to simulate the activity of the robot following the path by simply clicking on the "GSP->Simulation" menu or the simulation button on the toolbar.

The procedure of work-cell modeling and simulation is pretty similar to the above-mentioned steps. By using this software, can we evaluate a complex work-cell, which has serial robots and parallel robots working collaboratively.

### 5.1 Objects (Cube and Cylinder)

An important task is the planning and designing of robotic work cell layouts which consist of robots, tools and environment. A simple 3D CAD system is provided to create basic geometric parameterized primitives like cubes, cones, cylinders. Using a 3 button mouse, we can modify the dimensions and color of the selected object and move the object in the scene. The selected object can be translated about the world coordinate system, and rotated about its own coordinate axis. All objects such as robots, box, cylinders, target points can be selected and saved as a *.obj file. Any other applications can load these files for use. A *.obj file format is same as the *.wld file.

### 5.2 File Functions

The work cell data which includes the robot kinematic model and the geometric data related to other objects are stored in a *.wld file. The File menu contains the usual Windows functions that allow you to open, close, and save files, and to exit the software. In addition, you can open the most-recently opened work cell files from this menu.

### 5.3 View Functions

The view position and direction affect how the model appears to the user when it is displayed. This system supports various standard views such as ISO view, XZ(front) view, YZ(side) view, XY(top) view, and Zoom in, Zoom out. Through this menu you can toggle the display of the Toolbar and Status Bar.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# REFERENCES

Akcali, I.D., Mutlu, H., 2006, "A novel approach in the direct kinematics of Stewart platform mechanisms with planar platforms", Journal of Mechanical Design 128 (1): 252-263 JAN.

Albus, J.S., Bostelman, R., Dagalakis, N.G., 1993, "The NIST.RoboCrane", Journal of Robotic Systems, Vol. 10 No.5, pp.709-24.

Angel, E., 2003, "Interactive Computer Graphics: A Top-Down Approach with OpenGL", 3rd ed, Boston: Addison Wesley, c2003, ISBN: 0-201-77343-0.

Anli, E., Alp, H., 2004, "The Stewart Platform Mechanism - A Review", Transactions on engineering, computing and technology, ISSN 1305-5313, V2 December.

Anton, S., Fries, T., Horsch, T., Schroer, F.W., Willnow, C., Wolf, C., 2001, "A framework for Realistic Robot Simulation and Visualisation", url: http://www.easy-rob.com/data/Frame-RRS.pdf.

Bonev, I. A., Ryu, J., 2000, "A new method for solving the direct kinematics of general 6-6 Stewart platforms using three linear extra sensors", Mechanism and Machine Theory, 35(3) pp.423-436

Bonney, M. C., Moser, J., Yong, Y.F., 1985, "Evaluation and use of a graphical robot simulator", A Case Study from ITT/AMTC Using GRASP, International Conference on Simulation in Manufacturing, Stratford upon Avon, pp. 57-61.

Boren, R.R., 1985, "Graphics simulation and programming for robotic workcell design", Robotic Age, vol. 7. pp. 30-33.

Broyden, C.G., 1965, "A class of methods for solving nonlinear simultaneous equations." Math. Comp., 19, pp. 577-593.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Bruyninckx, H., 2005, "Parallel Robotics" The Robotics WEBook, 20 August, url: http://www.roble.info/robotics/parallel/html/ParallelRobots-1.html

Callegari, M., Tarantini, M., 2003, "Kinematic. Analysis of a Novel Translational Platform", Transactions of the ASME, 125, pp. 308-315.

Carretero, J.A., Podhorodeski, R.P., Nahon, M.A., Gosselin, C.M., 2000, "Kinematic Analysis and Optimization of a New Three Degree of Freedom Parallel Manipulator" Journal of Mechanical Design, 122, pp. 17-24.

Chapman, D., 1998, "Sams teach yourself Visual C++ 6 in 21 days", a division of macmillan computer publishing, ISBN: 0-673-31240-9.

Chen N.X., Song S.M., 1992, "Direct position analysis of the 4-6 Stewart platform", In 22nd Biennial Mechanisms Conf., pages 75-80.

Chen N.X., Song S-M, 1994, "Direct position analysis of the 4-6 Stewart platform", ASME J. of Mechanical Design, 116(1):61-66.

