REAL-TIME, OPEN CONTROLLER FOR RECONFIGURABLE MANUFACTURING SYSTEMS

By

MORETLO CELIA TLALE

Submitted in fulfillment of the requirements for the degree

MAGISTER INFORMATION TECHNOLOGY

in

FACULTY OF ENGINEERING AND INFORMATION TECHNOLOGY

(School of Information Technology)

at the

CENTRAL UNIVERSITY OF TECHNOLOGY, FREE STATE

Supervisor: Prof. H. J. Vermaak

Co-Supervisors: Mr. G. M. Muriithi

Dr N. S. Tlale

2013

DECLARATION

I declare that

Real-time, Open Controller for Reconfigurable Manufacturing Systems is my own work and that all the sources I have quoted have been indicated and acknowledged by means of references.

_____ _____

M. C. TLALE DATE

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and appreciation to the following:

- Firstly to God Almighty for his favour and his faithfulness in my life. For giving me strength, wisdom and understanding to complete this study.

- My Supervisors, Prof. H. J. Vermaak, Mr G. M. Muriithi and Dr N. S. Tlale for their thoughtful advice, criticism and comments that helped me to reach the final stage of the study.

- Thanks to my late father, Mr Jeremiah Tlale for believing in me and his continual support. May his soul rest in peace.

- To my colleagues for all the help I have received from you.

- Grateful appreciation to my friends for going the extra mile to support me throughout the study.

DEDICATION

This study is dedicated to my family, particularly to my mother Mrs Mamokhele Tlale and my brothers Mr Dipale Tlale and Mr Itumeleng Tlale.

ABSTRACT

Markets for manufactured products are characterized by a fragmentation of the market (with regards to size and time), and by shorter product cycles. This is due to the occurrence of mass customization and globalization. In mass customization, the same basic products are manufactured for a broad market, but then consumers are given the liberty to choose the “finishing touches” that go with the product. The areas that manufacturers now compete for are higher quality products, low cost and timely response to market changes. Appropriate business strategies and manufacturing technologies must thus be used to implement these strategic dimensions.

The paradigm of *Reconfigurable Manufacturing S*ystem (RMS) has been introduced to respond to this new market oriented manufacturing environment. The design of RMS allows ease of reconfiguration as it has a modular structure in terms of software and hardware. This allows ease of reconfiguration as a strategy to adapt to changing market demands. Modularity will allow the ability to integrate/remove software/hardware modules without affecting the rest of the system. RMS can therefore be quickly reconfigured according to the production requirements of new models, it can be quickly adjusted to exact capacity requirements as the market grows and products change, and it is able to integrate new technology.

In this research project, real-time, open controller is designed and developed for Reconfigurable Manufacturing Tools (RMTs). RMTs are the basic building blocks for RMS. Real time and openness of the controllers for RMT would allow firstly, for the modular design of RMTs (so that RMTs can be adapted easily for changing product demands) and secondly, prompt control of RMT for diagnosability.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

API’s	Application Programming Interfaces
COM	Component Object Model
COTS	Commercial-Off-The-Shelf
CSE	Control Software Elements
CSIR	Council For Scientific And Industrial Research
DH	Denavit Hartenberg
DML	Dedicated Manufacturing Lines
DMS	Dedicated Manufacturing Systems
DOF	Degrees Of Freedom
EMC	Enhanced Machine Controller
FMS	Flexible Manufacturing Systems
GMPTG	General Motors Powertrain Group
HM	Human Machine
HMI	Human Machine Interface
HOAM-CNC	Hierarchical Open Architecture Multi Processor-CNC

JVM	Java Virtual Machine
NIST	National Institute Of Standards And Technology
OAC’s	Open Architecture Controllers
OMAC	Open Modular Architecture Controllers
OS	Operating System
OSACA	Open Systems Architecture For Controls Within Automation Systems
OSEC	Open Systems Environment For Controllers
RM	Reconfigurable Machine
RMS	Reconfigurable Manufacturing Systems
RMT	Reconfigurable Manufacturing Tool
ROACS	Reconfigurable Open Architecture Controllers System
RTLinux	Real-Time Linux
RTOS	Real-Time Operating Systems
SA	South African
TUT	Tshwane University Of Technology

1. INTRODUCTION

1.1. CHAPTER OVERVIEW

The project entails the development of the control software and its architecture based on a reconfigurable machine that can produce higher quality products at lower cost with a timely response to market changes. The software will allow ease of reconfiguration as a strategy to adapt to market demands. The use of an open-architecture controller will provide the reconfigurable manufacturing tool (RMT) with the ability to be converted quickly to the production of new models.

The controller software to be developed should be implemented on different hardware platforms. The developed software will be modular. It should also allow for capacity and capability changes in the RMT. Capacity change of an RMT occurs when the throughput of the number of work-pieces of an RMT are changed. The capability change occurs when the RMT is changing its current manufacturing process to a different manufacturing process (this relates mostly to the number of degrees of freedom of an RMT). This software functionality will allow the reconfigurable manufacturing system to quickly adapt to market and product changes. Moreover, the ability of the software controller to easily integrate with new technology will be increased.

The project will focus on the control aspects of RMT. Less emphasis will be placed on the physical aspects of the machines.

The software should allow changes in production capacity and should be able to detect unacceptable behaviour to reduce the ramp-up time.

1.2. MOTIVATION FOR THE RESEARCH

The need to improve the standard of manufacturing in South African (SA) markets has led to this research. At the moment SA markets import manufacturing machines and goods because of many reasons, including: lack of technological know-how and expensive manufacturing costs due to lack of large markets. To ensure a broader knowledge base (in order to investigate different ways to improve manufacturing technology), RMS was suggested as a possible solution. Manufacturers of goods must be able to react to changes rapidly and cost-effectively in order to survive the new manufacturing environment. Reconfigurability of the systems (so that they can adapt to these changes) is a key enabler in avoiding costly modifications of the system. These also enable a large variety of products to be produced without compromising a system's performance.

1.3. THE RESEARCH PROBLEM

There is a need to design an open architecture controller to be implemented. Organizations like Open Systems Architecture for Controls within Automation systems (OSACA) in Europe, and Open Modular Architecture Controllers (OMAC) in United States, were extensively involved in defining and developing open architecture controllers. At the moment, some of their works are lying dormant. For example, OSACA ceased to operate officially in 1998 and some of its software solutions are now outdated. Moreover, the issue of satisfying real-time constraints has not been adequately met; hence there is a need to address the issue. Furthermore, in order to improve competitiveness of manufacturing enterprise, it should be able to produce new products without compromising quality and it should be able to respond to market changes without incurring more costs.

1.4. IMPORTANCE OF THE RESEARCH

Tshwane University of Technology (TUT), in conjunction with AMTS (now Technology Innovation Agency) and CSIR, is involved in developing a Reconfigurable Manufacturing System. There is therefore a need to design an open architecture controller to be implemented in this regard.

1.5. RESEARCH AIM

The project aims to develop, design and analyze a ROACS by system modeling to enable extendibility, scalability, inter-operatability and diagnostic ability from a remote support center at a low-cost. The ROACS to be developed should not depend on any specific hardware or software configurations.

1.6. PROJECT OBJECTIVES

1. Identifying RMS machine tool components, which include sensing and control modules.

2. Modeling an abstract machine monitoring and control methodologies in a modular approach, by:

    a. Modeling the different machine links,
    b. Modeling the different machine joints, and
    c. Kinematics modeling of overall machine tools.

3. To implement the controller in simulation or on a real platform.

4. To explore different methods/architectures that will allow for real-time control of RMTs.

5. To write portable and reusable algorithms which will be isolated from hardware configurations to enable modular and human machine interfaces (HMI).

1.7. THE HYPOTHESIS

The real-time control of machine tools can be achieved by:

- Implementation of an open-control architecture
- Development of model of machine components in software

1.8. DELIMITING THE RESEARCH

The research will involve survey of the current literature on open architecture controllers and a thorough analysis of the available open architecture controllers (OAC's) so as to summarize their advantages and limitations.

The study will limit itself to 5-axes reconfigurable manufacturing tools, or those platforms that are available at the Council for Scientific and Industrial Research (CSIR).

The reference architecture of the desired OAC might be designed around the currently available OAC.

LINUX, being the most easily accessible real-time operating system, will be used for the project. Its limitation compared to other real-time operating systems might be ignored.

The project would be based on the control aspects of RMS, and so less emphasis would be made on the physical aspects of the machines.

Either C, C++ or Java would be used to write the programs.

1.9. ASSUMPTIONS

Both SOLIDWORKS and MATLAB are capable of handling the modeling aspects of the project.

Most of the materials to be used are available at the university and at the CSIR – where practical training is undertaken.

1.10. DEFINING THE CONCEPTS

Ramp-up time:	The time it takes a newly introduced or reconfigured manufacturing system to reach sustainable, long-term production in terms of throughput and part quality, considering the impact of equipment and labour.
Scalability:	Ability to rapidly change production capacity.
Modularity:	Provision of hardware and software modules that can be integrated and used to design a system.
Integratability:	Each module built should have interfaces so as to promote component integration – regardless of the tool maker.
Customisation:	(i) At machine level: Building machines around parts of the family being manufactured. (ii) At control level: Use of an architecture technology to integrate control modules to achieve exact control of the functions desired.
Convertibility:	Promotion of shorter times for changing of tools, part programs fixtures and adjustments of degrees of freedom.
Diagnosability:	Ability to detect unacceptable behavior to reduce ramp-up time.

Open Architecture Controller: A controller designed and constructed for integration of new measurements and control devices and software modules by permitting access to a given set of internal controller variables.

1.11. DISSERTATION OUTLINE

1.11.1 Chapter 1

The first chapter is an introduction of study and gives details of what is expected in this research study.

1.11.2 Chapter 2

The chapter presents a literature review of open architectures, real-time controller and Reconfigurable Manufacturing Systems

1.11.3 Chapter 3

The chapter focuses on the theoretical background of the objects needed in order to develop a real-time, open architecture controller.

1.11.4 Chapter 4

The fourth chapter specifies the systematic process used to carry out the project.

1.11.5 Chapter 5

The chapter presents the discussion of the results achieved and the analysis of those results.

1.11.6 Chapter 6

The last chapter presents the conclusion of the study and also suggests future research studies.

1.12 CONCLUSION

The first chapter introduced the research topic. It stated the motivation and the importance of the research. The research aims and project objectives were also stated. Finally, the chapter layout of the thesis is detailed.

2. LITERATURE SURVEY

2.1. CHAPTER OVERVIEW

This chapter reviews a literature of open architectures, real-time controller and Reconfigurable Manufacturing Systems.

2.2. INTRODUCTION

The increased changes in modern manufacturing require a modular structure in terms of software and hardware. The application requires an improved system design that offers a degree of reconfigurability. The reconfiguration of the system controllers is the key factor in manufacturing.

2.3. EVOLUTION OF MANUFACTURING SYSTEMS

The history of manufacturing systems shows their development alongside the development of human kind. Automated manufacturing systems were initially created to relieve humans from tedious and boring jobs. The automated systems work more efficiently, faster, and produce a higher number of quality products than humans are able to.

In the past, consumers of goods purchased products that were easily available on the market – usually mass produced. The manufacturing systems output was of limited variety; however they satisfied the market at that time. As human

beings have progressed, they have also developed a need or desire for a variety of products. Products have become an expression of their individual personality. In addition to this, consumers are also becoming more highly educated and their demands are driven by the modern changes in technology. As the expectation of the society has grown, the market demand was no longer satisfied. The initial solution to this was to eliminate human operators in the process, as it is this factor which has been the cause of low productivity. Companies were forced to react to these new demands of the consumers' of products. This pushed manufacturing systems to another level.

In 1952, Massachusetts Institute of Technology developed numerical control (NC) in which simultaneous three–axis movements were demonstrated using laboratory built controller and Cincinnati Hindrotel vertical spindle. This was the start of flexible automation, which solved technological processing issues. During these years, the controller was hardcoded in electronic circuitry and part-programs were prepared with punch tapes (Liang, 2004:297).

The introduction of Computer Numerical Control (CNC) was in the early 1970's, and most of the electronic hardware and punch cards of the NC machines were replaced by a Dedicated Computer. CNC provided increased flexibility, greater reliability, and decreased floor space. The introduction of the flexible manufacturing systems (FMS) was then done to address the problem of inflexibility by providing generic processing capabilities by the use of CNC. The CNC control system allows programming to meet different manufacturing needs, but the architecture of the system or the algorithm cannot be altered. Flexible Manufacturing System (FMS) can produce a variety of products, with changeable mix, on the same system (Koren, Heisel, Jovane, Moriwaki, Pritschow, Ulsoy, Van Brussel, 1999:527). However, these machines have a longer payback

period (causing them to be expensive), as they do not provide an effective solution to capacity scaling and often possess excessive functionality.

Manufacturers of the high-volumes (e.g., the automotive industry) were enjoying a stable growing market with long product lifetimes. They did not have an urgent need to search for alternatives to the dedicated machining systems that they had been using for producing their machine components. Furthermore, the main building blocks of flexible systems, the CNC machines (which offered alternative solutions), were excluded. This exclusion was due to their high-cost, low reliability and low productivity.

In the 1980's, mass production was enhanced when production of good quality customized products at low prices was introduced. Elimination of waste and improved response time was an underlying principle. Unfortunately, the advances of all of the above-mentioned systems and principles were overshadowed by their short-comings as the quick change responsiveness in manufacturing became a key factor to competitiveness in the manufacturing industry.

Dedicated manufacturing lines (DML) produces single parts at a high volume. A new concept of Reconfigurable Manufacturing Systems (RMS) was identified as the vehicle to a successful manufacturing approach in the 21st century. The RMS is a modern system that bridges the gap between the DML and the FMS (Koren, 2010:17).

By the late 1990's, the paradigm of the RMS started to emerge in more entirety. This system was necessary due to more frequent and unpredictable market and

product changes, and the non-responsiveness of Dedicated Manufacturing Systems (DMS) and Flexible Manufacturing Systems (FMS). The changes include: abrupt changes in product demand and mix, frequent modifications to existing products, and the introduction of new products. In order to stay competitive and to accommodate these changes, manufacturing companies began to seek manufacturing systems that enabled a rapid response to market changes. Hu in ElMaraghy (2006:266) states that RMS is a manufacturing system with customized flexibility and FMS is a manufacturing system with general flexibility.

Manufacturers are now looking toward reconfigurable systems whose functionality and production can be changed to the exact capacity as needed. Companies must be able to react to changes rapidly and cost-effectively to survive the new manufacturing environment. Reconfigurability of the systems will avoid costly modifications of the system which would be required in order to adapt to these changes. The reconfigured systems would also enable a large variety of products to be produced, without compromising the system's performance.

Open architecture controllers (OAC) are designed to eliminate the problems involved in implementation. They do this by creating a flexible control system which can be attached to a wide variety of machine tools. From the outset, the controller is designed for reconfigurability. This allows a third party to integrate software and hardware modules without fear of failure. A provision is normally made for transparent exchange of information. Furthermore, the ability to transfer application software from one environment to another, whilst maintaining its capabilities, is essential. It is critical that the performance of an existing system is easily altered to suit the changing demands. OACs offer RMS the ability to be quickly converted for the production of new models, and to adjust to the exact

capacity requirements as the market grows and production changes. It is designed to offer services according to standard resources and standard rules that describe the syntax and services. The immediate impact of using standard resources is the reduction in incompatibility risks and the toning down of costs (as the resources used are more often freely available). A designer needs to take into consideration the fact that the controller needs to change its functionality, performance and dependability in order to adapt to different target platforms. This adaption must be done easily and cost effectively. Modules with proper communications interfaces are developed for different control tasks. The system should also be able to detect failures and easily implement the recovery process.

2.4. TYPES OF MANUFACTURING SYSTEMS

DMS and FMS are the most commonly used systems in the manufacturing industries to generate the products. DMS, or transfer lines, are based on inexpensive fixed automation and produce a company's core products (or parts) at high volume. Each dedicated line is typically designed to produce a single part at a high production rate – achieved by the operation of several tools simultaneously in machining stations, (Koren, 1999:527).

When the product demand is high the cost of the product will lower, and vice versa. DMS are not flexible and, as they have a fixed capacity, they are not scalable. Furthermore, it is difficult to convert them so that they are able to produce new products and they are designed to perform only a limited amount of operations.

FMS are made of general-purpose CNC machines and other programmable automation – which are expensive. They have changeable volume and mix (on

the same system), which can produce a variety of products. The initial cost is higher, and its production capacity is lower, than dedicated lines. FMS includes machines that are capable of performing a variety of operations, and through extension can produce a large range of different products (Koren, 2010:7). FMS are flexible and scalable. Unfortunately they are expensive, slow and are single-tool machines.

DMS focuses on the product volume by producing single products, whereas FMS focuses on product variety by producing multiple products.

2.5. RECONFIGURABLE MANUFACTURING SYSTEMS (RMS)

An RMS has the ability to reconfigure hardware and control resources at all of the functional and organizational levels. This allows the RMS to quickly adjust production capacity and functionality in response to sudden changes in the market or in regulatory requirements (Bi, Lang, Shen and Wang, 2008:975). These changes include: an increase in the frequency of the introduction of new products, changes in certain parts of existing products, large fluctuation in product demand and mix, changes in government regulations (safety and environment), and changes in process technology (Koren et al, 1999:527).

The advantage of using RMS is that the system can be upgraded in adaptability and functionality exactly when it is needed. By using a modular structure, the system can meet changeability requirements. In order for the system to be reconfigurable, the control software should be considered and be able to adjust to the new characteristics of the updated physical system.

Manufacturing systems that have fixed hardware and software are difficult, if not impossible; to be integrated into new hardware and software – hence there is a need for reconfiguration.

RMS can offer a rapid adaptability in their capacity and functionality in the new arising situations. It guarantees a high long-term benefit-to-cost-ratio as it provides system adaptability to new products. Another advantage of using RMS is that they can be upgraded in adaptability and functionality precisely when needed. This increases the lifetime of the system, as the system can produce new products by only changing at the software or hardware level. By using a modular structure, the system can meet changeability requirements. In order for the system to be reconfigurable, the control software should be considered and it should be able to adjust to any new characteristics of the updated physical system. The RMS can perform a pre-designed set of required operations due to repeatability and high productivity (Katz, 2007:431).

Figure 2.1 below demonstrates an RMS which is initially being designed to produce Product A. The figure shows how the system is reconfigured after some time to produce Product B as well. After a few years, Product A is phased out of the market but Product C is introduced. This figure demonstrates that an RMS can fulfil all of these requirements without major redesign of the system.

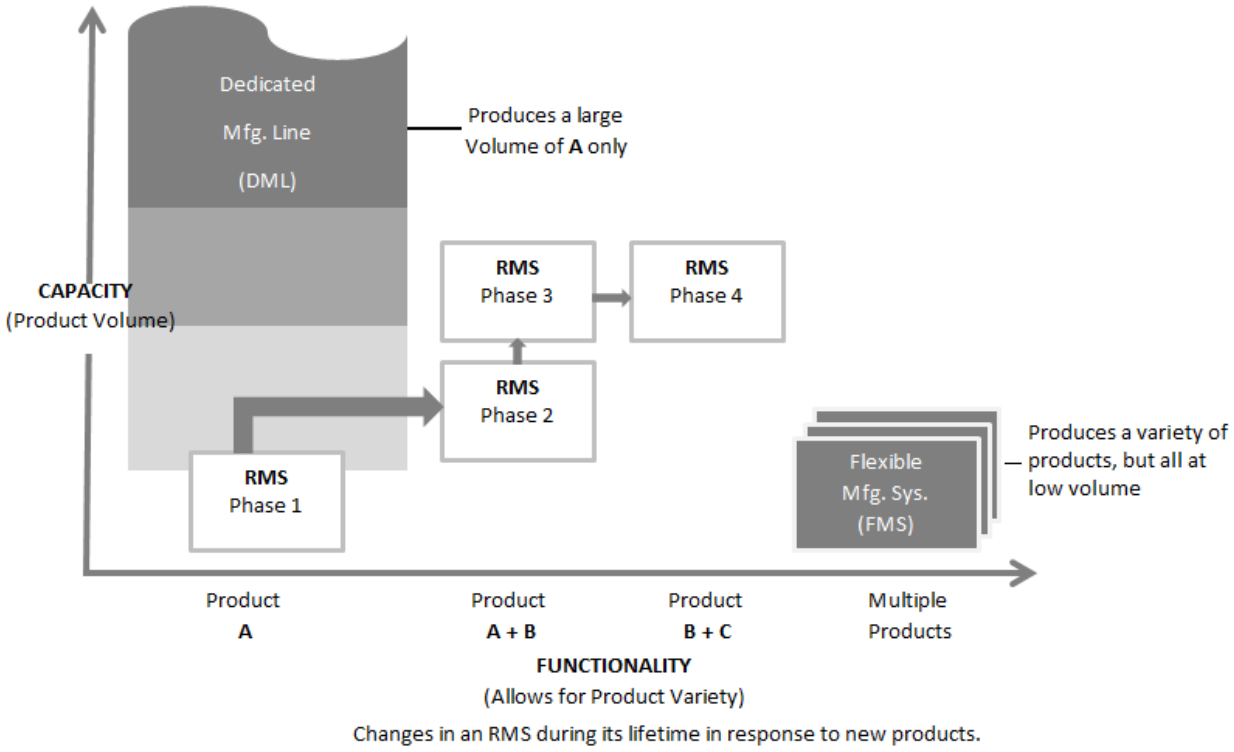

Figure 2.1 RMS (Koren, 2010, 18)

Reconfigurable Manufacturing Systems (RMS) are the cost-effective response to market changes towards manufacturing, that combine the high throughput of DML with the flexibility of FMS. RMS is able to react to changes quickly and effectively. The reconfiguration of the existing manufacturing system can be archived by the use of design systems that are created from modular components. The use of a modular structure will enable a required exchange for the manufacturing system by changing between modules. When the manufacturing systems are reconfigured, new functions must be added in order to accommodate the required changes and to produce new products.

In order to accommodate various productivity scenarios, a reconfigurable system should be able to adjust and react to changes. Although the design and

reconfiguration methodologies do not exist, the key to achieving reconfiguration is the enabling technologies.

2.6. OPEN ARCHITECTURE SOFTWARE PLATFORMS FROM INDUSTRY

2.6.1 OPEN SYSTEMS ARCHITECTURE FOR CONTROL WITHIN AUTOMATION SYSTEMS (OSACA)

The European based organization, OSACA, was formed in 1992 and stopped operating in 1997. It was formed with the intention of developing a vendor neutral open architecture controller to facilitate competitiveness and flexibility among suppliers and end-users of control systems. They focused on developing a communications system that would define both hardware and software independent interface – facilitating exchange of information between control modules. They also wanted to create reference architecture and a configuration system which would enable configuration of different application modules at boot-up. Their initial project, ESPIRIT III 6379, concluded by specifying a uniform system platform and modules for the application software. These were implemented in ESPIRIT 9115, where a software capable of running in different platforms was developed. Below figure is a reference model of the open architecture defined by OSACA, Figure 2.2.

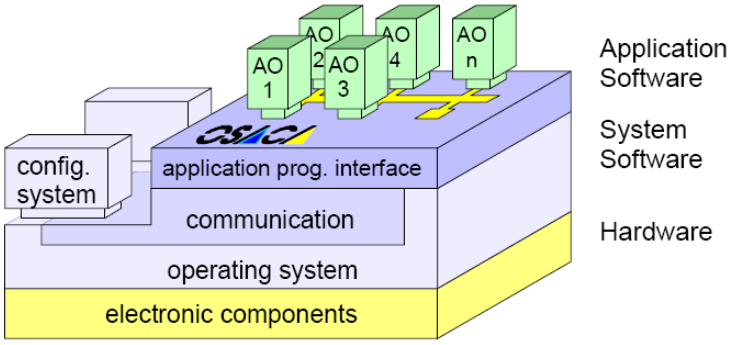

Figure 2.2: OSACA reference model, (Sperling and Lutz, 1997)

In their approach OSACA developed reference architecture for the effective management of a variety of machine tools, controllers, and application software and target platforms. OSACA attempted to standardize domain specific software constituent units and their interfaces, and to provide a communication system for their communication exchange. The OSACA architecture was developed using an object-oriented approach. They did this with the notion that object-oriented analysis and implementation generally provide a good support for software comprehensibility, interoperability, scalability and maintainability.

However, the interoperability of distributed application modules is supported by an infrastructure called OSACA platform. OSACA ceased to actively operate in 1997 and some of its algorithms are now dormant. Figure 2.2 illustrates the OSACA reference model.

2.6.2 HŰMNOS

This German government funded initiative started in 1995 and was created on the principles of OSACA. It had the intention of developing vendor-neutral

modules for the user-oriented application of the open architecture system. It involved machine tool builders like Alfing, Grunewald, Hellor, Homag, Huller; control vendors like Siemens, Bosch, DASA; and end-users like BMW and Daimler-Chrysler. They developed configuration systems, and demonstrations were done at both BMW and Daimler-Chrysler.

2.6.3. OPEN MODULAR ARCHITECTURE CONTROLLER (OMAC)

Open Modular Architecture Controllers (OMAC) is a United States based organization formed in 1994. It was geared towards producing a set of Application Programming Interfaces (API's) to be used by vendors to sell controller products to both automotive and aerospace industries. Their main objectives were to create a centre to share collective experiences on open architecture control from both software and hardware developers; to look at the merging of industry and government API's; and to promote the design of open architecture controllers among control designers. Furthermore, they collaborated with other international open control organizations to set international standards for open architecture control. To the present day, major United States companies like General Motors Powertrain Group (GMPTG) have been in the forefront of implementing OMAC technologies.

The OMAC API uses Microsoft Component Object Model (COM) as the initial framework in which to develop components. This means that control vendors could then concentrate on the application of specific improvements, instead of wasting time with programming resources.

OMAC API has been found to be extremely complicated. Despite this, it is widely accepted as a good potential foundation for open architecture models.

2.6.4. OPEN SYSTEMS ENVIRONMENT FOR CONTROLLERS (OSEC) AND JAPAN FA OPEN PROMOTIONS GROUP (JOP).

The Japanese initiative was formed with the intention of developing a Japanese open controller. Their main focus was PC-Based platforms and the Windows environment. The OSEC Architecture was intended to provide a standardized platform for industrial machine controllers. This platform was planned as a 'space' in which they could add their own unique values to the industrial machines – to end-users, machine makers, control vendors, software vendors, system integrators, etc. Thus far, a seven layer reference model and a programming interface referred to as Message Coordination Field were created. They also defined different C functions for each layer. Figure 2.3 illustrates the OSEC reference model.

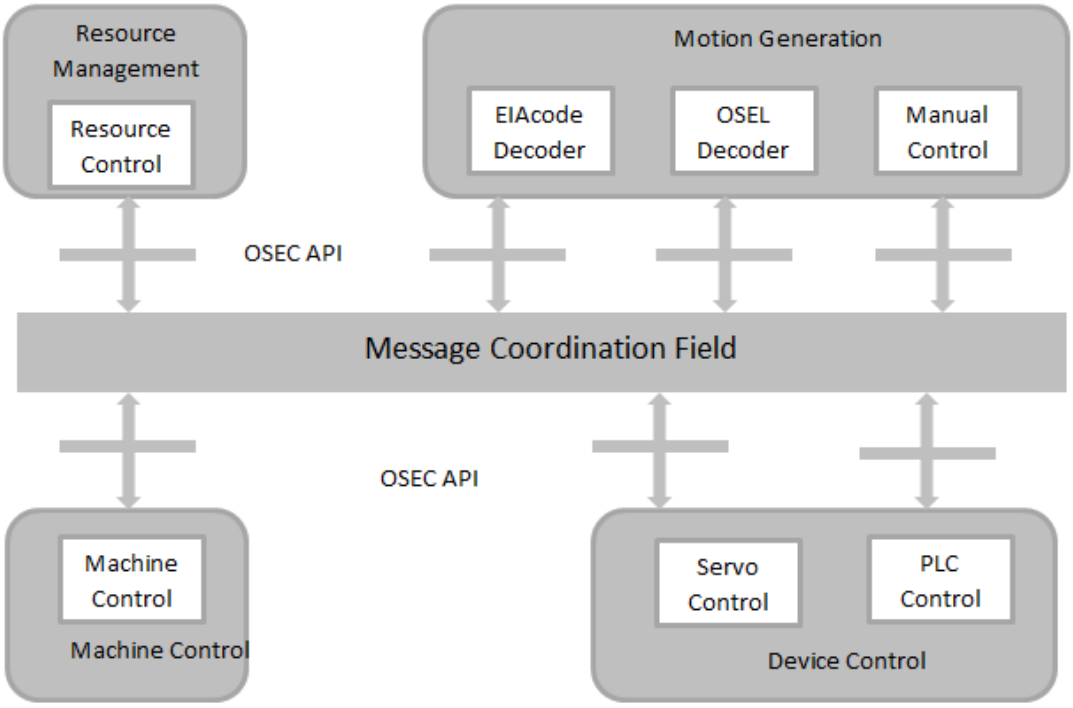

Figure 2.3 OSEC Reference Architecture (OSE, 1998)

Realizing the need for an open controller for factory automation, the Japanese companies merged and formed JOP. An Open-Controller Technical Committee was formed in 1996 (operational until 2000) to provide opportunities for various companies to discuss and work together on the standardization of open controller technologies. This was followed by the publication of an Application Programme Interface (API) between NC kernel (used for motion control tasks in real-time) and Human Machine Interface (HMI).

2.7. OPEN ARCHITECTURE CONTROLLERS FROM ACADEMIC PROJECTS

Educational institutions like the University of Michigan in Ann Arbor, United States, and the University of British Columbia in Vancouver, Canada, are also involved in open architecture control research. In their joint research Hierarchical Open Architecture Multi Processor-CNC (HOAM-CNC), they introduced the concept of having two buses in their machine hardware. One bus permits the introduction of new components, while the other is a CNC control bus.

The University of Michigan, through its Engineering Research Centre for RMS, has developed virtual and real machines controlled through a communication network and coordinated through unified software architecture. This University has a history of designing and developing control systems for multi-axes machines. In their research activities, they cover appropriate control structures design for reconfigurable machines and a common HMI API.

2.8. OTHER OPEN ARCHITECTURE CONTROLLERS FOR MANUFACTURING

The University of British Columbia developed a user-friendly, reconfigurable and modular tool-kit for motion and machining process control. The system can be used as an open architecture, modular operating system. It allows for the progressive development of real-time signal processing, and motion and process control application. The tool-kit uses script language to configure the control software in a highly open and modular way. The independent architecture for the hardware allows for the use of multiple DSP boards and host computers for machine tools. Additionally, it can be quickly configured to enable the control of different machine tools. It was, however, limited to CNC machines and more work was needed in tool path optimization.

2.9. OPEN CONTROLLER ARCHITECTURE

2.9.1 INTELLIGENT OPEN ARCHITECTURE CONTROLLERS

The development of systems requires machine tool controller architecture with high flexibility for monitoring and control demands. Pritschow, Altintas, Jovane, Koren, Mitsuishi, Takata, Van Brussel, Weck and Yamazaki in Mekid (2009:472) states that, PC-Based homogeneous and standardized platforms are pointed out as the best solution for flexibility. The use of OAC systems is still limited to research and development activities, since standardization still remains unresolved for their wide-spread application in the machine tool industry (Mekid, 2009:472). To enable customization, the software to be developed in this research study will be based on a reconfigurable machine tool.

2.10. REAL-TIME CONTROL

Gambier A (2004:1025) defines a real-time system as one in which the correctness of a result not only depends on the logical correctness of the calculation but also upon the time at which the result is made available. This illustrates that time is an important entity in the system that functions in real time. The system should produce the results within the specified time, which is within a specific time frame. If the system fails to produce the expected results within its time constraints, it has not met the real time constraints.

Zdenek Hanzalek in Petri Net Models for Manufacturing Systems (1996) states that in the real-time applications, a computer is connected directly to the physical equipment and is dedicated to controlling that equipment. Consequently, the system must meet response requirements that are mandated by the equipment itself, rather than those being dictated by the computer.

In order to operate a Real-Time system A Real-Time Operating Systems (RTOS) is needed. A RTOS provides facilities for implementing real-time systems, including multitasking (i.e. concurrency or potential parallelism), scheduling, inter-task communication mechanism, etc., (Gambier, 2004:1026).

A controller is designed in order to control the input/output of the system. The control system contains not only wired components but also algorithms, which must be programmed. This means that software is now included in the control loop (Gambier, 2004:1025). Verification of the reliability of the software is crucial in real-time systems, as one error can cause a fatal system failure – even leading to the serious injury of an individual. The software should do exactly what it is supposed, and designed, to do.

2.11. ENHANCED MACHINE CONTROLLER

Enhanced Machine Controller (EMC) was developed in 1992 by the United States of America's National Institute of Standards and Technology (NIST). EMC is a real-time controller for robots and machine tools. They used the well-known object-oriented programming languages, C and C++ to write the software. The controller was implemented on Windows NT and Sun Solaris machines running Linux operating systems. EMC uses real time extensions by running on Linux Kernels. Since its inception in 1992, many versions of EMC have been created and to date version 2.4 and 2.6 exist.

2.11. LINUX AS REAL-TIME APPLICATION

Linux is an open source platform that is freely available. Members of the public are allowed to modify the source code. Using this platform to develop applications reduces the development costs of buying the operating system (OS). Linux is able to run on most computers that are readily available. It has a variety of applications including graphical user interfaces, programming languages, etc. Sherer (2002: Online) states that Real-time Linux (RTLinux) is a hard real-time OS, as opposed to soft real-time systems or those that make no scheduling guarantees. A real-time OS is an operating system capable of guaranteeing timing requirements of the processes under its control (Barabanov, 1997:1). The use of Linux is recommended by many as it is open to the public and it is relatively cheap.

2.11. CONTROL FOR RECONFIGURABLE MACHINES IN OPEN ARCHITECTURE

Software modules are developed and then stored for reuse in a modular environment. When needed, they can then be implemented as open architecture controllers. The modules needed for the application are selected, and then configured by a method termed a “Control Configurator” that aids in integrating the controller to the selected machine (both continuous and discrete control), and automatically checks its real-time constraints (Koren, 1999:535). When creating the software modules, they should be independent of each other and be stored as a library. With a library of software modules, the modules can be re-used when needed. This will enable the machine using those modules to be reconfigured quickly to adapt to market demand.

2.11. RECONFIGURABLE SOFTWARE ARCHITECTURE

A Reconfigurable Software consists of reusable software components for the machine controller. The market demand drives the manufacturers to reuse the existing software components to produce new products. This lowers the cost of manufacturing and the response time to market demand. Software for machine control systems is usually designed and implemented with a set of components, such as: device drivers, control functions, and algorithms – all running on designated platforms (Wang and Shin, 2002:475). The programming language that will be used should be architecture neutral. The applications that will be developed should be able to run on multiple platforms.

2.12. RECONFIGURABLE CONTROLLERS FOR ROBOTS

The study which was performed by Atta-Konadu (2006:187) presents the design and implementation of modular controller for robotic application. It incorporates embedded technology using commercial-off-the-shelf (COTS) components and flexible architecture design patterns. It shows that C++ and Ada are robust real-time programming tools, whereas C++ is prone to poor readability and maintainability. Java was used because of its platform-independence. C++ and Ada showed dominance over Java in motion control design. Real-time communication architecture that was developed needs improvement with regards to its performance and openness, as the controller performed on an average level when it was tested.

2.13. MARKET OVERVIEW

The market provides different levels of openness for the controls that are available. Standardizing the computing platform is an important key factor in carrying out the human machine interface (HMI) and computer numerical control (CNC) software. In order to integrate third party software in the CNC products, the Application Programming Interface (API) must be used. Although most of today's controls offer openness concerning the operator related control functions, only few controls allow users to modify their low-level control algorithms to influence the machine-related control functions (Pritschow, 2001). Common definition is still not available for data that is passed back and forth via programming interface, even though many control systems provide open interfaces for software integration.

Market demands and external factors determine the necessary change requirements and the target degree of changeability (ElMaraghy, 2006:270). The manufacturing market should be responsive to changes that affect manufacturing

costs, such as oil prices, fluctuation of currency rate exchange, etc. This will ensure sustainability even if unpredictable situations arise. Manufacturers of products should generate products in a way that allows them to be responsive to these unpredictable situations. This can be achieved by producing the exact capacity of products, exactly when needed. This will ensure that the right capacity of products is available based on market demand. RMS ensures that production is adaptable to market demand.

2.14. GAP ANALYSIS

The literature reviewed in this chapter shows that there is a gap when it comes to software for the system that operates in real-time for a Reconfigurable Manufacturing Systems by the use of Open-Controllers. The concepts that were identified have been loosely defined and identified. Even though there are open architectures and Reconfigurable Controllers that exist, there is still much that needs to be done in order to enhance their openness. A real-time constraint in open architectures is still a concern that needs further research and development. The research aims to fill the identified gap so that real-time systems operate at a low cost by the use of open-controllers in the manufacturing environment.

2.15. CONCLUSION

This chapter reviewed literature on what has been done regarding Open Architecture Controllers. It highlights the changes to manufacturing that were stimulated by the Reconfigurable Manufacturing systems and its advantages towards manufacturing. The structure of RMS was also reviewed. The advantages of using Linux were examined and, based on these, Linux was

recommended for real-time application. Its limitations, especially compared to the real-time operating systems, may be disregarded.

3. THEORETICAL BACKGROUND

3.1 OPEN ARCHITECTURE SYSTEMS

An Open Architecture Control System can be gained through hardware and software standardization. This concept can be referred to as hardware and software independence. Any device that satisfies the functional requirements of a given logical device can be used. This is done by writing an appropriate class definition and incorporating it into the system. Hardware and software modules can be replaced in the system without requiring code changes outside of the scope of the class definition, and without affecting the behavioural operation of the system through this technique.

To alternate class definition(s) that can be incorporated in the system, a reconfigurable system often requires a re-building of the software (i.e. compile, link).

The system software for an open control system has to contain

(Pritschow in Koren, 2010:225):

1. An operating system to execute the software module functions.
2. A communication system to enable information interchange between modules utilizing a standard protocol.

3. STATIC VS DYNAMIC RECONFIGURABILITY

Static reconfigurability of the system requires the shutting down of the machine tool and the re-starting of the tool after the software has been modified. With this approach, the machine tool builders and integrators will be able to assemble the

working system by selecting modular components from vendors offering the current best combination of technology and cost.

Dynamic reconfigurability concept was introduced to further enhance a system. Dynamic reconfigurability refers to the ability to reconfigure a system at run-time (without requiring a system shut-down). This allows communication and control methods to be tested, evaluated, modified, and to be replaced without going through a re-start process. A rapid debugging and optimization that leads to a compressed development cycle is facilitated. As process conditions and requirements change, the dynamic reconfigurability allows the redistribution of process monitoring tasks.

3.1. RECONFIGURABLE MACHINE

A Reconfigurable Machine (RM) is a machine whose structure can be altered to provide either alternative functionality or incremental increase in its production rate in order to meet changing demand (Koren, 2010:206).

3.2. RECONFIGURABLE MACHINE TOOLS

Koren et al (1999) describes Reconfigurable Machine Tools (RMT) as a new type of modular machine with a changeable structure that allows adjustments of its resources (e.g., adding a second spindle unit). The modular design of machine tools is a key enabling technology to reconfigurability, as the machining system can easily be reconfigured by simply removing, adding or changing the constituent units or modules of the system or the machine (Y. Koren, 1999:533). RMTs enable the effective handling of the changing demands in the market, in terms of products and parts that needs to be manufactured by the system. The

possible changes can be workpiece size, production volume, production rate, etc. These possible changes can be achieved by just adding and/or removing specific modules. RMTs are cost effective when the products to be manufactured need changes. When the new product need arises, this need prompts a change in a machining process in order to produce new products to meet the demand.

3.3. RECONFIGURABLE SOFTWARE FOR MACHINE CONTROL SYSTEMS

The software components are implemented with configuration information, such as the number of inputs-outputs and their locations; the number and type of processors for execution; communication channels; and protocols for information exchange (Wang and Shin, 2002:476). A control plan is required to determine which module to use and should be implemented during the design time of the software. This control plan will enable the machine to be reconfigured by the reuse of existing modules.

3.4. DESIGN OF THE SYSTEM

To enable system scalability and response to market demand, the design of the system and its machines must have an adjustable structure. To enable reconfiguration, the system structure should be able to be adjusted at the system level (by adding machines) as well as at the machine level (by changing machine hardware and control software).

3.5. CONTROL SOFTWARE ELEMENTS

Control software elements (CSE) are deposited in a modular library. A high reusability of the modules results from the structural and functional information that is encapsulated into independent modules. By selecting modules into a

project, and associating the provided communication interfaces with each other, the new configurations are generated based on the module library. Automatic testing of these modules prevents generated configurations which are faulty. Using libraries reduces the production cost of the software and increases its quality. In this way, products can be produced at a lower cost.

3.6. CONTROL SYSTEMS INTERFACES

The control system is built up of internal interfaces that are utilized for interaction and data exchange between components. The important factor in this area is that they support real-time mechanisms. Platform concept enables the reconfigurability and adaptability of the controls for the internal architecture of the control system. To establish a defined but flexible way of communication between software components, the purpose is to hide hardware specific details from the software components.

The controls that are available in the market provide different levels of openness by the use of standardized computing platform. However, only a few controls are able to permit users to modify their low-level control algorithms to direct the functions of the related machines.

3.7. COMPUTER AUTOMATED MANUFACTURING (CAM) SOFTWARE

As the variety of machining processes and the number of control axes are increased, the supporting CAM software becomes more important to efficiently run the multi-functional machine tools (Spur in Moriwaki, 2008). There are a number of CAM software packages available on the market. Moriwaki (2008) states that, as compared to the advancement of machine tool hardware, the

development of software for multi-functional machining is still behind. This study is intended on advancing the development of software in the machining tool controller.

3.8. MODELLING SYSTEMS WITH MATLAB/SIMULINK – SIMMECHANICS TOOLBOX

SimMechanics toolbox is used for modeling the system by the use of help function blocks. The advanced controls application allows trimming and linearizing motion; the analyzing and designing of controllers; generation of the code from the controller modules; and simulation of the controller on dedicated hardware. The complete model can be downloaded as a unit, or separately – as the controller and the plant. In order to simulate the model in real-time, the model should be broken down into parts (each to be simulated by its own model). MATLAB/ Simulink provide the mathematical dynamics of the model which can further be written in more than one programming language.

Although SimMechanics is a simulation package, it can also be used with hardware in the loop for control, but it is not real-time.

3.9. ROBOT PROGRAMMING LANGUAGE

In order to program a robot using a programming language, the kinematics and dynamics of the robot must be available. The program to be written for the robot depends on the dynamics and kinematics of the particular robot to be programmed. The software developed for robots consists of controlling modules. A communication link should be established between the user and the robot in

order for the user to direct the robot to execute the given task. High-level programming is one of the approaches that is used to solve the communication problem between the user and the robot. This communication is further complicated as the objects to be manipulated by the robot are three-dimensional in nature and have a variety of physical properties; robots operate in a spatially complex environment; and sensory information has to be monitored, manipulated and properly utilized (Fu Gonzalez and Lee, 1987:451). One of the approaches to controlling the robot is the use of the object-oriented programming and robot programming. Robot programming focuses on the motions and task oriented programming describes the assembly tasks. AL and AML programming languages were used as stated in Fu's book as examples for position specification, motion specification and sensing. Figure 3.1 is a brief summary of the AL and AML programming languages.

AL was developed by Stanford University. Currently AL can be executed on a VAX computer and real-time control of the arms is performed on a stand alonePDP-11. Its characteristics are:

- High-level language with features of ALGOL and Pascal
- Supports both robot-level and task-level specifications
- Compiled into low-level language and interpreted on a real time control machine
- Has real-time programming language constructs like synchronization, concurrent execution, and on-conditions
- ALGOL like data and control structure
- Support for world modeling

AML was developed by IBM. It is the control language for the IBM RS-1 robot. It runs on a Series-1 computer (or IBM personal computer) which also controls the robot. The RS-1 robot is a Cartesian manipulator with 6 degrees of freedom. Its first three joints are prismatic and the last three joints are rotary. Its characteristics are:

- Provides an environment where different user-interface can be built
- Supports features of LISP=like and APL=like constructs
- Supports data aggregation
- Supports joint-space trajectory planning subject to position and velocity constraints
- Provides absolute and relative motion
- Provides sensor monitoring that can interrupt motion

Figure 3.1: A brief summary of the AL and AML programming languages (Fu et al, 1987:453)

Although AL has real-time programming language and AML is able to provide sensor monitoring, it is tedious and cumbersome to use these robot programming languages. In order to complete the task, the user is required to program each robot motion required to complete the task. High-level language can be used to overcome this problem as it will be transformed into a robot-level program to accomplish the required task. Figure 3.2 shows the structure of a possible task planner.