D.E.Joyce, 1997, "Euclid's Elements", url: http://aleph0.clarku.edu/~djoyce/java/elements /toc.html.

Dafaoui, M., Amirat, Y., Pontnau, J., Francois, C., 1998, "Analysis and design of a six-dof parallel manipulator, modeling, singular configurations and workspace", IEEE Trans. on Robotics and Automation, 14(1) pp. 78-92.

Dai, W., Kampker, M., 1999, "PIN- a PC-based robot simulation and offline programming system using macro programming techniques", The 25th Annual Conference of the IEEE Industrial Electronics Society, Vol.1, pp.442-446.

Dasgupta, B., Mruthyunjaya, T.S., 1994, "A Canonical Formulation of the Direct Position Kinematics Problem for a General 6-6 Stewart Platform", Mechanism and Machine Theory, 29(6), pp. 819-827.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Derby, S., 1982, "Computer graphics simulation of robot arms", froc. MIT Conf. CAD/CAM Technology for Manufacruring Eng., Cambridge, pp. 215-221.

Derby, S.J., 1982, "Computer graphics robot simulation programs: A corn- parison", Proc. Winter Annual Meeting oJASME, pp 203-211.

Di Gregorio, R., 2001, "Kinematics of a New Spherical Parallel Manipulator with Three Equal Legs: The 3-URC. Wrist", Journal of Robotic Systems, 18(5), pp. 213-219.

Di Gregorio, R., 2002, "Translational Parallel. Manipulators: New Proposals", Journal of Robotic Systems, 19(12), pp. 595-603.

Ding, Z.Q., ElMaraghy, W.H., 2005, "A Unified Robotic Kinematic Simulation Interface" Thesis, University of Windsor.

Djuric, A.M., ElMaraghy, W.H, ElBeheiry, E.M., 2004, "Unified integrated modeling of robotic systems", NRC International Workshop on Advanced Manufacturing, June 2004, London, Canada.

Flow software technologies, 2002, "WORKSPACE 5.03 User Manual". url: http://www.workspace.com.

Fu, K.S., Gonzalez, R.C., Lee, C.S.G., 1987, "Robotics: Control, Sensing, Vision and Intelligence", McGraw-Hill International, ISBN: 0070226261.

Geng, Z., Haynes, L., 1994, "A 3-2-1 kinematic configuration of a Stewart platform and its application to six degrees of freedom pose measurements", Robotics and Computer-Integrated Manufacturing, Vol. 11 No.1, pp.23-34.

Gough, V.E., 1956, "Contribution to discussion of papers on research in automobile stability", control and tyre performance, vol. 171, pp. 392-395.

Howie, P., 1984, "Graphic simulation for off-line robot programming", Ro- dustrial and Management Engineering at the Unibotics Today, vol. 6, pp 63-66.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Huang, M.Z., 1996, "A note on kinematics of in-parallel actuated platform manipulators", In 2nd National Applied Mechanism and Machine Theory, 31(8) pp.1009-1018.

Hunt, K.H., 1978, "Kinematic Geometry of Mechanisms", Oxford University Press, ISBN: 0198561245.

Husain M., Waldron K.J., 1994, "Direct position kinematics of the 3-1-1-1 Stewart platform", ASME J. of Mechanical Design, 116(4):1102-1108.

Herman Bruyninckx, 1998, "Closed-Form Forward Position Kinematics for a (3-1-1-1) Fully Parallel Manipulator.", IEEE Transactions on Robotics and Automation, Vol.14, No. 2, April.

Husty, M. L., 1996, "An Algorithm for Solving the Direct Kinematics of General Stewart-Gough Platforms", Mechanism and Machine Theory, Vol. 31, No. 4, pp. 365-380.

Innocenti, C., 1998, "Forward Kinematics in Polynomial. Form. of the General Stewart Platform" , Trans. of. ASME, 3. of. Mechanical Design, vol. 123, pp. 254-260.

Innocenti, C., Parenti-Castelli, V., 1990, "Direct Position Analysis of the Stewart Platform. Mechanism", Mechanism and Machine Theory, 25(6), pp.611-621.

Jakobovic, D., Jelenkovic, L., 2002, "The Forward and Inverse Kinematics Problems for Stewart Parallel Mechanisms", International Scientific Conference on Production Engineering, II-001 - II-012.

Karger, A., Husty, M., 1996, "On Self-Motions of a Class of Parallel Manipulators", Recent Advances in Robot Kinematics, Kluwer Academic Publishers, pp. 339-348.

Kim, J., Park, F. C., 2001, "Direct kinematics analysis of 3-RS parallel mechanisms", Mechanism and Machine Theory, 36, pp. 1121-1134.

Knapczyk, J., Dzierzek, S., 1992, "Kinematic analysis of 6S-5S type Stewart platform mechanism by using vector method", In ARK, pages 123-128.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

42. Kong, X., Gosselin, C.M., 2002, "Generation and forward displacement analysis of analytic planar parallel manipulators", ASME J. of Mechanical Design, 124(2) pp. 294-300.

Koseeyaporn, P., 2003, "Component-based robotic simulation", Ph.D. dissertation, University of Vanderbilt.

Kovacs, W., 1985, "Previewing robotic motion with computer graphics", courses in information systems design, micro-Robotics Age, vol 7, pp. 16-19.

Ku, D., 1999, "Direct Displacement Analysis of a. Stewart Platform Mechanism", Mechanism and Machine Theory, 34(3), pp. 453-465.

Ku, D-M., 2000, "Forward Kinematic Analysis of a 6-3 Type Stewart Platform Mechanism", Proceedings of the I MECH E Part K Journal of Multi-body Dynamics, Volume 214, pp. 233-241(9), Number 4, 18 December.

Laloni, C., Wahl, F., 1995, "Principles of Robot Simulation and their Application in a PC-based Robot Simulation System", In Graphics and Robotics, Springer, 1-30.

Lee, K., Shah, D.K., 1988, "Kinematic Analysis of a. Three-Degrees-of-Freedom. In-Parallel. Actuated Manipulator", IEEE Journal of Robotics and Automation, 4(3), pp. 354-360.

Lee, T.Y., Shim, J.K., 2001, "Forward Kinematics of the General 6-6 Stewart Platform Using Algebraic Elimination" , Mechanism and Machine Theory 36 (9), pp. 1073~1085.

Lewis, F.L., 1999, "Robotics". Mechanical Engineering Handbook. Ed. Frank Kreith. Boca Raton: CRC Press LLC. url: http://www.itiomar.it/pubblica/dispense/ MECHANICAL%20ENGINEERING%20HANDBOOK/ch14.pdf

Li ,Y., Huang, Z., Chen, L., 2003, "Singular loci analysis of 3/6-Stewart manipulator by singularity-equivalent mechanism Robotics and Automation", Proceedings. ICRA '03. IEEE International Conference on Volume 2, 14-19 Sept. 2003 Page(s):1881 - 1886 vol.2.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Lin, W., Crane, C.D., Duffy, J., 1992, "Closed-form forward displacement analysis of the 4-5 in-parallel platforms", In 22nd Biennial Mechanisms Conf., volume DE-45, pages 521-527.

Lin, W., Duffy, J., Griffis M., 1990, "Forward displacement analysis of the 4-4 Stewart platform", In ASME Proc. of the the 21th Biennial Mechanisms Conf., pages 263-269.

Liu, K., Fitzgerald, J. M., Lewis, F. L., 1993, "Kinematic Analysis of a Stewart Platform. Manipulator", IEEE Transactions on Industrial Electronics, 40(2), pp. 282-293.

Ma, O., Angeles, J., 1992, "Architecture Singularities of Platform Manipulators", Proceedings of the IEEE International Conference on Robotics and Automation, Sacramento, CA, USA, April 11-14, pp. 1542-1547.

Mark, S., 1999, "Solving nonlinear problems in Matlab", url: http://www.math.mtu.edu/~msgocken/intro/node23.html.

Merlet, J. P., 1993, "Direct Kinematics of Parallel. Manipulators", IEEE Transactions on Robotics and Automation, 9(6), pp. 842-846.

Merlet, J.P., 1992, "Direct kinematics and assembly modes of parallel manipulators", International Journal of Robotics Research, 11(2):150-162.

Merlet, J.-P., 2006, "Parallel robots 2nd ed.", Springer, c2006. ISBN-10: 1-4020-4132-2.