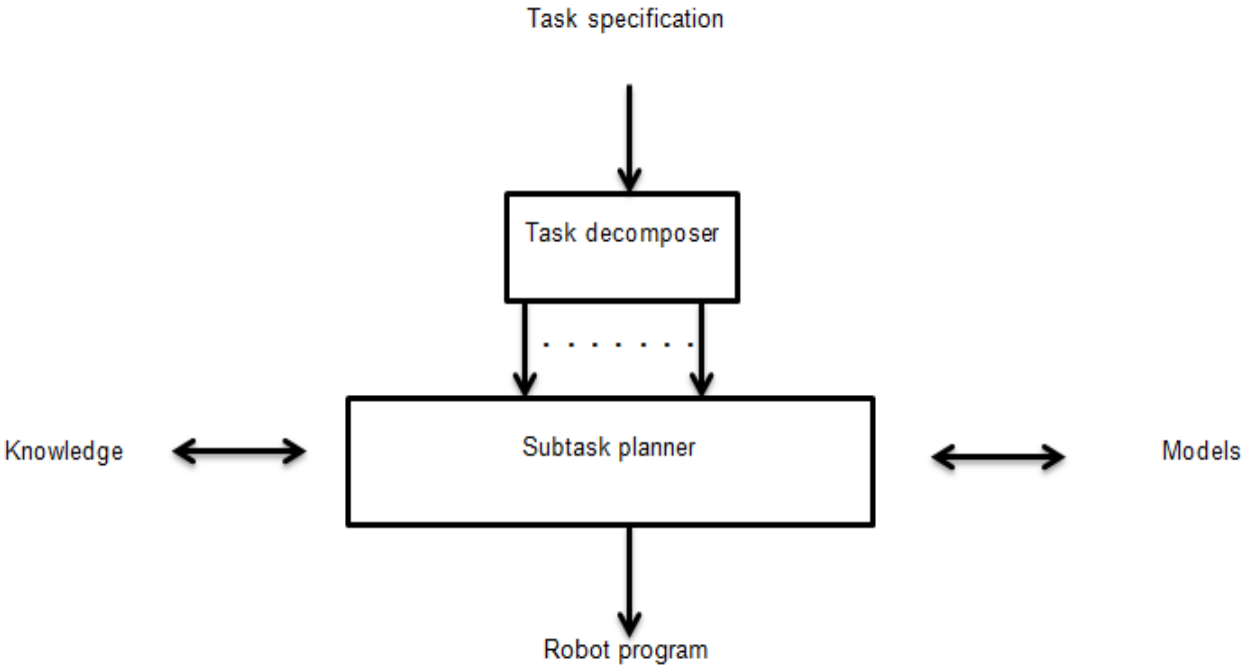

Figure 3.2 Task Planner (Fu, 1987:464).

High level language uses commands to complete the specified task at hand. At the highest level one would like to have natural languages as the input, without having to give the assembly steps (Fu et al, 1987:466).The programmer needs to know the programming language and required steps to execute the required steps, and need not worry about how the computer is going to execute those steps. The programmer can easily interpret and modify the program.

3.10. SOFTWARE ARCHITECTURE

Software architecture is an abstraction of the runtime behaviour of a software system during some phase of its operation (Kumar, Srivastava and Bajpai, 2007:298). The system architectural description determines the properties of that particular system, e.g. user interface. This is determined by the architectural style. There are several architectural styles that exist. Kim and Garlan (2007:5) highlight the following significant benefits of architectural styles:

1. Styles promote design reuse, since the same architectural design is used across a set of related systems.

2. Styles can lead to significant code reuse. For example, many styles (like J2EE or .Net) provide prepackaged middleware to support connector implementations.

3. It is easier for others to understand a system's organization if standard architectural structures are used.

4. Styles support interoperability, which is one of the requirements for the controller.

Design vocabulary and constraints are required when designing a particular style. To define an architecture style, design vocabulary can be specified as a set of components. Constraints may define what is allowed when configuring the elements of the particular style. Constraints are therefore used to restrict the possible configurations that may be created, using the design vocabulary (Kim and Garlan, 2007:7).

For the purpose of this study, object oriented architecture will be considered. Objects can be manipulated easily by the hardware and software. This allows for a high-level user interface as they provide a high-level description.

3.11. PROGRAMMING SOFTWARE CONTROLLER

The main aim when programming a software controller is to write an application that uses modules. The use of modules makes it possible to achieve the change of functionality at a software level. The design of the software controller should allow fast transformation from one module to the next – so fast, in fact, that it does not slow down the production time. The software that is not modular in structure does not allow rapid changes and is not flexible enough. The process of reconfiguration at a software level should take place seamlessly.

Human interaction is required at user interface level to create the modules. This type of system will provide customized flexibility and will be open-ended – so that it can be improved, upgraded, and reconfigured, rather than replaced (Mehrabi, Ulsoy and Koren, 2000:404). Reconfigurable software controller is meant to be used and reused over and over again. Replacing the software is expensive, as new software might require new hardware in order to be implemented and therefore some hardware to be left redundant. The modular nature of the software architecture makes it easy to add modules to accommodate changes (Kumar et al, 2007:303).

4. KINEMATIC AND DYNAMIC MODELLING

4.1 OVERVIEW

This chapter specifies the systematic process to be used to carry out the project. The controller to be developed is for a serial manipulator (robot) in Java. The controller must be able to control manipulators with up to five (5) degrees of freedom.

A Denavit Hartenberg (DH) forward kinematics will be used. The user will be able to specify the architectural configuration characteristics of the model.

4.2 INTRODUCTION

A software controller is developed by using programming language. Graphical User Interface (GUI) is designed to allow the user to input the coordinates of the robot and to output the robot arm position. The arm position of the robot is based on the input captured. Modules are used to enable code reuse.

4.3 KINEMATIC MODELLING

4.3.1 KINEMATIC CHAIN

Robotics structure consists of a number of links and joints, of which joints can either be prismatic (translating) or revolute (rotational) joints. The prismatic and revolute joints are the ones that are common in manipulators. Rotation about the *x*-axis is made possible by a revolute joint, while the sliding along an x-axis is

made possible by a prismatic joint (which does not permit rotation). Links are joined by these joints to create a serial link manipulator. A material body with two or more kinematic elements is called a link (Hartenberg and Denavit, 1964:52). The links are numbered from 0 at the base to *N* at the end-effector (Orin and Schrader, 1984:66). When a joint and a link are paired together they form or constitute a one (1) degree-of-freedom mechanism. One end of the manipulator, the base, is attached to the ground and the other end, the end-effector, is free to move in space (Tsai, 1999:54).

Figure 4.1a shows an illustration of a revolute joint and can be used for specifying the rotational degrees of freedom which defines an axis of revolution. Figure 4.1b illustrates a prismatic joint and is used for specifying the translational degrees of freedom which defines an axis of translation. If the joint is prismatic, a positive force applied to the joint moves the follower in the positive direction along the axis of translation (SimMechanics™ User's Guide, 2010).

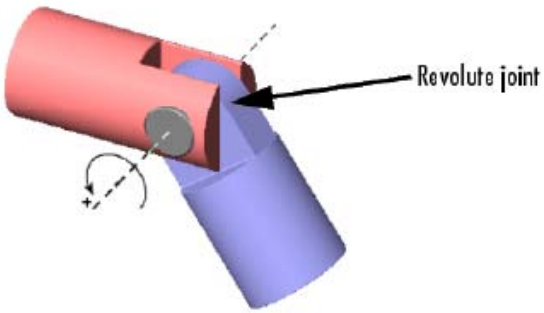

Figure 4.1a Revolute Joint (SimMechanics™ User's Guide, 2010)

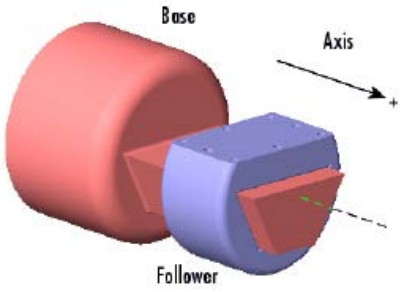

Figure 4.1b Prismatic Joint (SimMechanics™ User's Guide, 2010)

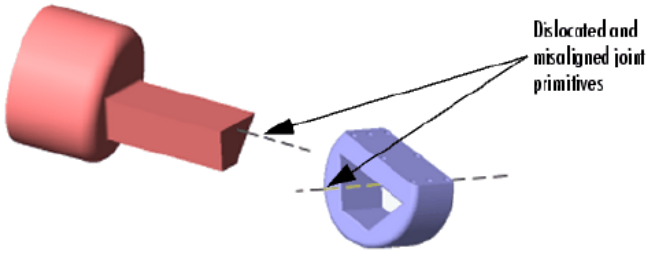

Figure 4.1c Disassembled Prismatic Joint (SimMechanics™ User's Guide, 2010)

4.3.2 KINEMATIC CALCULATIONS

The kinematic calculations of a robot provide the position of the end-effector of the robot arm. The position and the orientation of the end-effector (here called its pose) can be described by its generalized coordinates. These are usually the coordinates of a specific point of the end-effector and the angles that define its orientation, but may be any other set of parameters that allow one to define uniquely the pose of the end-effector (Merlet, 2006:1).

The robot arm motion behaviour is described by the set of mathematical equations mentioned. The equations are useful in the computer simulation of the motion of the robot arm. There are different methods used in deriving and evaluating the dynamic equations of motion of a robot arm. The commonly used mathematical equation is a matrix algebraic, which comprises of quantitative measurements. This will be applied in this study. They are generally used in operational controlling and their application serves for control and steering of success potentials (Riba, Pérez, Ahuett, Sànchez, Domínguez and Molina, 2006:236 - 237).

The matrix has a structure, which can be 2 x 2 matrix, 3 x 2 matrix, etc. If a matrix has structure, then it is usually possible to exploit it (Golub and Van Loan, 1996). A 2 x 2 matrix can be displayed as follows:

$$\begin{Bmatrix} A & B \\ C & D \end{Bmatrix}$$

This type of matrix is a two-dimensional rotation matrix which operates in a two dimensional space and it maps its coordinates in a specified rotated coordinate system.

This concept can be used in theory and also in practice. By substituting the coordinates, one is able to determine the corresponding points. For this project, a 4 x 4 homogenous transformation system needs to be used which will enable rotation as well as translation. The homogeneous transformation matrix is a 4 x 4 matrix that is defined for the purpose of mapping a homogeneous position vector from one coordinate system into another (Tsai, 1999:43).

4.4 DH RESPRESENTATION

The motivation for using Hartenberg and Denavit (DH) method is that it is a commonly used convention for selecting frames of reference in robotics applications. The objectives are to derive the forward kinematics solution for two (2) to five (5) degrees of freedom (DOF). The configurations consist of different combinations that result from using prismatic and revolute joints in two (2) to five (5) DOF mechanisms.

Denavit and Hatenberg (1955) proposed a systematic and generalized approach of utilizing matrix algebra to describe and represent the spatial geometry of the links of a robot arm with respect to a fixed reference frame (Fu et al, 1987, 6). The proposed method uses a 4 x 4 homogenous transformation matrix. The use of DH transformation enables the derivation of robot arm dynamic equations. These equations are the ones that will be used as a set of mathematical calculations – to simulate the robot arm motion used for computer simulation. Four parameters are used to describe each successive joint and link pair – the joint angle (θ_i) and offset distance (d_i) as well as the link length (a_i) and twist (α_i) (Orin and Schrader, 1984:67).

4.5 THE STEPS OF TRANSFORMATION USING DH PARAMETERS

The joint axis will be represented by *i*. The joint is found where two links connect. The parameter d_i is the shortest distance measured along the common normal between the joint axes (i.e., the z_{i-1} and θ_i axes for joint *i* and joint *i* + 1, respectively), and α_i is the angle between the joint axes measured in a plane perpendicular to θ_i. d is the distance between adjacent links. θ_i is the angle between adjacent links.

The joint angle rotates about $z_{i-1,}$ the link offset (d) translates along z_{i-1}. The link length (a) translates along x_i and link twist (α_i) rotates about x_i.

The DH transformation matrix for adjacent coordinate frames is expressed by the four operations of rotation, translation, translation and rotation by basic homogeneous matrix in Figure 4.1e for forward kinematics. Figure 4.2a displays a translation about $z_{i-1}1$ axis for a distance d_i. Figure 4.2b displays a rotation about z_{i-1}axis for θ_i angle. Figure 4.2c displays a translation about x_i axis for a distance a_i . Figure 4.2d displays a rotation about x_i axis for α_i angle. The $^{i-1}\mathbf{T}_i$ matrix gives both position and orientation changes between successive coordinate systems (Orin and Schrander, 1984:67). These four operations are expressed as homogeneous matrix displayed in Figure 4.2e Transformation Matrix.

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_{z,d}$$

Figure 4.2a Translation about z_{i-1} for distance d

$$= \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{T}_{z,\theta}$$

Figure 4.2b Rotation about z_{i-1} for angle θ

$$= \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_{x,a}$$

Figure 4.2c Translation about x_i distance a

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}_{x,\alpha}$$

Figure 4.2d Rotation about x_i for α angle

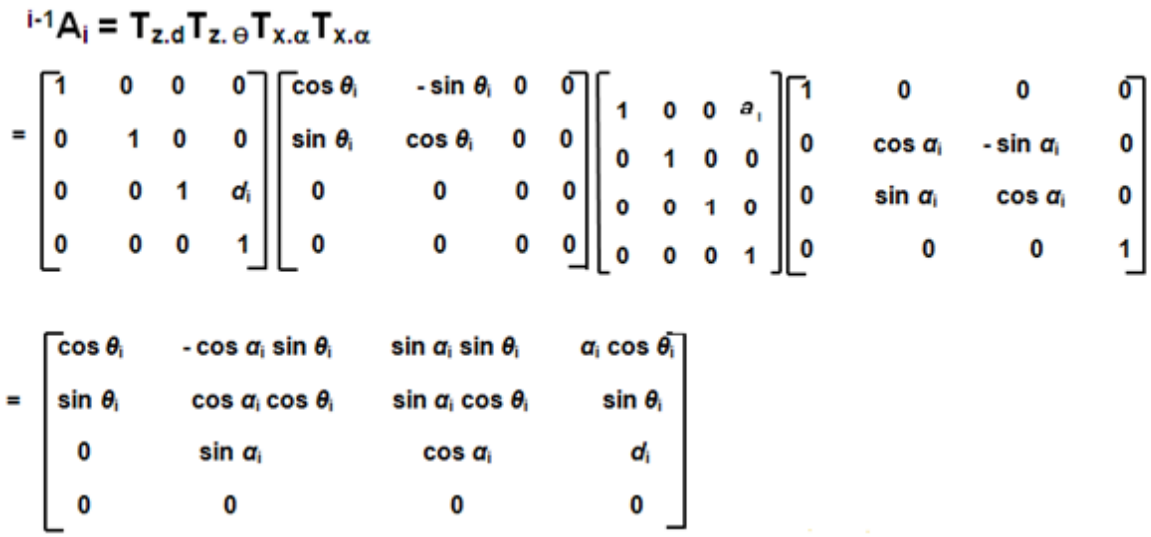

Figure 4.2e Transformation Matrix (Fu et al, 1987, 40)

The DH transformation matrix uses only four variables, α_i, θ_i, d_i and a_i. For a revolute joint, α_i, d_i and a_i are constants and θ_i represents a joint variable. For a revolute joint, α_i, θ_i and a_i are constants and d_i represents a joint variable.

The DH transformation matrix represents each link's coordinate system at the joint with respect to the previous link's coordinate system. The end-effector can be transformed and expressed in the "base coordinates", which is the reference frame.

4.6 FORWARD KINEMATICS

Forward kinematics is also referred to as direct kinematics. With forward kinematics, the joint angles and geometric link parameters are given. With respect to the reference coordinate system, one needs to determine the orientation and position of the end-effector of the manipulator. Murray, Li and

Sastry (1994: 83) defines the forward kinematics of a robot as determining the configuration of the end-effector (the gripper or tool mounted on the end of the robot) given the relative configurations of each pair of adjacent links of the robot. This problem is referred to as forward kinematics. Errors that can arise in kinematics can put strong limitations on displaying the correct output for the end-effector. The accumulation of these errors could lead to the failure of executing nominal tasks (Wang and Chirikjian, 2006: 95).

Figure 4.3 is an illustration of direct (forward) kinematics. The figure demonstrates the input and output parameters for forward kinematics. The input is joint angles and link parameters, and the output displays the orientation and position of the end-effector with respect to the reference coordinate system.

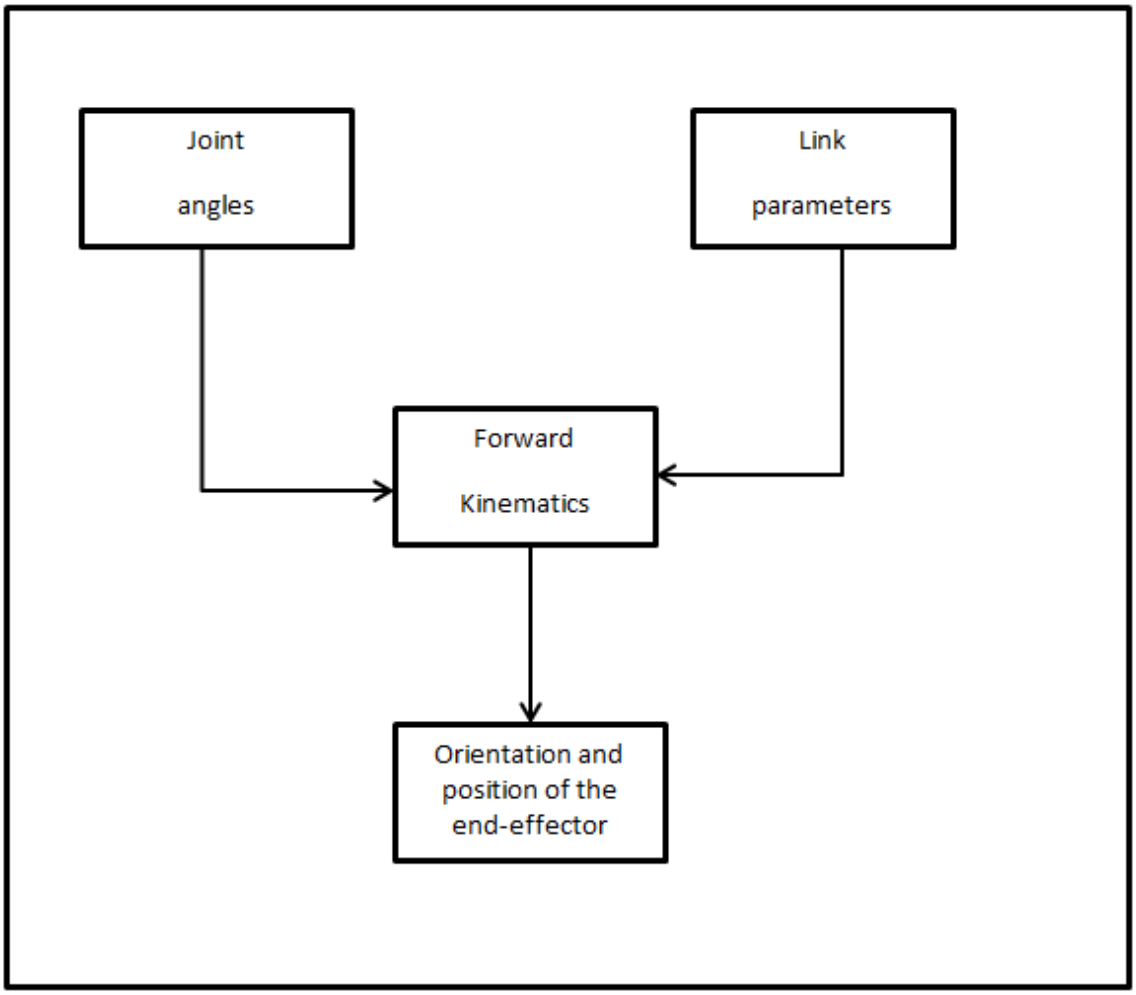

Figure 4.3 Forward Kinematics

Firstly there should be a robot to work with. Each robot has its own architectural style. Possible robot architectures are summarized in the table below. The robot can only consist of one of the architectural styles specified with revolute and/or prismatic joints. Table 4.1, Architectural Styles displayed, shows the results from 1 DOF robot to 3 DOF. The complete table is available in Appendix A.

Each joint-link pair represents a one degree-of-freedom. For *n* number of degrees in a manipulator, there is *n* number of joint-link pairs. Table 4.1 for example, in 1 DOF robot, shows that the robot can have either revolute or

prismatic joints. Prismatic and revolute joints each constitute 1 DOF – hence, for 1 DOF there is 1 joint-link pair. Where there are two joints, it simply means that the architecture of the robot consists of 2 DOF, etc. However, there are other joints which exist that constitute more than 1 DOF.

The configurations are calculated as follows:

For *n* DOF, where *n* represent a number for number of DOF

2^n (2 to the power of *n*)

e.g., if $n = 2$

Therefore

$2^2 = 4$

Hence for 2 DOF, there are only 4 possible configurations. Summary of configurations is displayed in Table 4.2 Architectural Styles. The joints are labeled as *R* for revolute joint and *P* for prismatic joint. As joints are labeled from the base upwards, the architecture *RPR* represents 3 joints in a sequence of Revolute-Prismatic-Revolute.

Table 4.1 Architectural Styles

Degrees of Freedom	Architectural Style	Simplified Architecture
1 Degree-of-freedom	Revolute	*R*
	Prismatic	*P*
2 Degrees of freedom	Revolute, Revolute	*RR*
	Revolute, Prismatic	*RP*
	Prismatic, Revolute	*PR*
	Prismatic, Prismatic	*PP*
3 Degrees of freedom	Revolute, Revolute, Revolute	*RRR*
	Revolute, Revolute, Prismatic	*RRP*
	Revolute, Prismatic, Revolute	*RPR*
	Revolute, Prismatic, Prismatic	*RPP*
	Prismatic, Prismatic, Revolute	*PPR*
	Prismatic, Prismatic, Prismatic	*PPP*
	Prismatic, Revolute, Revolute	*PRR*
	Prismatic, Revolute, Prismatic	*PRP*

Table 4.2 Summary of Possible Configurations, displays possible configurations for each number of DOF. Column 1 is the number of DOFs. The second column displays the calculations for each configuration. The third column displays the total number of all possible configurations for each number of DOF.

Table 4.2 Summary of Possible Configurations

POSSIBLE CONFIGURATIONS		
Degrees of Freedom	**Calculations**	**No. of Configurations**
1 DOF	2^1	2
2 DOF	2^2	4
3 DOF	2^3	8
4 DOF	2^4	16
5 DOF	2^5	32

4.7 DYNAMIC MODELING

The dynamic modeling of the system makes predictions of the capabilities of the system over time. Before any money is invested for creating the actual system, it is recommended to start with system modeling. The model will help explain how the system will perform when it is implemented. Dynamic modeling provides a way for observing the system behaviour before it is even implemented. This also reduces the development time.

It's difficult, and sometimes impossible, to make changes when the system has already been implemented. It is not only expensive, but it also takes a long time to implement the changes once the system has been created.

The dynamic modeling in this case uses mathematical concepts to model the behaviour of the system. The dynamic modeling helps to see the output of the system before it is implemented. It is simple to make, and track, changes using the mathematical model.

The system software becomes complex and difficult to manage when the number of parameters increases. The fewer the number of joints, the fewer number of parameters that needs to be managed. The parameters are increased as the number of joints increases. De-Jiu Chen (2001:50) pointed out in his thesis that managing software complexity and flexibility is a challenging area. With increased number of parameters, the functions of the system also increase. Programming these parameters becomes a complex process when there are many parameters.

One can use DH transformation matrix to predict the behaviour of the system. If given the robot arm link coordinate parameters, a correct matrix could be used to find the solution for the position of the end-effector. For 1 DOF, the basic DH transformation matrix is used. There is a need to identify the type of joint, either prismatic or revolute. For 2 DOF there is a need to use the basic matrix to find the position of the end-effector with the use of all arm link coordinate parameters, depending on the number of joints. In this case, each joint should be identified so as to establish the correct solution.

The need to develop a unique matrix to evaluate each element for the systems that has prismatic and revolute joints was satisfied. The models were developed for different degrees of freedom using MATLAB (Appendix B) to compute all the coordinate transformation matrixes before they were implemented into a system using a programming language. The model helps identify the different parameters for each joint depending on the number of DOF.

4.8 DYNAMIC COMPUTATIONS

Key metrics are defined as quantitative measurements that give useful information related to measurable facts through aggregation and relativization (Riba et al. 2006:236). The metrics were computed using DH representation to develop the model that will be used. Forward kinematics is used for transformation for the end-effector coordinates and joint coordinates.

The developed model was computed using variables that can be easily identified for programming purpose. The variables are used as follows in the computations for calculating the end-effector in the mathematical model. Although these variables can be used as they are in programming language, it is always advisable to use meaningful names for variables. This means that when the program needs to be modified by another person, they will easily be able to understand and identify what variables are used without reading supplementary information or details on what they are used for.

cθ = cos θ (cos theta)

cα = cos α (cos alpha)

sθ = sin θ (sin theta)

sα = sin α (sin alpha)

These variables are used for each DOF. For 1DOF, the subscript of 1 along the variable is used; for 2DOF the subscript of 2; and a subscript of 12 along the variable simply represents that the variable will be used for 1 DOF and 2 DOF and both will be multiplied together to get the result, e.g., sθ_{12} represents sθ_1 x

sθ_1. This makes it easy to identify what the variables are used for. The solution for 2 DOF is established by multiplying 2 basic homogenous matrices to enable evaluation of each element in the resultant matrix. This enables the input for each element so that all coordinates can be computed in order to find the desired position for the end-effector. Using this method, each element is computed accordingly.

For each problem there is a computed matrix used to find the solution for the end-effector. The computation and communication underpinning a given system invariably reflects a style (Kumar et al, 2007:297). When there are more joints that needs to be calculated in a system, the matrix to be used increases in size and becomes complex to manage. These matrices displayed make it easier to find the position of the end-effector. These matrices are the solutions for Direct Kinematic solution for 2 DOF to 5 DOF.

The computations for 2 DOF to 5 DOF were formulated as follows:

$$\mathbf{T}_1 = {}^0\mathbf{A}_5 = {}^0\mathbf{A}_1\ {}^1\mathbf{A}_2 = \begin{pmatrix} nx & sx & ax & px \\ ny & sy & ay & py \\ nz & sz & az & pz \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1)$$

4.8.1. COMPUTATION FOR 2 DOF

$$nx = c\theta_{12} - c\alpha_1 s\theta_{12} \quad (2)$$

$$sx = s\alpha_{12}s\theta_1 - c\alpha_2 c\theta_1 s\theta_2 - c\alpha_{12}c\theta_2 s\theta_1 \quad (3)$$

$$ax = c\alpha_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\alpha_2 s\theta_1 \quad (4)$$

$$px = a_1 c\theta_1 + a_2 c\theta_{12} + \underline{d}_2 s\alpha_1 s\theta_1 - \underline{a}_2 c\alpha_1 s\theta_1 \quad (5)$$

$$ny = c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2 \quad (6)$$

$$sy = c\alpha_{12}c\theta_{12} - c\alpha_2 s\theta_{12} - c\theta_1 s\alpha_{12} \quad (7)$$

$$ay = s\alpha_2 s\theta_{12} - c\alpha_2 c\theta_1 s\alpha_1 - c\alpha_1 c\theta_{12} s\alpha_2 \quad (8)$$

$$py = a_1 s\theta_1 + a_2 c\theta_2 s\theta_1 - c\theta_1 d_2 s\alpha_1 + a_2 c\alpha_1 c\theta_1 s\theta_2 \quad (9)$$

$$nz = s\alpha_1 s\theta_2 \quad (10)$$

$$sz = c\alpha_{12}s\alpha_{21}c\theta_2 \quad (11)$$

$$az = c\alpha_{12} - c\theta_2 s\alpha_{12} \quad (12)$$

$$pz = d_1 + c\alpha_1 d_2 + a_2 s\alpha_1 s\theta_2 \quad (13)$$

$$\mathbf{T}_1 = {}^0\mathbf{A}_5 = {}^0\mathbf{A}_1\ {}^1\mathbf{A}_2\ {}^2\mathbf{A}_3 = \begin{pmatrix} nx & sx & ax & px \\ ny & sy & ay & py \\ nz & sz & az & pz \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (14)$$

4.8.2. COMPUTATION FOR 3 DOF

$$nx = c\theta_3(c\theta_{12} - c\alpha_1 s\theta_{12}) - s\theta_3(c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) \quad (15)$$

$$sx = s\alpha_3(c\alpha_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\alpha_2 s\theta_1) - c\alpha_3 s\theta_3(c\theta_{12} - c\alpha_1 s\theta_{12}) - c\alpha_3 c\theta_3(c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) \quad (16)$$

$$ax = c\alpha_3(c\alpha_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\alpha_2 s\theta_1) + s\alpha_3 s\theta_3(c\theta_{12} - c\alpha_1 s\theta_{12}) + c\theta_3 s\alpha_3(c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_1 c\alpha_2 c\theta_2 s\theta_1) \quad (17)$$

$$px = a_1 c\theta_1 + d_3(c\alpha_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\alpha_2 s\theta_1) + a_3 c\theta_3(c\theta_{12} - c\alpha_1 s\theta_{12}) + a_2 c\theta_{12} + d_2 s\alpha_1 s\theta_1 - a_3 s\theta_3(c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) - a_2 c\alpha_1 s\theta_1 s\theta_2 \quad (18)$$

$$ny = c\theta_3(c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) - s\theta_3(c\theta_1 s\alpha_{12} + c\alpha_2 s\theta_1 s\theta_2 - c\alpha_1 c\alpha_2 c\theta_{12}) \quad (19)$$

$$sy = - s\alpha_3(c\alpha_2 c\theta_1 s\alpha_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\alpha_2) - c\alpha_3 s\theta_3(c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) - c\alpha_3 c\theta_3(c\theta_1 s\alpha_{12} + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) \quad (20)$$

$$ay = s\alpha_3 s\theta_3(c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) - c\alpha_3(c\alpha_2 c\theta_1 s\alpha_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\alpha_2) + c\theta_3 s\alpha_3(c\theta_1 s\alpha_{12} + c\alpha_2 s\theta_1 s\theta_2 - c\alpha_{12} c\theta_{12}) \quad (21)$$

$$py = a_1 s\theta_1 - d_3(c\alpha_2 c\theta_1 s\alpha_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\alpha_2) + a_3 c\theta_3(c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) + a_2 c\theta_2 s\theta_1 - c\theta_1 d_2 s\alpha_1 - a_3 s\theta_3(c\theta_1 s\alpha_{12} + c\alpha_2 s\theta_1 s\theta_2 - c\alpha_{12} c\theta_{12}) + a_2 c\alpha_1 c\theta_1 s\theta_2 \quad (22)$$

$$nz = s\theta_3(c\alpha_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2 \quad (23)$$

$$sz = s\alpha_3(c\alpha_1 c\alpha_2 - c\theta_2 s\alpha_{12}) + c\alpha_3 c\theta_3(c\alpha_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23} \quad (24)$$

$$az = c\alpha_3(c\alpha_{12} - c\theta_2 s\alpha_{12}) - c\theta_3 s\alpha_3(c\alpha_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + s\alpha_{13} * s\theta_{23} \quad (25)$$

$$pz = d_1 + c\alpha_1 d_2 + d_3(c\alpha_{12} - c\theta_2 * s\alpha_{12}) + a_3 s\theta_3(c\alpha_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + a_2 s\alpha_1 s\theta_2 + a_3 c\theta_3 s\alpha_1 s\theta_2 \quad (26)$$

The formulated computations that will be used for programming have been displayed in Appendix C from 2DOF to 5 DOF.

4.9 CONCLUSION

In this chapter the structure of the robot manipulator is clarified as consisting of a combination of either prismatic and/or revolute joints. One joint constitutes 1 DOF. The basic homogenous transformation matrix was identified which permits translation as well as rotation by prismatic and revolute joints. The steps were carried out for using transformation matrix. The table for architectural styles is presented, to show possible architectures. The solutions for 2 and 3 DOF are presented in this chapter which will be used when programming the controller.

5 PROGRAMMING SOFTWARE CONTROLLER

5.1 CHAPTER OVERVIEW

In this chapter we look at programming software controller and analysis of the results found.

Graphical User Interface (GUI) is designed to allow the user to input the coordinates of the robot and to output the robot arm position. The arm position of the robot is based on the input captured. The code is written in a programming language selected and tested on a PC. Modules are used to enable code reuse.

5.2 SOFTWARE DESIGN

With regards to the design, the system must firstly allow the user to specify the architectural style to be used. The method chosen should be as easy as possible for the user to understand. When the system is easy to use, the user is less likely to make mistakes. An easy to understand and easy to use system reduces the errors that the user might make. The Graphical User Interface has been identified as one of the communication links that is the most convenient, easy and understandable method to use.

The mathematical model developed will be transformed using a programming language. The programming language to be chosen will enable the openness of the controller.

5.3 SELECTING PROGRAMMING LANGUAGE

There are three (3) programming languages that are capable of developing an open architecture controller, and all were considered. C, C++ and Java were considered for this project. C is a procedural language. With a procedural language, the code breaks into functions. C is a low-level language, like a machine language. Most users find it difficult to interpret C because it is closer to a machine-level code. Pure virtual function is enforced in order to create an interface in C++. C++ language is the extension of C programming language. Due to this fact, C was no longer considered for this project.

C++ programming language is an object oriented language. C++ language requires one to create reference to objects. C++ is more complex and error prone (as compared to Java language). Although these two (2) programming languages are both object oriented programming languages, Java was derived from C++.

The Java Virtual Machine (JVM) loads, verifies and executes the byte code of a Java program (Kumar et al, 2007:299). Java language is more portable as compared to other languages. Java Software platform is architecture neutral. Through the Java Virtual Machine (JVM), the same application is capable of running on multiple platforms as compared to other software platforms. Java programming language is also a high-level programming language. A high-level language allows one to write English-like statements that correspond to machine instructions. It is also a communication link between the user and the system. Java software is the most commonly used software for portable devices such as portable phones. It is been selected for this project due to its many advantages in the development of an open architecture controller, as compared to C and C++.

In the programming of the robot manipulator a typical set of desired positions and orientations, and perhaps the time derivatives of the positions and orientations of the end-effector, are specified in space (Tsai, 1999:46). Java language is able to derive the joint variables to bring the end-effector to the desired position.

5.4 GRAPHICAL USER INTERFACE

The GUI is developed around the following points:

1. The system should permit the user the liberty to exit at any time, and allow the user to terminate the system at any stage.
    a. The system should not force a user to complete the task before it can be terminated or exited.

2. The system status should always be visible, in order to communicate to the user what is required.

3. The user should always be aware of what is required at any stage in the system.
    a. If the GUI is difficult to use and the user does not understand what is required, the user might get frustrated and exit without completing the desired task.

4. The coordinate parameters should be displayed along with the output so as to enable comparison of input with the output on one screen.

5.5 DEVELOPING THE GRAPHICAL USER INTERFACE

There are number of techniques that can be used to develop the graphical user interface. The graphical user interface developed uses the command buttons as a communication system – to process the input and to move to the next step or stage in a system. When clicked, the command buttons initiate an immediate action. These buttons where selected instead of using a menu. A menu needs to be clicked twice to initiate an action that needs to be invoked.

The option buttons and drop-down boxes were possible options that could be used where only one option is possible at a time. The use of option buttons needs all options to be visible on the GUI at the same time. This occupies a lot of space on the GUI, especially if there are many options to choose from. With the use of drop-down boxes, only one option is visible at a time. This can be considered as default option should the user not select any option available from the drop-down options. With the option buttons there is a fear that there might not be a default option and that may result in an error on system level.

The drop-down boxes are used to help the user not choose more than one option at a time. This assists in eliminating the error which occurs when a user selects more than one option (should the option buttons not function correctly, as it should not be possible to have more than one option selected at a time).

The textboxes were selected for the user to input the different robot coordinates. Listbox is used to display the end-effectors' position. The GUI for selecting number of DOF uses command buttons. The command buttons assist for not selecting more than one option. Figure 4.3 is a screen shot where DOF is

specified by clicking on a number to indicate the number of DOF. The same design displayed is applied throughout the system.

5.5.1 WELCOME SCREEN

The welcome screen is used to welcome the user to the system by the use of a welcome message. The title bar clearly explains that the system is for an open architecture controller. This heading appears consistently on every screen. There are two buttons which are clearly visible on this screen; EXIT and PROCEED buttons.

The user has the liberty of exiting the system at this point by clicking the EXIT button. The clear visibility of exit button assures the users that they are not forced to use, or to proceed with using, the system. Should the EXIT button be clicked, the message box will appear to ensure that it was not selected by mistake. This is to make sure that the system is not exited by mistake. Upon clicking the EXIT button, the message box displays a message to confirm whether the user wants to exit the system or not. This measure is applied throughout the system to ensure stability. This message box displays 2 buttons; YES and NO. Upon clicking YES, the system is terminated. By clicking NO, the user stays in the same window.

Clicking the NEXT button ensures that the user proceeds to the next stage of using the system. The system guarantees that the user proceeds at their own will, by clicking on PROCEED button.

Figure 5.1 Welcome Screen, shows a screen that welcomes the user to the system. The screen provides the user with two options: to exit the system by clicking EXIT button, or to proceed using the system by clicking NEXT button. Objects used are identified and labeled as follows for clarification: Title Bar, Welcome Message, Exit Button and Next Button.

Figure 5.1 Welcome Screen

5.5.2 SELECTING THE DEGREES OF FREEDOM

To explore the openness of the system, the system should allow the user to select the number of degrees-of-freedom that will be used to input link coordinate parameters of the robot in question. The user is not limited to use one (1) model, but the models that can be selected are limited to five (5) DOFs. The number of DOFs is determined by the number of joint-link pairs and will always be equal in this case. These can be explored at the design level.

Figure 5.2 below demonstrates the number of DOFs for the specific robot that will be used is specified. The interface screen for selecting the number of DOFs allows the user of the system to click on the command button box to select number of DOFs. Icons with numbers are used as the buttons.

Upon selecting the appropriate DOFs by clicking on the number displayed, the user will be directed to the architecture selection screen. This minimizes the number of clicks required and saves time. This also eliminates the learning of the systematic process for using the system, or looking for what should be done after selecting the DOFs, e.g., pressing the PROCEED button, etc. Customizing the design of the system starts where the number of DOFs is specified. The other advantage for selecting the number of DOFs is the flexibility that the system provides through re-configurability.

The EXIT command button allows the user to exit the system, and by clicking any of the DOFs command buttons the user is allowed to proceed to the next step or stage of the system. The DOFs command buttons are identified by numbers which represent the number of DOFs that will be used.

Figure 5.2 Number of DOFs Selection Screen

The code for selecting the number of degrees-of-freedom, when pressing the button, is as follows:

```
1        degreeOfFreedom = x;
2        new RobotTypeForm().setVisible(rootPaneCheckingEnabled);
3        this.dispose();
```

Line 1 is where the degreeOfFreedom is determined. The number of DOFs is equal to the number of degrees that will be selected by the user represented by an integer value *x*. The *x* variable gets the value when any of the buttons is

clicked. The value cannot be zero, or false, as this will prevent the robot type form GUI from been displayed on the screen.

Line 2 opens a new form and centers it in the same position as the current form (that is open). setVisible is set by default to display the new form. The Robot Type Form is displayed by executing this line of code.

Line 3, this.dispose(), closes the current form. The new form, that displays the robot type, becomes the main form that is displayed. These lines of code are used to determine the number of DOFs that will be used.

5.5.3 ROBOT ARCHITECTURE

The specification of the robot architecture is restricted by the number of DOFs selected for the robot to be used. The system does not at this stage allow the user to change the number of DOFs, unless the back button is used to go back to change the number of DOFs. The architecture cannot be more than the number of DOFs specified. The architecture can be a combination of prismatic and revolute joints, and the number of DOFs will be equal to the number of joints specified. As each joint-link pair represents a one degree-of-freedom, the system only allows the architecture to be specified based on the number of DOFs specified in the previous stage. This is to bring a sense of balance between the number of DOFs and the joint-link pairs. The system does not allow the number of joint-link pairs to be more than the number of DOFs. This can be explored at a system level. A configuration is determined when all of its components are selected (Bi, Lang, Verner and Orban, 2008:1241). Stole

Figure 5.1 specified earlier shows the section in which 1 DOF is selected following the preceding step. The information to be provided at this stage will always be aligned with the preceding step. The architecture to be specified will be specified by selecting relevant joints of prismatic and revolute. Figure 5.3 is a display of where 1 DOF was selected – it can be seen that the option for selecting the joint for 1 DOF is the only one that is active on the screen. This allows the machine to be reconfigured according to the specifications, and for the changeability in the architecture to take place (or be transformed easily) when the necessary changes have been determined. This customization provides better flexibility and it allows reconfiguration to respond quickly to sudden market demands and changes. The controller software needs to be reconfigured only when the physical configuration changes and the application specifications need to be altered when the product requirements change (Wang and Shin, 2002:476).

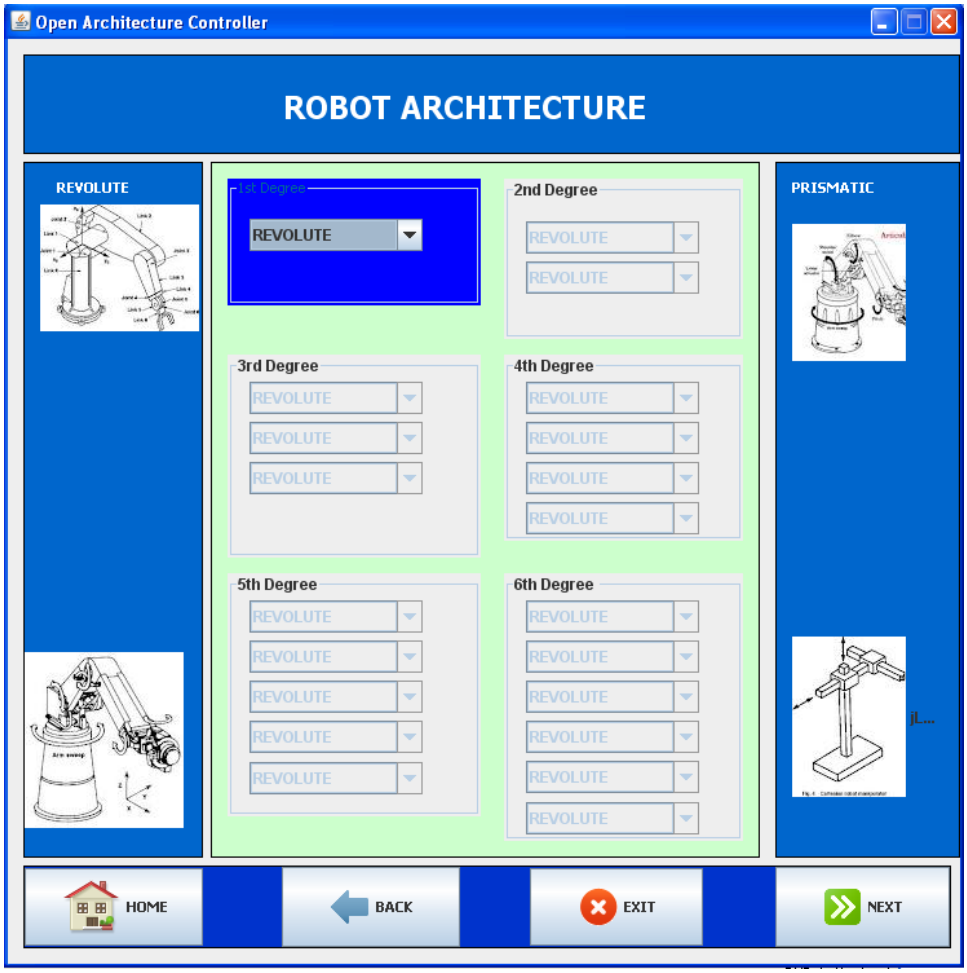

Figure 5.3 Robot Architecture

The revolute joint option is displayed as the default joint for all the joints. Should the joint be prismatic for any of the joints, the drop-down box needs to be clicked to show the prismatic joint option for selection as displayed below on Figure 5.4 Selecting Joint Type for 2 DOF. The figure shows where the second joint drop-down box is clicked and two options are displayed: revolute and prismatic. Only one option can be selected at a time. For any of the joints, the prismatic joint should be selected by clicking the drop-down button, or else revolute joint is selected by default and will be the one used for the architectural style. The selected joint is the one that is visible inside the box.