Mirolo, C., Pagello, E., 1989, "A solid modeling system for robot action planning", IEEE Computer Graphics & Applications, pp. 55-69.

Nanua, P., Waldron, K.J., 1990, "Direct kinematic solution of a special parallel robot structure", In 8th RoManSy, pp. 134-142.

Nanua, P., Waldron, K.J., Murthy, V., 1990, "Direct Kinematic Solution of a Stewart Platform", IEEE Transactions on Robotics and Automation, 6(4), pp. 438-444.

Nielsen, J., Roth, B., 1996, "The direct kinematics of the general 6-5 Stewart-Gough mechanism", In ARK, pages 7-16.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Novak, B., 1984, "Robotic simulation facilitates assembly line design", in the Journal of Qualit Technology, Coin-Simulation, vol. 43, pp 298-299.

O'Leary, J.J., 1998, "CROBOTS: CAD Based Robot Simulation Tool", Master thesis, Memorial University of Newfoundland.

Orady, E. A., Osman, T. A., Bailo, C. P., 1997, "Virtual reality software for robotics and manufacturing cell simulation", Computers & Industrial Engineering, Vol.33, pp.87-90.

Owens, J., 1994, "WORKSPACE- a microcomputer-based industrial robot simulator and off-line programming system", Next Steps for Industrial Robotics, IEE Colloquium on , 17 May, pp.1-4.

Phillips, J., 1984 "Freedom in Machinery: Volume 1 Introducing Screw Theory", Cambridge University Press, ISBN: 0521236967.

Shi, X., Fenton, R.G., 1992, "Solution to the. Forward Instantaneous Kinematics for a. General 6-dof Stewart Platform", Mechanism and Machine Theory, 27(3), pp. 251-259.

Springfield, J. F., 1993, "An Open Extenrible System for Robot Simulation", Ph.D. Dissertation, Vanderbilt University.

Sreenivasan, S.V., Waldron, K.J., Nanua, P., 1994, "Closed-form direct displacement analysis of a 6-6 Stewart platform", Mechanism and Machine Theory, 29(6), pp. 855-864.

Stauffer, R.N., 1984, "Robot system simulation", Robotics Today, vol. 6, pp 81-90.

Stewart, D., 1965, "A Platform with. Six Degree of Freedom", Proc. of the Institute of Mechanical Engineering, Vol. 180, pp. 371-386.

Stoughton, R. S., Arai, T., 1993, "A Modified Stewart Platform Manipulator with Improved Dexterity", IEEE Transactions on Robotics and Automation, 9(2), pp.166-173.

Tsai, M.S., Shiau, T.N., Tsai, Y.J. and Chang, T.H., 2003, "Direct Kinematic Analysis of a 3-PRS Parallel Mechanism", Mechanism and Machine Theory, 38, pp. 71-83.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Wang, T., Chen, C.C., 1993, "On the numerical kinematic analysis. of general parallel robotic manipulators," IEEE Trans. Robot. Automat.,. vol. 9, pp. 272-285.

Wohlhart, K., 1994, "Displacement analysis of the general. spherical Stewart platform", Mechanism and Machine Theory, 29(4), pp. 581-589.

Yiu, Y. K., Cheng, H., Xiong, Z.H., Liu, G.F., Li, Z.X., 2001, "On the Dynamics. of Parallel Manipulators", Pr. of the 2001 IEEE Int. Conf. on Robotics and Automation, May, pp.21-26.

Yurt, S. N., 2002, "6-3 Stewart Platform Mechanism Kinematics", Ph.D Dissertation, Istanbul Technical University.

Zlatanov, D., Fenton, R.G., Benhabib, B., 1994, "Analysis of the Instantaneous Kinematics and Singular Configurations of Hybrid-Chain Manipulators," Proceedings of the ASME 23rd Biennial Mechanisms Conference, DE-Vol. 72, Minneapolis, MN, USA, September 11-14, pp. 467-476.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

# VITA AUCTORIS

NAME: Anqi Wang
PLACE OF BIRTH: P. R. China
EDUCATION: University of Windsor, Windsor, Ontario
2005-2007 M. A. Sc.
Dalian University of Technology, P. R. China
1999-2002 M. A. Sc.
Taiyuan University of science Technology
1993-1997 B. A. Sc.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.