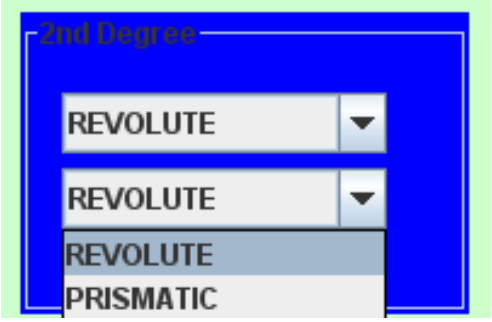

Figure 5.4 Selecting Joint Type

The code used for selecting the revolute or prismatic joints is as follows:

```
if(cBox1D.getSelectedItem().equals("REVOLUTE"))
 {
   robotArray[0] = 'R';
 }
   else if(cBox1D.getSelectedItem().equals("PRISMATIC"))
 {
   robotArray[0] = 'P';
 }
   if(robotArray != null)
 {
   new InputFrame().setVisible(true);
   this.dispose();
 }
```

5.5.4 ROBOT COORDINATE PARAMETERS

To be able to supply the coordinate parameters, the robot architecture should be selected first. Depending on the architecture selected, the coordinate parameters screen will be displayed as illustrated in Figure 4.8 Coordinate Parameters.

During configuration time, the geometric parameters of the components are set and linked with parameter expressions (Van Brussel, Sas, Nemeth, De Fonseca and den Braembussche, 2001:92).

Figure 5.2 illustrates where 1 DOF was selected, and the revolute joint was selected for the architecture. For revolute joint, the joint distance is set to zero for convenience and displayed appropriately. For prismatic joint, the joint angle is set to zero. By default the joint distance for a revolute joint is zero and the joint angle for prismatic joint is zero.

The user interface allows up to six sets of coordinates to be supplied for calculating the orientation and position of the end-effector of the manipulator. Depending on the number of DOFs and the architecture selected, the interface will be displayed accordingly. The coordinates that are zero by default are displayed to guide the user with regards to defaults and to minimize human error. The code for setting the default values is as follows:

```
if(RobotTypeForm.robotArray != null)
 {
   if(RobotTypeForm.robotArray[robot] == 'P')
  {
     if(robot == 0)
    {
                                                                txtJointAngle1.setText("0");


     }
    else if(robot == 1)
     {
                          txtJointAngle2.setText("0");
```

```
 }

}
      else if(RobotTypeForm.robotArray[robot] == 'R')
{
if(robot == 0)
{
                        txtJointDistance1.setText("0");

}
else if(robot == 1)
{
                        txtJointDistance2.setText("0");

}

}
   }
```

The code above verifies the architecture for each joint in order to check if the joint is a revolute or prismatic joint. When the joint is prismatic, the joint angle is set to zero and when it is revolute, the joint distance is set to zero (both by default).

The coordinates supplied at this stage are very crucial, as they are the ones that will determine the end-effector position. The input must be documented accurately because it is the one that will be used in the calculations. One error at this stage will affect the results of the position of the end-effector of the robot arm. The input needs to be verified before proceeding to the next phase of the system. This one step will ensure desirable and accurate results are achieved.

Figure 5.5 shows Coordinate Parameters, and the screen for inputting the robot coordinates is shown.

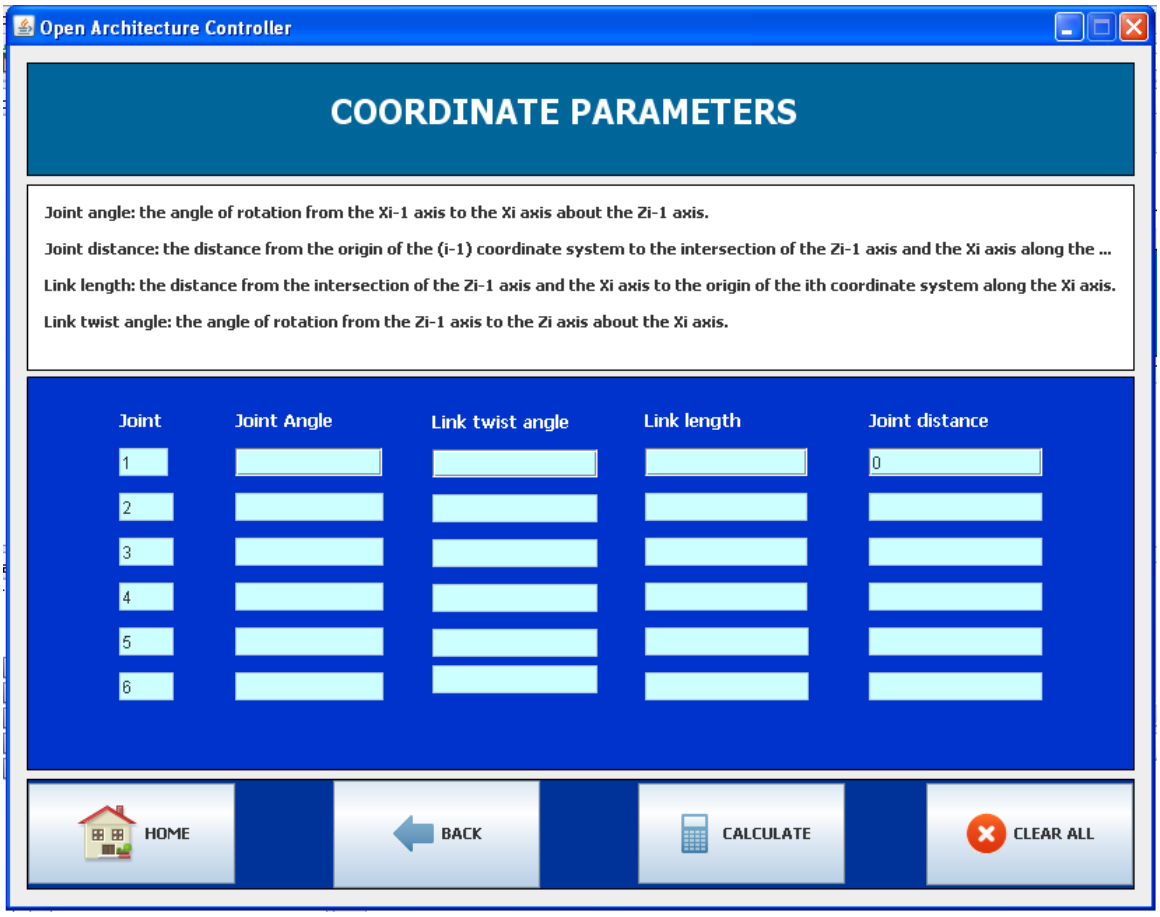

Figure 5.5 Coordinate Parameters

The results are stored in an array that holds four subscripts. The code for declaring an array is as follows:

```
double [ ] displayResults = new double[4];
```

A type double array is declared because the results that the system produces for the end-effector position have decimal places. The results are further rounded off

to the nearest 10. The system uses *Maths-sin* and *-cos*, where *Math.cos*(*variable*) method will calculate the *cos* of a variable declared and *Math.sin*(*variable*) method calculates the sin of the variable declared. The input is read from the textboxes for various variables and processed by using the relevant formula to produce the coordinate parameters in respect to the reference frame. The results are stored in an array as they are calculated.

The array has been declared as follows:

```
//Holds all the Joints provided by the user
public static ArrayList<Joints> jointsList = new ArrayList<Joints>();


//Holds all calculated Coordinates
public static ArrayList<Coordinates> coordinateList = new ArrayList<Coordinates>();


//Holds Coordinates as they are calculated
String[] messageDisplay = new String[4];
```

The code shows how the joints provided by the user are captured. It further describes the code that is used for holding the coordinates. The coordinates are calculated on the last line of code displayed.

The following example shows the code used on how the two DOFs are calculated in a program. The rest can be viewed in Appendix D.

1	double results = cosJointAngle1*cosJointAngle2 – cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2+ sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 - cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2 * sinJointAngle1+ cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1* sinJointAngle2+linkLength1*cosJointAngle1 + linkLength2*cosJointAngle1*cosJointAngle2 + jointDistance2*sinLinkTwistAngle1*sinJointAngle1 - linkLength2*cosLinkTwistAngle1*sinLinkTwistAngle2* sinJointAngle1;
2	displayResults[0] = Math.round(results);
3	results = cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2+ cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1 * cosJointAngle2 - cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1+ sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosJointAngle1*cosJointAngle2*sinJointAngle1 - cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2* sinJointAngle2+linkLength1*sinLinkTwistAngle1 + linkLength2*cosJointAngle2*sinJointAngle1 - jointDistance2*cosJointAngle1*sinJointAngle1 + linkLength2*cosLinkTwistAngle1*cosJointAngle1* sinLinkTwistAngle2;
4	displayResults[1] = Math.round(results);
5	results = sinLinkTwistAngle1*sinJointAngle2+ cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1 + cosJointAngle1 * cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2+ jointDistance1 + jointDistance2*cosJointAngle1 +

	linkLength2*sinLinkTwistAngle1*sinLinkTwistAngle2;
6	displayResults[2] = Math.round(results);
7	results = 0+0+0+1;
8	displayResults[3] = Math.round(results);
9	return new Coordinates (displayResults[0],displayResults[1],displayResults[2], displayResults[3]);

5.5.4.1 CODE EXPLANATION

Line 1 code calculates the first line of coordinates and the results are stored in variable results declared as double.

Line 2 code places the results in the first subscript on the array.

```
displayResults[x] = Math.round(results)
```

This method holds the subscript in an array called *displayResults* and the answer stored in variable called results is placed in an array at position x. *Math.round* method is used to round the results of the array to nearest 10.

Line 3 code calculates the second line of coordinates and the results are stored in variable results.

Line 4 code places the results in the second subscript on the array.

Line 5 and Line 7 of code calculates the third and fourth lines of the coordinate parameters and the results are stored in the third and fourth subscripts in the array with code that is on Line 6 and Line 8 respectively.

Line 9 code returns all the coordinates which are displayed as output after computing all the calculations. The results are returned using the following method:

```
return new Coordinates(displayResults[0],displayResults[1],displayResults
```

Four coordinates are returned that are being stored in an array and are displayed in Figure 5.6 Input and Output Coordinates. This figure displays the input parameters as well as the output coordinate parameters displaying the transformation matrix results.

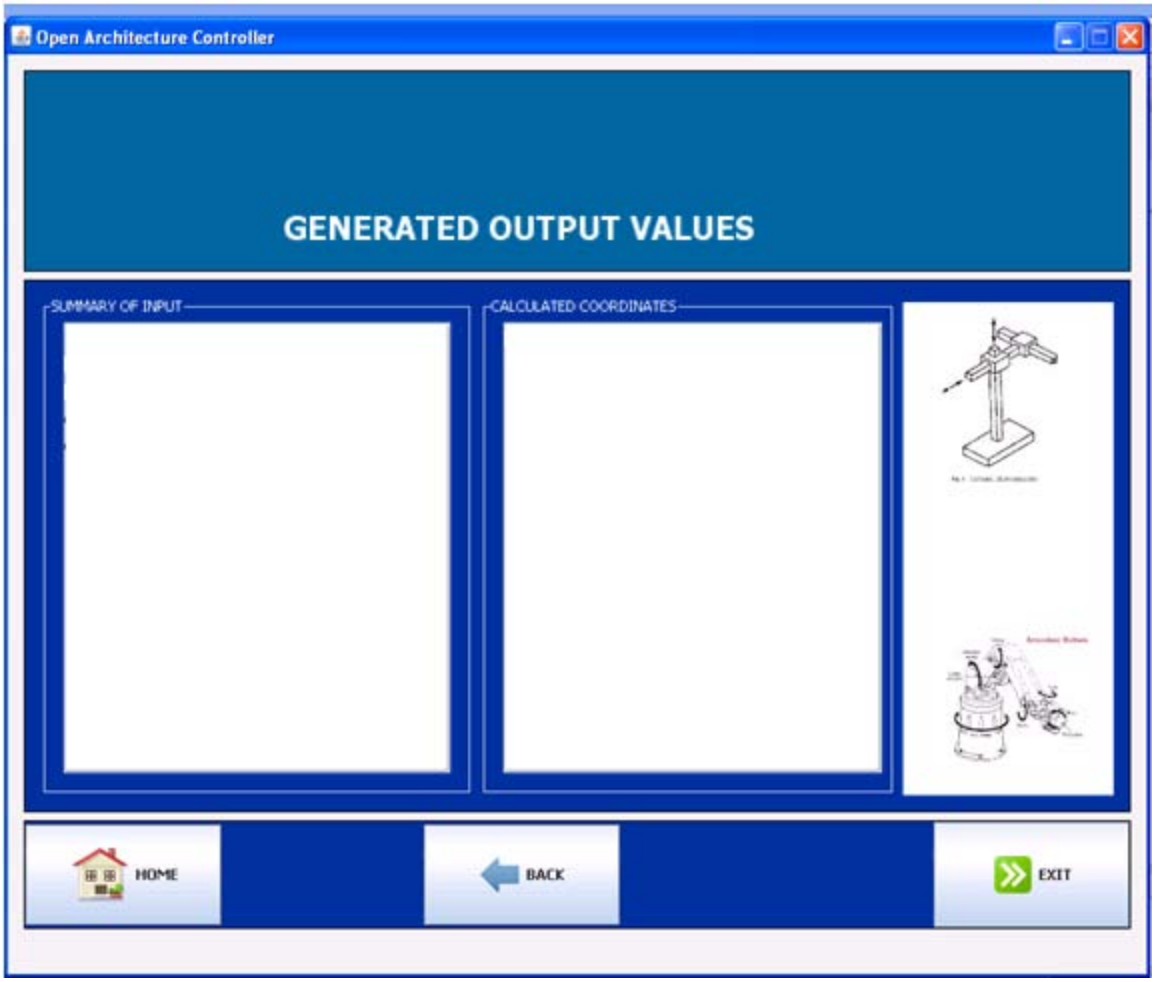

Figure 5.6 Input and Output Coordinates

5.5 RECONFIGURATION PROPERTIES

Modules are accessed through the use of an interface. For the controls on the interface (e.g. the command buttons in order to initiate an action), there is a programming code behind that is executed. This code is hidden from the operator and the operator does not need to know how the system initiates the processes that are carried throughout the system. Because they possess a well-defined interface, each individual control module can be changed independently of others (Tilbury and Kota, 1999:634). Given the coordinates for particular robot

architecture, there is a module that will be used for that specific case to calculate the solution for the end-effector position. The reason why a configuration design is also regarded as a part of system control is that for a reconfigurable system the system should be reconfigured frequently. This is done to meet quick changes and so that reconfigurable variables can be changed during controlling process (Bi et al, 2008:982). However, these variables need to be processed to determine the end-effector position.

Should the need to change or improve a module arise, only that part of the module will be affected – while the rest of the program remains unaffected. This has two great advantages: it enables changes to be tested for that particular module only; and it enhances the program as the rest of the program need not to be changed, and is also not affected by the changes made at the other parts of the program. The controller software consists of reusable software components corresponding to the physical machine configuration, and defines only the functionality of the machine control system (Wang and Shin, 2002:476).

5.6 ERROR HANDLING

Error handling is done at different levels throughout the system. The system allows the user to select the number of DOFs by clicking on the button for the number of DOFs to be used. This method does not permit the user to select more than one option at a time. The architectural style to be chosen is limited to the number of DOFs selected. Error handling is done mainly at design level. The system allows the input to be based on the option selected. For example, if one DOF is been selected, the system will permit one joint (either revolute or prismatic to be selected).

Unless the back button is clicked to select another option, the system displays the input option textboxes based on the number of DOFs selected. The other textboxes are disabled to minimize human error. The system displays the input as well as the output on the same screen, so as to make comparison between input and output results. The system permits the alteration of input if it was not captured correctly by clicking the back button and entering the correct parameters on the input screen.

5.7 VERIFICATION OF DEVELOPED MODEL

The model is reconfigurable as it allows one to specify the architectural style to use. The service that is provided at system level is the end-effector-motion control. The systems functionality is able to provide feedback as follows, and can be verified at system design:

- The system is able to provide a timeous feedback regardless of constraints posed by the environment
- The system is able to communicate errors effectively
- The system allows changes to be made in as simple, straightforward and effortless a way as possible
- The system is easy to use

From one DOF to five DOFs, it is possible to specify the number of DOFs and the architectural style at hand. It is also possible to find the end-effector position in an easy and effortless way. These measures can be evaluated on the design level.

5.8 CONCLUSION

In this chapter the three programming languages were considered: C++, C and Java. Java programming language was selected as a suitable programming language to carry out the project. The user friendly GUI is developed around forward kinematics to enable input for joint angles and link parameters. The GUI developed permits the selection of DOFs, architectural style, input of arm coordinate parameters and an output screen to map the new coordinates. The model has been verified to ensure whether it can carry out all the tasks specified.

6 ANALYSIS

6.1 CHAPTER OVERVIEW

In this chapter the Graphical User Interface will be analyzed in order to determine whether it is a suitable interface for the project. It will also be determined whether the interface is easy and user friendly to use. The components used to develop the GUI will also be examined.

Verification of the results will be done in order to establish whether the system developed is producing the desired and/or accurate results. The results will be verified by using simulation software. Puma robot manipulator will be used as the case study in order to verify the orientation and position of the end-effector.

6.2 GUI ANALYSIS

The user interface has been selected instead of a console screen. The main disadvantage of the console screen is that it is not visually appealing to the user. The console screen displays only text – this can be tedious for the user as they will have to always read the instructions. On the other hand, the user interfaces are visually appealing to the user and tend to be used for a long time (as they have a long time span). However, this requires careful planning and creativity in the design plan.

Below are two figures that demonstrate the use of option buttons, and the use of dropdown list boxes. The figures also demonstrate the comparison of these two options. Figures 4.3 and 4.4 are examples of a display for selecting four DOFs

reconfiguration architecture. Figure 4.3 uses the option button to select the architecture. The objective is to choose the type of joint, either revolute or prismatic. The option button needs a clear set of instructions alongside in order to clarify how to make the selection for the joints. It might be confusing to the user as to whether to choose the options horizontally or vertically. And if the GUI is improved by separating the options by lines (to show which options are pairs), it will not be as appealing to the user and will appear to be more traffic. It might be confusing to the user to use the system if there are more options buttons displayed on one screen. The more the number of DOFs increases, the more the screen will get cluttered. It also takes time to add radio buttons to display options that are available. There is a need to change the text of each option button and to program each button if they are not an array. In this case, eight option buttons are displayed for selecting a number of four DOFs. This demonstration shows that for each joint there are two options that should be added in the GUI to present a proper selection between the two. It takes considerable time to program each option button.

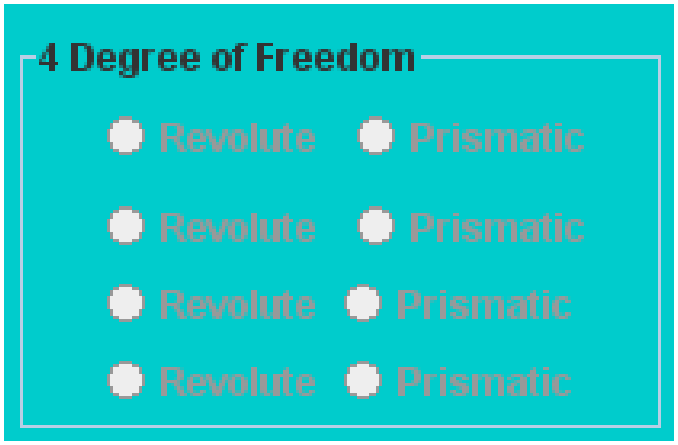

Figure 6.1 Option Button Selection Screen

Figure 6.2 shows the revised version of Figure 6.1 and shows the use of the dropdown listboxes. The screen is more portable and easy to understand, as

compared to the previous one. The screen shows the default option for each option that the user should choose from. Should the user not select another option, the default option is assumed. The default option in this case is a revolute joint for each joint. Should there be no change in any of the joints option, the system processes the information as all revolute joints to avoid the error whereby there is no revolute or prismatic joint chosen for a specific number of degrees. The arrow at the end of the listboxes confirms that there are other options available.

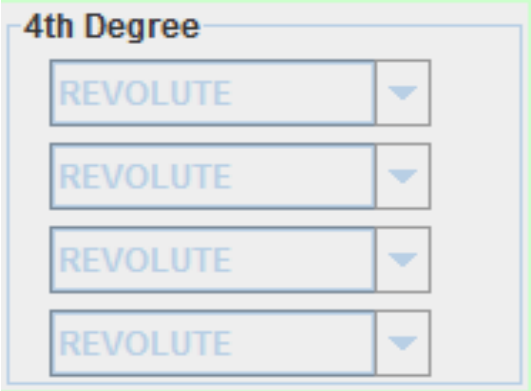

Figure 6.2 Dropdown List Selection Screen

6.3 VERIFYING THE RESULTS

6.3.1 CASE STUDY USING PUMA 560 ROBOT

To demonstrate the use of the controller designed, a PUMA Robot has been selected. PUMA 560 robot consist of six links and six revolute joints of an angle θ. Each joint consist of one DOF. The first three DOFs are located in the arm, which allows determining of the robot position (Benitez, Huitzil, Casiano, De la Calleje and Medina, 2012:18). The PUMA robot arm link coordinate parameters will be used to demonstrate the results of the controller developed. Figure 6.3a is

a display of a structure of the PUMA robot. Figure 6.3b shows the revolute joints of the robot by numbering them in order, starting from the base. The other two joints are located at the end-effector.

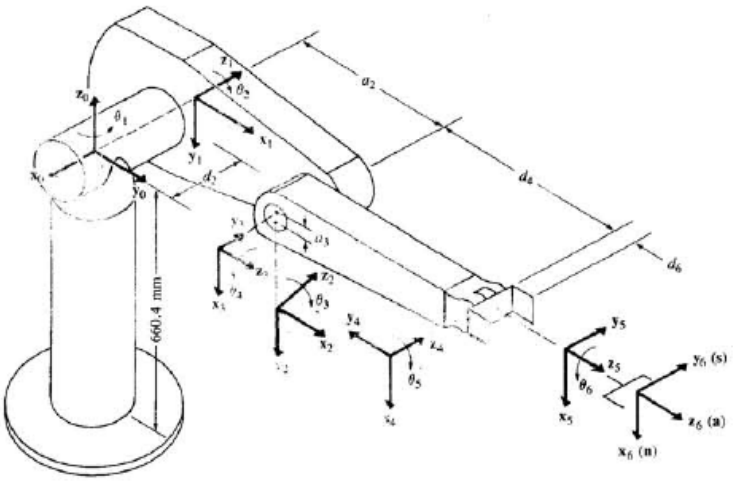

Figure 6.3 PUMA Robot (Fu et al, 1987, 37)

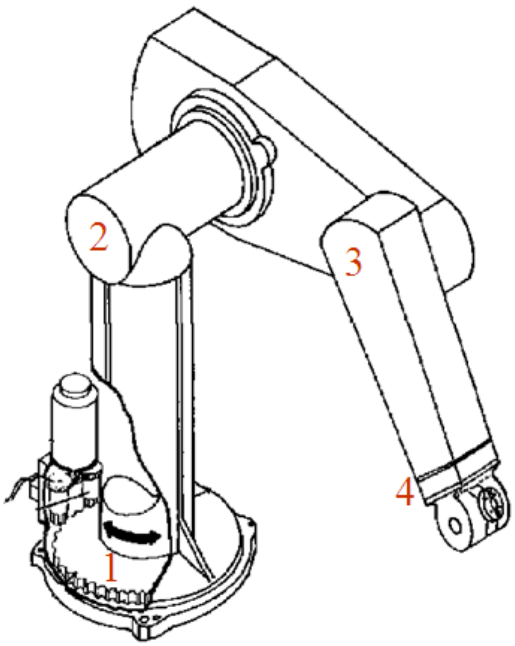

Figure 6.3b PUMA Robot joint numbers (Melamud R, Accessed: 20 June 2012)

The arm link coordinate parameters for the PUMA robot are displayed in Table 6.1 which establishes link coordinate systems for a PUMA robot. This figure shows the joint range. The joint range determines the motion limitations of each joint.

Table 6.1 Establishing link coordinate systems for a PUMA robot (Fu et al, 1987:37)

PUMA robot arm link coordinate parameters					
Joint i	θ_i	α_i	a_i	d_i	Joint Range
1	90	-90	0	0	-160 to +160
2	0	0	431.8 mm	149.09 mm	-225 to 45
3	90	90	-20.32 mm	0	-45 to 225
4	0	-90	0	433.07 mm	-110 to 170
5	0	90	0	0	-100 to 100
6	0	0	0	56.25 mm	-266 to 266

Arm link coordinate parameters in Table 6.2 will be used as a case study to demonstrate the controller developed and simulating the results by the use of MATLAB software. For the purpose of the study, the table below shows only the parameters for the joint angle and joint twist angle, which are the parameters that are used for revolute joints.

Table 6.2 Arm link coordinate parameters

Arm link coordinate parameters				
Joint	θ	α	a	d
1.	90	-90	0	0
2.	30	60	0	0
3.	90	90	0	0
4.	60	90	0	0
5.	180	0	0	0

6.3.1.1 CASE PROBLEM FOR 1 DOF

6.3.1.1.1 *ARM SOLUTION FOR 1 DOF USING THE PROGRAM DEVELOPED*

The first step is to choose the type of joint – which will determine the architectural style of the robot to be used. A revolute joint is selected for 1 DOF and is displayed in Figure 6.4. Figure 6.5 shows an illustration of a revolute joint motion.

Figure 6.4 Selecting Revolute Joint for 1 DOF

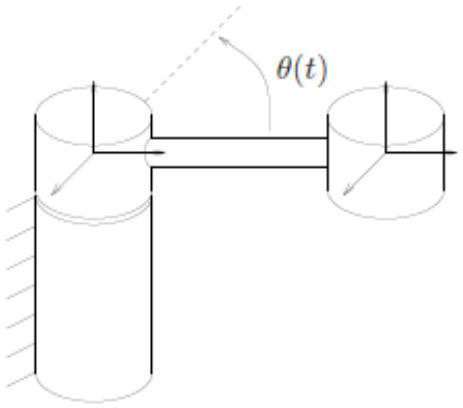

Figure 6.5 Rotational motion of a one degree-of-freedom manipulator (Murray et al, 1994:53)

6.3.1.1.2 *INPUT FOR 1 DOF ARM COORDINATES*

The input for 1 DOF is derived from the joint 1 coordinate parameters displayed in the table for case study. Figure 6.6 demonstrates how the input for the 1 DOF was captured in the system through the use of textboxes. Selecting 1 DOF, as illustrated in Figure 6.5, will allow the input for 1 DOF only. Other textboxes that are used for other DOF's are disabled and therefore no input is allowed in those textboxes. The other measure that can be used is that, even if there can be input, the system should only use the relevant input to calculate the rotating coordinate frame with respect to the reference frame. Figure 6.6 demonstrates how input was captured into the system.

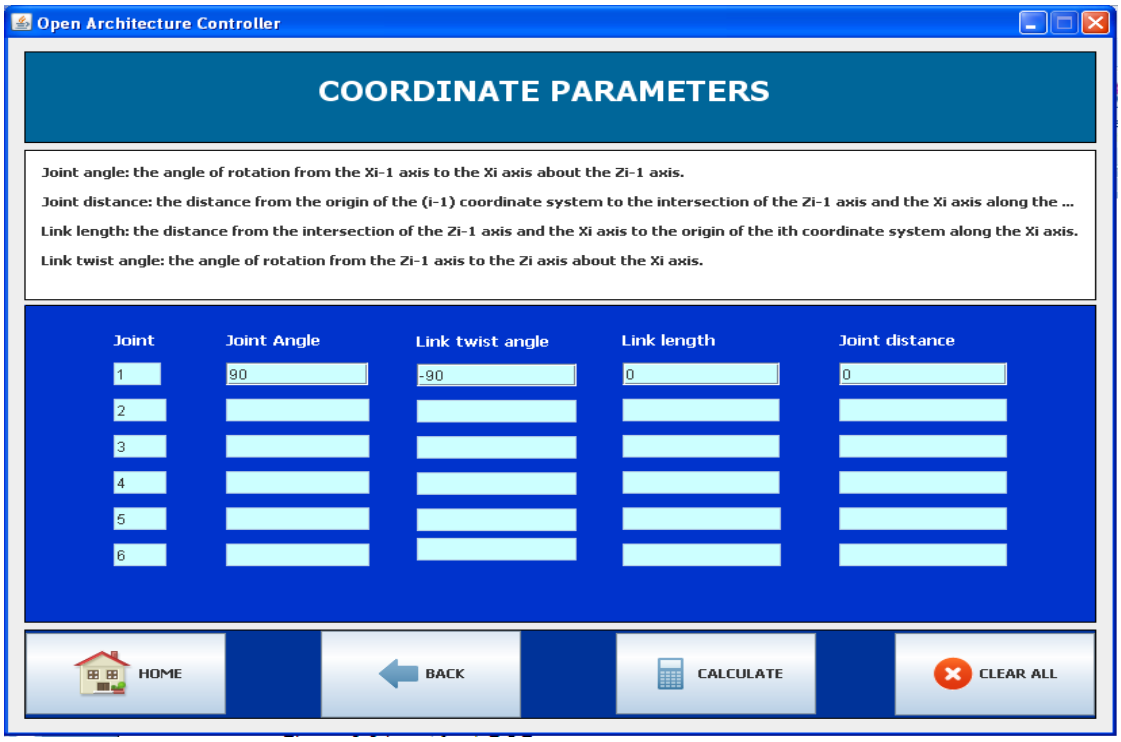

Figure 6.6 Input Parameters for 1 DOF

To process the input, the CALCULATE button needs to be clicked and the output is displayed in the next screen. The input and output is displayed in Figure 6.7.

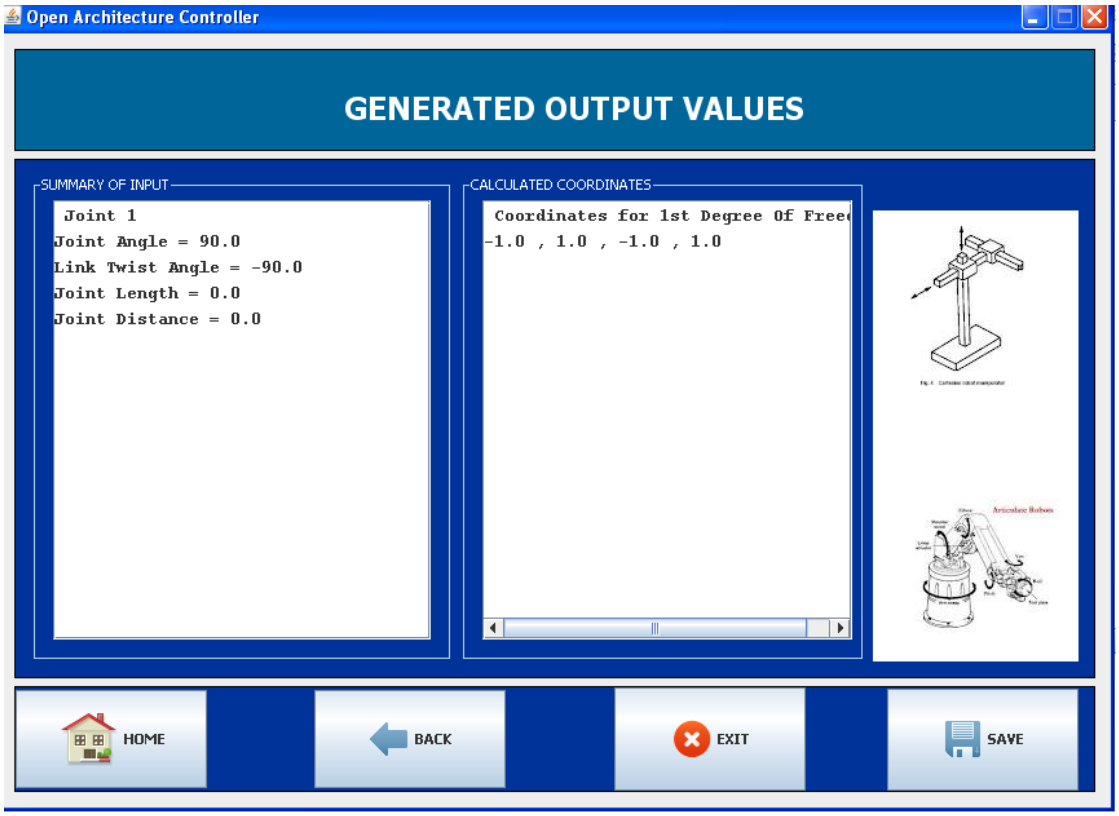

Figure 6.7 Input and Output Coordinates for 1 DOF

After all the input for 1 DOF was computed, the final output from the system for 1 DOF is as follows. The corresponding points, with respect to the reference coordinate system, are as follows:

1DOF = $\begin{pmatrix} -1.0 \\ 1.0 \\ -1.0 \\ 1.0 \end{pmatrix}$

6.3.1.1.3 *ARM SOLUTION FOR 1 DOF USING MATLAB*

To verify the results found when using the system developed, MATLAB program has been used and the results were as follows for 1 DOF after replacing all the variables with the required input parameters.

OneDOF =

$$\mathbf{T}1 = {}_{0}\mathbf{A}_{1} = \begin{pmatrix} -0.4481 & 0.4006 & -0.7992 & 0 \\ 0.8940 & 0.2008 & 0.4006 & 0 \\ 0 & -0.8940 & -0.4481 & 0 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix} = \begin{pmatrix} (-0.4481) + (0.4006) + (-0.7992) + (0) \\ (0.8940) + (0.2008) + (0.4006) + (0) \\ (0) + (-0.8940) + (-0.4481) + (0) \\ (0) + (0) + (0) + (1.0000) \end{pmatrix}$$

$$= \begin{pmatrix} -0.8467 \\ 1.4954 \\ -1.3421 \\ 1.0000 \end{pmatrix} = \begin{pmatrix} -1.0 \\ 1.0 \\ -1.0 \\ 1.0 \end{pmatrix} \qquad (27)$$

6.3.1.1.4 *CONCLUSION FOR 1 DOF*

Comparing the results for 1 DOF using the system developed and MATLAB Software resulted in the same output. This confirms that the system developed is producing the correct results for 1 DOF. However, a basic matrix is used for 1 DOF. It is expected that the results will be similar.

6.3.1.2 CASE PROBLEM 2

6.3.1.2.1 *ARM SOLUTION FOR 2 DOF USING THE PROGRAM DEVELOPED*

Figure 6.8 illustrates the selection of revolute joints for joint 1 and joint 2.

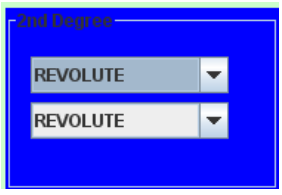

Figure 6.8 Selecting Revolute Joint for 2 DOF

Following the selection of the joint types, which determines the architectural style for 2 DOF, the input parameters coordinates were captured on the system as illustrated in Figure 6.9 from the Case Table.

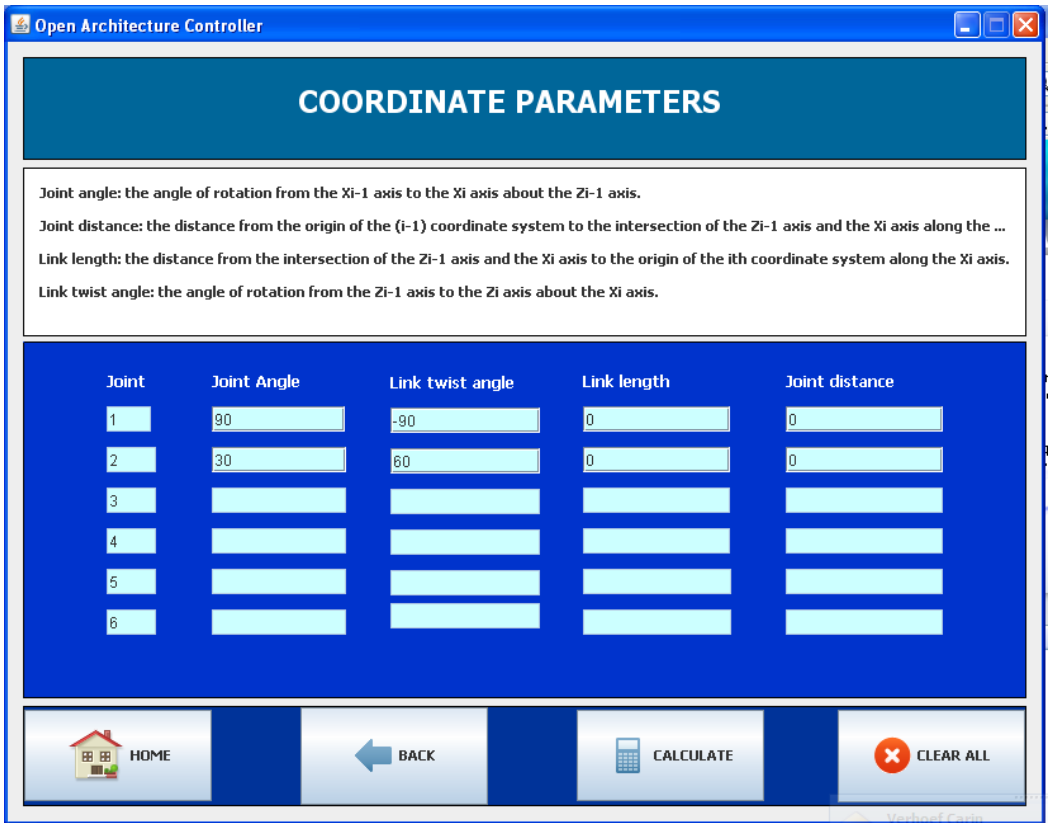

Figure 6.9 Input Parameters for 2 DOF

The results were as displayed in Figure 6.10 Input and Output Coordinates for 2 DOF after been processed.

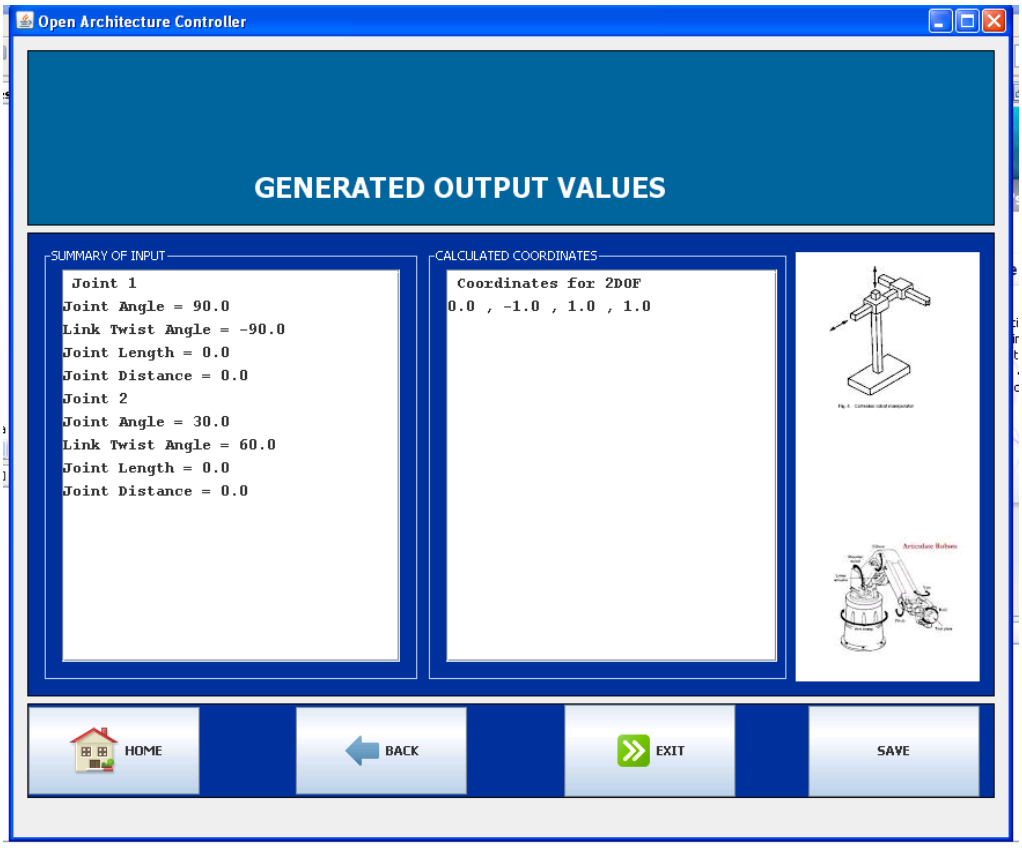

Figure 6.10 Input and Output Coordinates for 2 DOF

The final output from the system for 2 DOF is as follows, after all the input for 2 DOF was computed. The corresponding points, with respect to the reference coordinate system, are as follows:

2DOF = $\begin{pmatrix} 0.0 \\ -1.0 \\ 1.0 \\ 1.0 \end{pmatrix}$

6.3.1.2.2 *ARM SOLUTION FOR 2 DOF USING MATLAB*

The results were computed as follows after processing the input parameters from the study case table using the mathematical software to verify the results that the system developed produces.

TwoDOF =

$$
{}_0\mathbf{T}_1 = {}_0\mathbf{A}_2 = \begin{pmatrix} -0.4649 & 0.6064 & -0.1972 & 0 \\ -0.0605 & -0.9929 & 0.3616 & 0 \\ 0.8833 & 0.2679 & -0.2054 & 0 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix} = \begin{pmatrix} (-0.4649) + (0.6064) + (-0.1972) + (0) \\ (-0.0605) + (-0.9929) + (0.3616) + (0) \\ (-0.0605) + (-0.9929) + (0.3616) + (0) \\ (0) + (0) + (0) + (1.0000) \end{pmatrix}
$$

$$
= \begin{pmatrix} 0.0557 \\ -0.6918 \\ 0.9458 \\ 1.0000 \end{pmatrix} = \begin{pmatrix} 0.0 \\ -1.0 \\ 1.0 \\ 1.0 \end{pmatrix} \qquad (28)
$$

6.3.1.2.3 *CONCLUSION FOR CASE 2*

When the MATLAB results were rounded off to the nearest ten, they matched the results that the system developed produces. The results produced by the system were also rounded off to the nearest ten. This confirms that the system developed is producing the correct results for 2 DOF, as one difference from the results will mean that there is a problem or inconsistency with the formula when it was coded to calculate the results. Accuracy and consistency are very significant issues when working with formulas.

6.3.1.3 CASE PROBLEM 3

6.3.1.3.1 *ARM SOLUTION FOR 3 DOF USING THE PROGRAM DEVELOPED*

The 3 revolute joints were selected as displayed in Figure 6.11 follows for joint 1, joint 2 and joint 3 to determine the architectural style for 3 DOF.

Figure 6.11 Selecting Revolute Joint for 3 DOF

The next step is to capture the input parameters for the architectural style that has been selected. Figure 6.12 illustrates how the input was captured using the case table data for joint 1, joint 2 and joint 3.

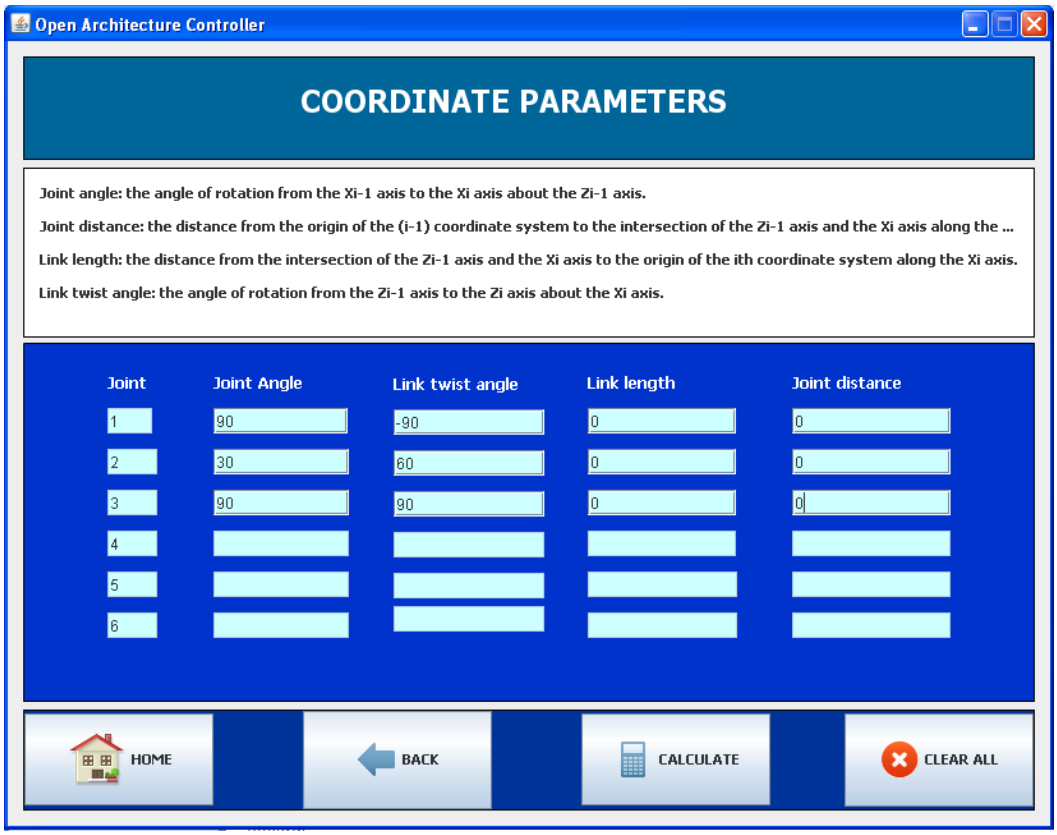

Figure 6.12 Input Parameters for 3 DOF

The results were as displayed in Figure 6.13 Input and Output Coordinates for 3 DOF.

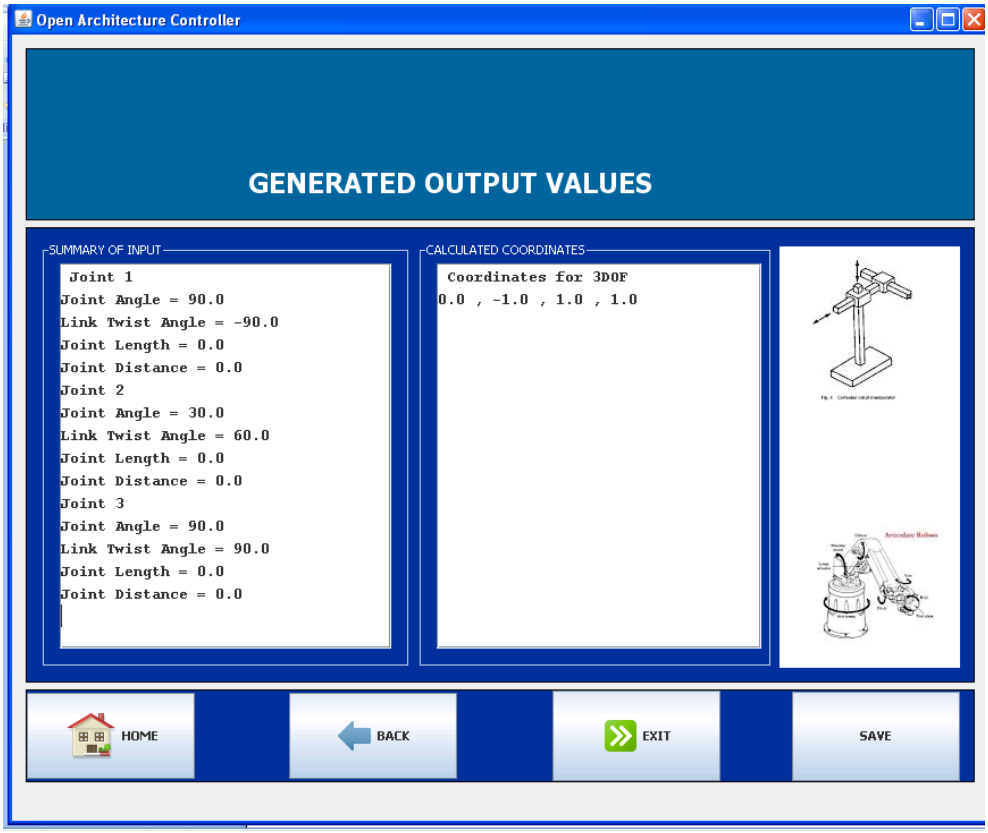

Figure 6.13 Input and Output Coordinates for 3 DOF

3DOF = $\begin{pmatrix} 0.0 \\ -1.0 \\ 1.0 \\ 1.0 \end{pmatrix}$

6.3.1.3.2 *ARM SOLUTION FOR 3 DOF USING MATLAB*

The results were computed as follows after processing the input parameters from the study case table for 3 DOF using mathematical software.

ThreeDOF =

$$ {}_0\mathbf{T}_3 = {}_0\mathbf{A}_3 = \begin{pmatrix} 0.7504 & -0.2408 & -0.0403 & 0 \\ -0.8605 & 0.0997 & -0.6081 & 0 \\ -0.1563 & 0.2240 & 0.9053 & 0 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix} = \begin{pmatrix} (0.7504) + (-0.2408) + (-0.0403) + (0) \\ (-0.8605) + (0.0997) + (-0.6081) + (0) \\ (-0.1563) + (0.2240) + (0.9053) + (0) \\ (0) + (0) + (0) + (1.0000) \end{pmatrix} $$

$$ = \begin{pmatrix} 0.4693 \\ -1.3689 \\ 0.9730 \\ 1.0000 \end{pmatrix} = \begin{pmatrix} 0.0 \\ -1.0 \\ 1.0 \\ 1.0 \end{pmatrix} \quad (29) $$

6.3.1.3.3 *CONCLUSION FOR 3 DOF*

The results for 3 DOF that were produced by the system were matched to the ones that MATLAB software produces. The results from the two systems correspond to each other. This confirms that the system developed is producing the correct results for 3 DOF and also indicates that the formulas in the system developed were captured correctly.

6.3.1.4 CASE PROBLEM 4

6.3.1.4.1 *ARM SOLUTION FOR 4 DOF USING THE PROGRAM DEVELOPED*

The joint types for 4 DOF were selected as illustrated in Figure 6.14 to determine the architectural style for 4 DOF.

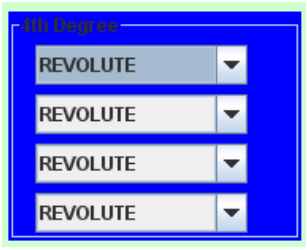

Figure 6.14 Selecting Revolute Joint for 4 DOF

The more the parameters, the more complex the program becomes, as it is supposed to compute an increased number of variables. The input for 4 DOF were as illustrated in Figure 6.15.

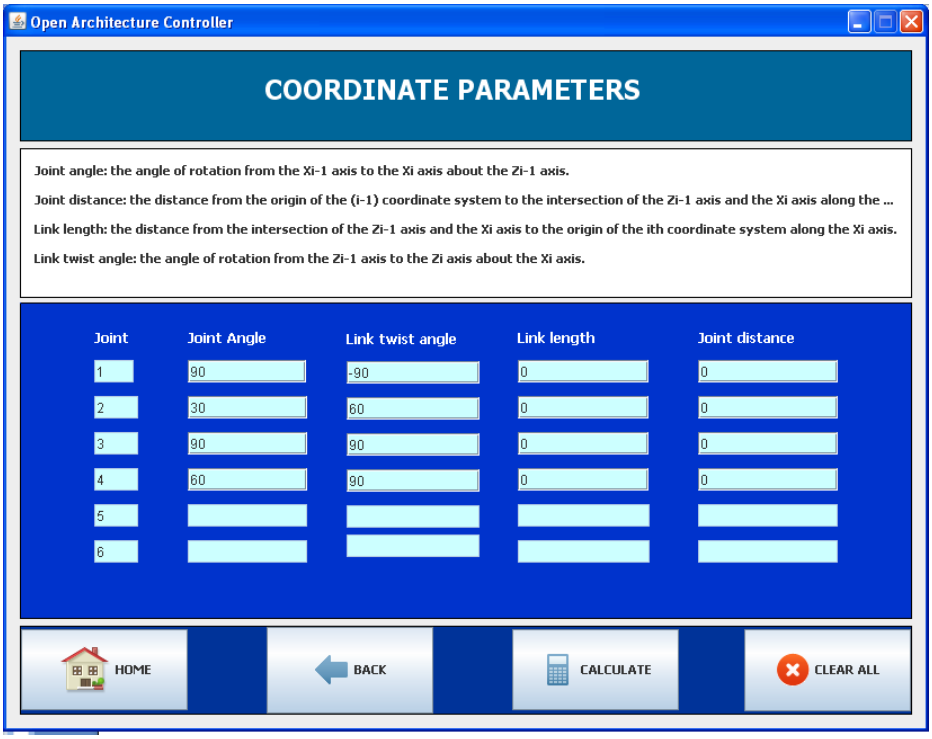

Figure 6.15 Input Parameters for 4 DOF

The output for 4 DOF is illustrated in Figure 6.16. The output is interpreted as follows:

4DOF = $\begin{pmatrix} -1.0 \\ 1.0 \\ 0.0 \\ 1.0 \end{pmatrix}$

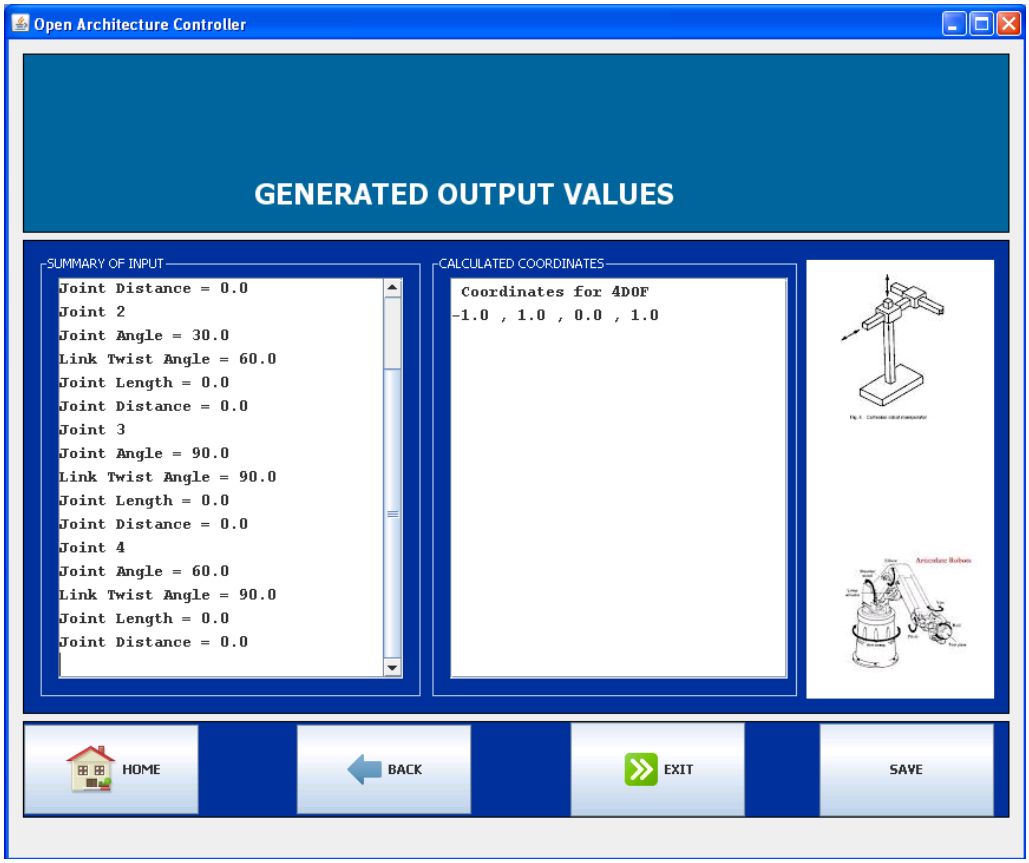

Figure 6.16 Input and Output Coordinates for 4 DOF

6.3.1.4.2 *ARM SOLUTION FOR 4 DOF USING MATLAB*

To verify the results by simulation the results were as displayed below using mathematical software:

FourDOF =

$$ {}_0\mathbf{T}_4 = {}_0\mathbf{A}_4 = \begin{pmatrix} -0.6413 & -0.2413 & -0.0962 & 0 \\ 0.7892 & -0.3835 & 0.7847 & 0 \\ 0.0805 & 0.9263 & -0.8847 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} (-0.6413) + (-0.2413) + (-0.0962) + (0) \\ (0.7892) + (-0.3835) + (0.7847) + (0) \\ (0.0805) + (0.9263) + (-0.8847) + (0) \\ (0) + (0) + (0) + (1) \end{pmatrix} $$

$$ = \begin{pmatrix} -0.9788 \\ 1.1904 \\ 0.1221 \\ 1.0000 \end{pmatrix} = \begin{pmatrix} -1.0 \\ 1.0 \\ 0.0 \\ 1.0 \end{pmatrix} \quad (30) $$

6.3.1.4.3 *CONCLUSION FOR 4 DOF*

The MATLAB results matched the results that the system developed produces. This confirms that the system developed is producing the correct results for 4 DOF. The formulas were captured accurately for 4 DOF in the system developed.

6.3.1.5 CASE PROBLEM 5

6.3.1.5.1 *ARM SOLUTION FOR 5 DOF USING THE PROGRAM DEVELOPED*

The system was tested as the number of joints increased. The type of joints that were selected for 5 DOF was all revolute joints. The joints were selected as illustrated in Figure 6.17 to determine the architectural style for 5 DOF.

Figure 6.17 Selecting Revolute Joint for 5 DOF

The input for 5 DOF is illustrated in Figure 6.18 to show how it was captured using the data from the case table.

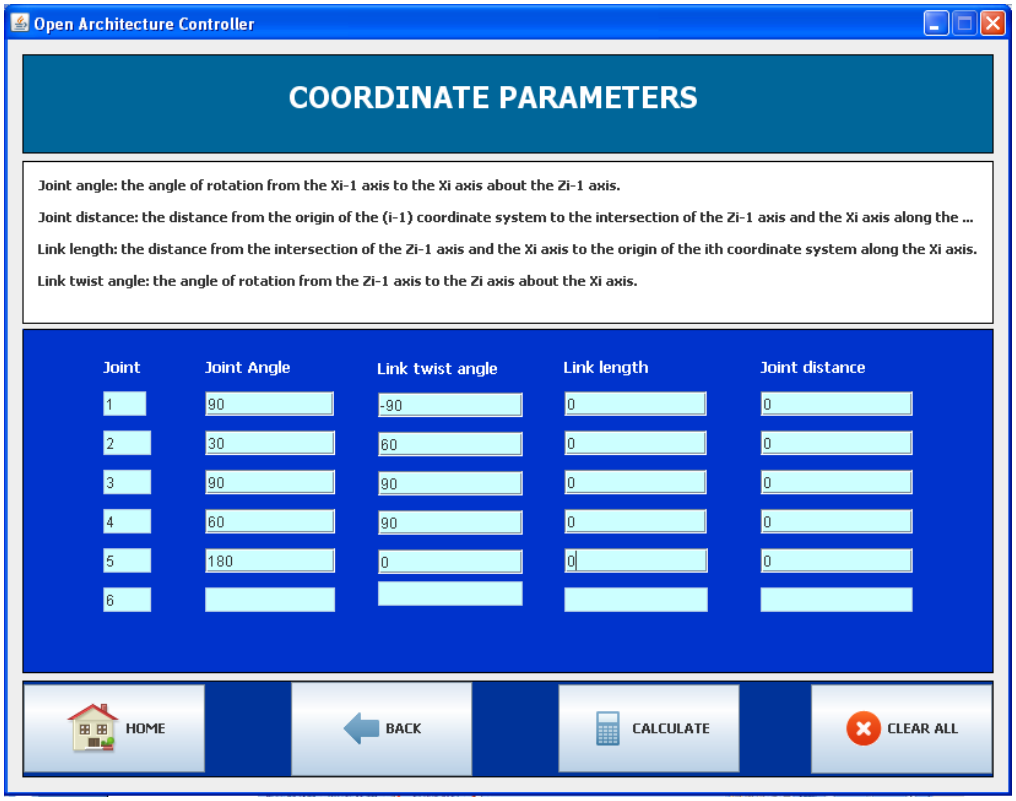

Figure 6.18 Input Parameters for 5 DOF

The input for 5 DOF was verified as to whether it has being captured correctly – that is, whether it matches the one provided in the case table. The input was processed and yielded the results as illustrated in Figure 6.19 with respect to the reference frame.

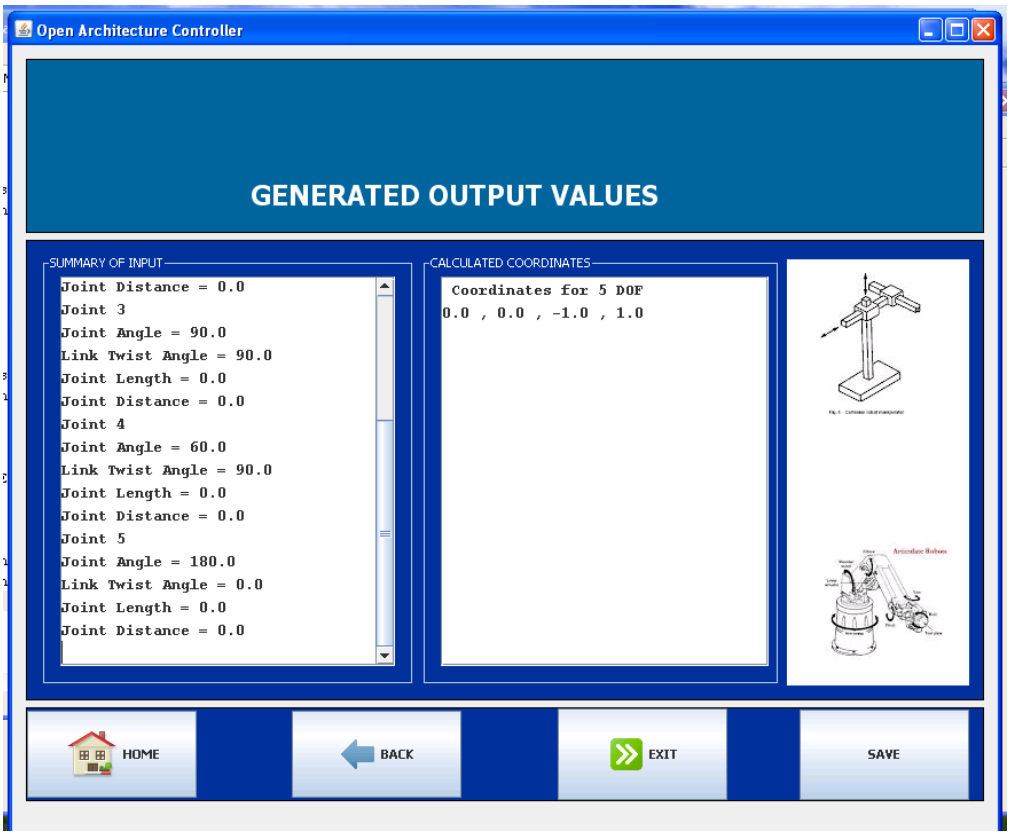

Figure 6.19 Input and Output Coordinates for 5 DOF

The results are interpreted as follows:

5DOF = $\begin{pmatrix} 0.0 \\ 0.0 \\ -1.0 \\ 1.0 \end{pmatrix}$

6.3.1.5.2 *ARM SOLUTION FOR 5 DOF USING MATLAB*

To verify if the results from the system were accurate, the same input from the case table was used in mathematical software to simulate the results. With 5

DOF there are many variables that needed to be used in order to verify the results, as each degree has at least 4 variables for holding the data provided by the user. The results were as follows and correspond to the ones that the system produced:

FiveDOF =

$$
{}_0\mathbf{T}_5 = {}_0\mathbf{A}_5 = \begin{pmatrix} 0.5771 & -0.3694 & 0.1733 & 0 \\ -0.1650 & 0.8618 & -0.2857 & 0 \\ -0.7903 & -0.4898 & 0.0853 & 0 \\ 0 & 0 & 0 & 1.0000 \end{pmatrix} = \begin{pmatrix} (0.5771) + (-0.3694) + (0.1733) + (0) \\ (-0.1650) + (0.8618) + (-0.2857) + (0) \\ (-0.7903) + (-0.4898) + (0.0853) + (0) \\ (0) + (0) + (0) + (1.0000) \end{pmatrix}
$$

$$
= \begin{pmatrix} 0.3810 \\ 0.4111 \\ -1.1948 \\ 1.0000 \end{pmatrix} = \begin{pmatrix} 0.0 \\ 0.0 \\ -1.0 \\ 1.0 \end{pmatrix} \qquad (31)
$$

6.3.1.5.3 *CONCLUSION FOR CASE 5*

The results produced by the system were matched to the ones that were found when using MATLAB. Regardless of the complexity of the formulas, the results corresponded to each from both the software that was used to calculate the end-effector. Therefore, this confirms that the system developed is producing the correct results for 5 DOF.

6.4 CONCLUSION

In this Chapter, the GUI has been analyzed to determine whether it is suitable for the system developed. The GUI developed allows all the necessary input to be captured in the system and permits the output to be displayed in a way that it can be interpreted accordingly.

Puma Robot 560 is used as the case study to determine and test if the system developed is producing the correct results. The system was tested to verify the results for 1 DOF to 5 DOF. The system permits the selection of number of DOFs to be selected. It further allows the selection of joints between prismatic and revolute joints, based on the number of DOFs, in order to determine the architectural style. Input coordinates are captured for processing using the arm link coordinate parameters. The input is processed for each DOF and verified by the use of the mathematical software in each aspect. The system is able to map the coordinates in a specified rotated coordinate system from one position to another.

The models were verified to ensure the correctness of the results. For each joint, the model was verified to see if it will be possible to input all the coordinates of the arm regardless of the number of degrees of freedom specified. Using the mathematical

model that has been developed, it has been noted that it is possible to find the end-effector position using all the given coordinates.

Therefore the system developed has proved to be able to simulate the results in real time.

7 CONCLUSION

7.1 CHAPTER OVERVIEW

This chapter reflects back on the objectives of the study to determine if they have been achieved. Conclusion of the study and recommendations for further study are also specified.

7.2 OVERVIEW OF OBJECTIVES

RMS machine tool components were identified by the use of OAC's from the industry, academic projects and other OAC for manufacturing. EMC was also identified as a real-time controller for robots and machine tools. These tool components possess the ability to be converted quickly for the production of new models and assist to adjust to the exact capacity requirements based on the demand by using control modules. The RMS machine tools have changeable structure in nature as they allow adjustments to be made quickly.

Different machine links and machine joints were modeled in which the user is able to select the desired architectural from one DOF to five DOF's. The joints that can be selected are prismatic and/or revolute joints for the selected number of DOF's. The number of degrees of freedom determines the number of joints to be selected. The underlying kinematics model have been developed and implemented in software for the overall machine. The system can further be modified to include other joints and may be improved to accommodate further DOFs.

The controller was further implemented in simulation and the results produced from the system were verified by comparing them to those received from MATLAB software.

Open architecture were explored by the use of Java software. The system developed is able to produce accurate results as soon as they are needed. The system uses logical approach to calculate the results and further displays them as soon as the button for calculating the results is clicked.

Portable and reusable algorithms were written to model different links and joints which are isolated from hardware configurations. The links and joints that can be modeled using the system are not for specific hardware. The usable machine interface has been designed to allow human interaction. The researcher has contributed to the research by optimizing the calculations implemented in software controller.

7.3 CONCLUSION AND RECOMMENDATIONS

A real-time, open controller for reconfigurable manufacturing systems has been developed for up to 5 DOF. The developed controller has proven to be able to carry out the simulation process of mapping the coordinates in a specified coordinate system, from one position to another, in respect of the reference frame. The system uses an easy to use GUI, as each phase emphasizes clearly what is needed for human interaction.

The objectives of the project to model the different machine links and different machine joints, as well as to model kinematics of the overall machine tool, were achieved with the software controller that has been developed.

The research that was conducted for this study has produced controller software that can be reconfigured for different manufacturing processes. The number of DOF can be changed quickly based on the production capacity needed.

The aim of developing ROACS that does not depend on any specific hardware or software configuration was accomplished.

Recommendation for future work that needs to be completed is to develop a physical prototype for a reconfigurable manufacturing system where the controller can be implemented to establish its openness to the platform to allow the physical test. The controller can further be enhanced by catering for up to 6 DOF robots and can comprise of other joints such as a revolving joint, twisting joint, etc. The process of an open-architecture is seen as an ongoing process.

REFERENCES:

Atta-Konadu, R., “Modular Controller For Robotic Machines”, Thesis (PHD), University of Saskatchewan, Saskatoon, (2006, 187).

Barabanov M., “A Linux-based Real-Time Operating System“, *Thesis (MSc in Computer Science), New Mexico Institute of Mining and Technology Socorro, New Mexico*, June 1997.

Benitez A., Huitzil I., Casiano A., De La Calleja J. and Medina M. A., “PUMA 560: Robot Prototype with Graphic Simulation Environment”, 2012, *Advances in Mechanical Engineering, ISSN: 2160-0619,* vol.2, no. 1, pp. 17 - 24, March 2012.

Bi Z. M., Lang S.Y.T., Verner M., Orban P., “Development of Reconfigurable machines”, *International Journal of Advanced Manufacturing Technology,* vol. 39, pp. 1227 – 1251, 2008.

Bi Z.M., Langa S. Y. T., Shena W. and Wanga L., “Reconfigurable manufacturing systems: the state of art”, *International Journal of Production Research*, vol. 46, no. 4, pp. 967-992, February 2008.

C Riba R., Pérez R.R., Ahuett H.G., Sànchez J. L. A., Domínguez M. D. and Molina A. G., “Metrics for Evaluating Design of Reconfigurable Machine Tools”. *Y:Luo (Ed):CDVE , LNCS 4101,* pp. 234 – 241, 2006.

De-Jiu Chen, “*Architecture for Systematic Development of Mechatronics Software Systems*”, 2001*,* Engineering of Complex Computer Systems, Proceedings, Seventh IEEE International Conference, 2001, pp.170 – 179.

ElMaraghy H. A., Flexible and reconfigurable manufacturing systems paradigm. *Intelligent Manufacturing Systems*, vol. 17, no 4, pp. 261 – 276, 2006.

Fu K.S., Gonzalez R.C. and Lee C.S.G., "*Robotics – Control, Sensing, Vision and Intelligence*", United States:
McGraw-HilI Book Company, 1987.

Gambier A., "Real-time Control Systems: A Tutorial", *Control Conference*, vol. *5, pp. 1024 - 1031,* 2004.

Golub G. H. and Van Loan C. F., "*Matrix Computations*", Baltimore and London: The Johns Hopkins University Press, 1996.

Hanzálek Z., "Petri Net Models for Manufacturing Systems", 1996.

Hartenberg, R. S. and Denavit J., "*Kinematic Synthesis of Linkages*", New York: McGraw-Hill Book Company, 1964.

Katz R., "Design principles of reconfigurable machines", 2007, The International Journal of Advanced Manufacturing Technology, vol. 34, pp. 430 -439.

Katz R., Moon Y M, "Virtual Arch type Reconfigurable machine too design: *Principles and Methodology*", *The University of Michigan, NSFERC for RMS*, pp. 1 - 23, 2000.

Katz R., Min B., Pasek Z., "Open Architecture Control Technology Trends",

2000, *ERC/RMS Report 35,* pp. 1 - 26.

Kim J. S., Garlan D., "*Analyzing Architectural Styles*", Carnegie Mellon University Research Showcase, pp. 1 – 55, 2007.

Kolla S., Michaloski J. and Rippey W., "Evaluation of Component –Based Reconfigurable machine Controllers" *Automatic Congress 2002,* vol. 14, pp. 625 – 630.

Konadu A., "Design and Implementation of a modular controller for robotic machines" 2006, *Thesis (PHD). Saskatoon, University of Saskatchewan.*

Koren Y., Heisel U, Jovane F, Moriwaki T, Pritschow G, Ulsoy G and Van Brussel H, "Reconfigurable Manufacturing Systems". *Annals of the CIRP*, vol. 48, pp. 527- 540, 1999.

Koren Y., "*The Global Manufacturing Revolution: Product-Process-Business Integration and Reconfigurable Systems*",A John-Wiley & Sons, Inc., Publication, Hoboken, 2010.

Kumar Y., Srivastava S. K. and Bajpai A. K., "*Intelligent Modular Controller for Robotic Machines*", 13th National Conference on Mechanisms and Machines , pp. 297 – 304, 2007.

Linux [Online], http://www.linuxcnc.org/, [Accessed 10 July 2010].

Mathworks, SimMechanics™ User's Guide, 2010, Available:

http://www.Mathworks.com, [Accessed: 8 August 2010].

Mehrabi M. G., Ulsoy A. G., Koren Y., Heytler P., “Trends and perspectives in flexible and reconfigurable manufacturing systems”. *Journal of Intelligent Manufacturing*,
Vol 13, pp 135 – 146, 2002.

Mehrabi M. G., Ulsoy A. G., Koren Y., “Reconfigurable Manufacturing Systems and their enabling technologies” *International Journal of Manufacturing Technology and Management,* vol. 1, no. 1, pp.114 - 131, 2000.

Mehrabi M. G., Ulsoy A. G., Koren Y., “Reconfigurable Manufacturing Systems: Key to Future Manufacturing”. *Journal of Intelligent Manufacturing,* pp. 403 – 419, 2000.

Mekid S., Pruschek P. and Hernandez J., “Beyond intelligent manufacturing: A new generation of flexible intelligent NC machines”, *Mechanism and Machine Theory,* vol. 44, 466 – 476, 2009.

Melamud R., “*An Introduction to Robot Kinematics”*,
http://www.site.uottawa.ca/.../generalrobotics.org, [Accessed: 20 June 2012].

Merlet J.P., “*Parallel Robots*”, The Netherlands: Springer, 2006.

Molina A., Rodriguez C. A., Ahuett H., Cortés J. A., Ramírez M., Jiménez G. and Martinez S., “Next generation manufacturing systems: key to research

issues in developing and integrating reconfigurable and intelligent machines." *International Journal of Computer Integrated Manufacturing,* 2005, vol. 18, no. 7, pp.525 – 536.

Moriwaki T., "Multi-functional Machine Tool", "*CIRP Annals-Manufacturing Technology"*, vol. 57, no. 2, pp. 736 – 749*, 2008.*

Moyne J., Korsakas J. and Tilbury D. M., "Reconfigurable Factory Testbed (RFT): A distributed testbed for reconfigurable manufacturing systems", *In proceedings of the Japan-USA symposium on flexible Automation,* Denver, Colorado, July 2004.

Moyne J., Tilbury D. M., Sukerkar K, Wijaya H., "An Integrated Distributed Software System for Reconfigurable Manufacturing." *Advanced Manufacturing Technologies*", 2004.

Murray R., Li X. and Sastry S. S., "*A Mathematical Introduction to Robotic Manipulation*", Boca Raton: CRC PressINC, 1994.

Nasca J., "Comparison of three different open architecture controllers" *Computer & Automation Research Institute, Inproceedings Nacsa01comparisonof,* 2001, pp. 2 – 4.

Oldknow K.D. and Yellowley I., "Design, implementation and validation of a system for the dynamic reconfiguration of open architecture machine tool controls". *International Journal of Machine Tools and Manufacture,* vol. 41, pp. 795 – 808, 2001.

Orin D. E., Schrader W. W., “Efficient Computation of the Jacobian for Robot Manipulators”, *The International Journal of Robotics Research, vol. 3, no 4*, pp. 66 – 75, 1984.

Orin D. E., Schrader W W, “Efficient Computation of the Jacobian for Robot Manipulators”*, The International Journal of Robotics Research,* vol. 3, no. 4, pp. 66 - 75, Dec 1984.

Padayachee J., Bright G. and Masekamela I., “Modular Reconfigurable Machine Tools: Design, Control and Evaluation”, *South African Journal of Industrial Engineering*, vol. 20, no. 2, pp. 127 – 143, Nov 2009.

Pritschow G., Altintas Y., Jovane F., Koren Y., Mitsuishi M, Takata S, van Brussel H, Weck M and Yamazaki K, “Open Controller Architecture-Past, Present and Future” *Annals of the CIRP,* Vol 50, no.2, pp. 446 - 463, 2001.

Seames W. S., “Computer numerical control: concepts and programming“, United States: Thomson Learning, Delmar, 2001.

Sherer M., “*RTLinux Application Development Tutorial*“ [Online] , 2002, Available from: http://www.linuxjournal.com/article/5694, [Accessed: May 22, 2011].

Smith-Atakan S., Human-Computer Interaction, New Jersey, Pearson Education, Inc., 2006

Sperling W. and Lutz P., “Designing applications for an OSACA Control”, *Proceedings of the International Mechanical Engineering Congress and Exposition, Dalles/ USA*, pp. 16 – 21, Nov 16 – 21, 1997.

Spicer P., Koren Y., Shpitalni M. and Yip-Hoi D, “Design Principles for Machining System Configurations” *Annals of the CIRP,* vol. 51, no. 1, pp. 275 – 280, 2002.

Tilbury D. M. and Kota S., “Integrated Machine and Control Design for Reconfigurable Machine Tools”, *Proceedings of the 1999 IEEE/ASME, International Conference on Advanced Intelligent Mechatronics*, September 19-23, 1999, pp. 629 - 634.

Tsai L., “*Robot Analysis: The Mechanics of Serial and Parallel Manipulators”*, Canada: John-Willey & Sons, 1999

Ulmer B. C., Kurfess T. R., “Integration of an open architecture controller with a diamond turning machine” *Mechatronics,* 1999, vol. 9, no 4, pp. 349 – 361.

Ulsoy G., Pasek Z. J., Shan Y., Koren Y., Park J, Birla S and Shin KG, “An Open architecture Testbed for Real-Time monitoring and control of machining processes” *American Control Conference,* 1995, vol. 1, 200 – 204.

Van Brussel,H., Sas P., Nemeth I., De Fonseca P., den Braembussche P., “Towards a Mechatronic Compiler”, IEEE/ASME Transactions on Mechatronics, vol. 6, No. 1, pp. 90 – 105, 2001.

Wang S. and Shin K. G., "Constructing Reconfigurable Software for Machine Control Systems“, *Robotics and Automation, IEEE Transactions,* August 2002, pp. 475 – 486.

Wang Y. and Chirikjian G. S., “Large kinematic error propagation in revolute manipulators”, *Advances in Robot Kinematics,* 2006, pp. 95 – 102.

Wills L., Sander S., Kannan S., Kahn A., Prasad J. V. R. and Schrage D., "An Open Control Platform for Reconfigurable, Distributed, Hierarchical Control Systems, “*Proceedings of the Digital Avionics Systems Conference,* Philadelphia, PA, October 2000.

APPENDIX A

TABLE FOR ARCHITECTURAL STYLES

Degrees of Freedom	Architectural Style No	Architectural Style	Simplified Architecture
1 Degree-of-freedom	1.1	Revolute	R
	1.2	Prismatic	P
2 Degrees of freedom	2.1	Revolute, Revolute	RR
	2.2	Revolute, Prismatic	RP
	2.3	Prismatic, Revolute	PR
	2.4	Prismatic, Prismatic	PP
3 Degrees of freedom	3.1.	Revolute, Revolute, Revolute	RRR
	3.2.	Revolute, Revolute, Prismatic	RRP
	3.3.	Revolute, Prismatic, Revolute	RPR
	3.4.	Revolute, Prismatic, Prismatic	RPP
	3.5.	Prismatic, Prismatic, Revolute	PPR
	3.6.	Prismatic, Prismatic, Prismatic	PPP
	3.7.	Prismatic, Revolute, Revolute	PRR
	3.8.	Prismatic, Revolute, Prismatic	PRP

Degrees of Freedom	Architectural Style No	Architectural Style	Simplified Architecture
4 Degrees of freedom	4.1.	Revolute, Revolute, Revolute, Revolute	RRRR
	4.2.	Revolute, Revolute, Revolute, Prismatic	RRRP
	4.3.	Revolute, Revolute, Prismatic, Revolute	RRPR
	4.4.	Revolute, Prismatic, Revolute, Revolute	RPRR
	4.5.	Prismatic, Revolute, Revolute, Revolute	PRRR
	4.6.	Revolute, Revolute, Prismatic, Prismatic	RRPP
	4.7.	Revolute, Prismatic, Prismatic, Revolute	RPPR
	4.8.	Prismatic, Prismatic, Revolute, Revolute	PPRR
	4.9.	Prismatic, Prismatic, Prismatic, Prismatic	PPPP
	4.10.	Prismatic, Prismatic, Prismatic, Revolute	PPPR
	4.11.	Prismatic, Prismatic, Revolute, Prismatic	PPRP
	4.12.	Prismatic, Revolute, Prismatic, Prismatic	PRPP
	4.13.	Revolute, Prismatic, Prismatic, Prismatic	RPPP
	4.14.	Prismatic, Revolute,	PRRP

Degrees of Freedom	Architectural Style No	Architectural Style	Simplified Architecture
		Revolute, Prismatic	
	4.15.	Prismatic, Revolute, Prismatic, Revolute	PRPR
	4.16.	Revolute, Prismatic, Revolute, Prismatic	RPRP
5 Degrees of freedom	5.1.	Revolute, Revolute, Revolute, Revolute, Revolute	RRRRR
	5.2.	Revolute, Revolute, Revolute, Revolute, Prismatic	RRRRP
	5.3.	Revolute, Revolute, Revolute, Prismatic, Revolute	RRRPR
	5.4.	Revolute, Revolute, Prismatic, Revolute, Revolute	RRPRR
	5.5.	Revolute, Prismatic, Revolute, Revolute, Revolute	RPRRR
	5.6.	Prismatic, Revolute, Revolute, Revolute, Revolute	PRRRR
	5.7.	Revolute, Revolute, Revolute, Prismatic, Prismatic	RRRPP
	5.8.	Revolute, Revolute, Prismatic, Prismatic,	RRPPR

Degrees of Freedom	Architectural Style No	Architectural Style	Simplified Architecture
		Revolute	
	5.9.	Revolute, Prismatic, Prismatic, Revolute, Revolute	RPPRR
	5.10.	Prismatic, Prismatic, Revolute, Revolute, Revolute	PPRRR
	5.11.	Revolute, Revolute, Prismatic, Revolute, Prismatic	RRPRP
	5.12.	Revolute, Prismatic, Revolute, Revolute, Prismatic	RPRRP
	5.13.	Prismatic, Revolute, Revolute, Revolute, Prismatic	PRRRP
	5.14.	Revolute, Prismatic, Revolute, Prismatic, Revolute	RPRPR
	5.15.	Prismatic, Revolute, Revolute, Prismatic, Revolute	PRRPR
	5.16.	Prismatic, Revolute, Prismatic, Revolute, Revolute	PRPRR
	5.17.	Prismatic, Prismatic, Prismatic, Prismatic, Prismatic	PPPPP

Degrees of Freedom	Architectural Style No	Architectural Style	Simplified Architecture
	5.18.	Prismatic, Prismatic, Prismatic, Prismatic, Revolute	PPPPR
	5.19.	Prismatic, Prismatic, Prismatic, Revolute, Prismatic	PPPRP
	5.20.	Prismatic, Prismatic, Revolute, Prismatic, Prismatic	PPRPP
	5.21.	Prismatic, Revolute, Prismatic, Prismatic, Prismatic	PRPPP
	5.22.	Revolute, Prismatic, Prismatic, Prismatic, Prismatic	RPPPP
	5.23.	Prismatic, Prismatic, Prismatic, Revolute, Revolute	PPPRR
	5.24.	Prismatic, Prismatic, Revolute, Revolute, Prismatic	PPRRP
	5.25.	Prismatic, Revolute, Revolute, Prismatic, Prismatic	PRRPP
	5.26.	Revolute, Revolute, Prismatic, Prismatic, Prismatic	RRPPP
	5.27.	Prismatic, Revolute, Prismatic, Revolute,	PRPRP

Degrees of Freedom	Architectural Style No	Architectural Style	Simplified Architecture
		Prismatic,	
	5.28.	Revolute, Prismatic, Revolute, Prismatic, Prismatic	RPRPP
	5.29.	Revolute, Prismatic, Prismatic, Prismatic, Revolute	RPPPR
	5.30.	Revolute, Prismatic, Prismatic, Revolute, Prismatic	RPPRP
	5.31.	Prismatic, Prismatic, Revolute, Prismatic, Revolute	PPRPR
	5.32.	Prismatic, Revolute, Prismatic, Prismatic, Revolute	PRPPR

APPENDIX B

DIFFERENT DEGREES OF FREEDOM USING MATLAB

1DOF =

$$[\ c\theta_1,\ -c\alpha_1\ s\theta_1,\ \ s\alpha_1\ s\theta_1,\ a\ c\theta_1] \tag{1}$$

$$[\ s\theta_1,\ \ c\alpha_1\ c\theta_1,\ -c\theta_1\ s\theta_1,\ a\ s\theta_1] \tag{2}$$

$$[\quad 0,\quad s\alpha_1,\quad c\alpha_1,\quad d] \tag{3}$$

$$[\quad 0,\quad 0,\quad 0,\quad 1] \tag{4}$$

2DOF =

$$[\ c\theta_{12} - c\alpha_1\ s\theta_{12},\ s\alpha_{12}\ s\theta_1 - c\alpha_2\ c\theta_1\ s\theta_2 - c\alpha_{12}\ c\theta_2\ s\theta_1,\ c\theta_2\ s\alpha_1\ s\theta_1 + c\theta_1\ s\alpha_2\ s\theta_2 + c\alpha_1\ c\theta_2\ s\theta_{12},$$
$$a\ c\theta_1 + a_2\ c\theta_{12} + d_2\ s\alpha_1\ s\theta_1 - a_2\ c\alpha_1\ s\alpha_2\ s\theta_1] \tag{5}$$

$$[\ c\theta_2\ s\theta_1 + c\alpha_1\ c\theta_1\ s\theta_2,\ c\alpha_{12}\ c\theta_{12} - c\alpha_2\ s\theta_{12} - c\theta_1\ s\alpha_2\ s\theta_1,\ s\alpha_2\ s\theta_{12} - c\theta_{12}\ s\theta_1 - c\alpha_1\ c\theta_{12}\ s\theta_2,$$
$$a\ s\alpha_1 + a_2\ c\theta_2\ s\theta_1 - d_2\ c\theta_1\ s\theta_1 + a_2\ c\alpha_1\ c\theta_1\ s\alpha_2] \tag{6}$$

$$[\quad s\alpha_1\ s\theta_2,\quad c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1,\quad c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2,$$
$$d + d_2\ c\theta_1 + a_2\ s\alpha_{12}] \tag{7}$$

$$[0,\quad 0,\quad 0,\quad 1] \tag{8}$$

3DOF =

$$\begin{bmatrix} c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}) - s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1), & s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) \\ - c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}), & c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) \\ + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}), & a\,c\theta_1 + d_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) \\ + a_2\,c\theta_{12} + d_2\,s\alpha_1\,s\theta_1 - a_3\,s\alpha_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + a_3\,c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}) - a_2\,c\alpha_1\,s\alpha_2\,s\theta_1 & \end{bmatrix} \quad (9)$$

$$\begin{bmatrix} c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2) - s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}), & -s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) - \\ c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2), & c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - \\ c\theta_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + s\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2), & a\,s\alpha_1 - d_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + \\ a_2\,c\theta_2\,s\theta_1 - d_2\,c\theta_1\,s\theta_1 - a_3\,s\alpha_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + a_3\,c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2) + a_2\,c\alpha_1\,c\theta_1\,s\alpha_2 & \end{bmatrix} \quad (10)$$

$$\begin{bmatrix} s\theta_3\,(c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) + c\theta_3\,s\alpha_1\,s\theta_2, & s\alpha_3\,(c\theta_{12} - c\theta_2\,s\alpha_1\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) - c\alpha_3\,s\alpha_1\,s\theta_{23}, \\ c\theta_3\,(c\theta_{12} - c\theta_2\,s\alpha_1\,s\theta_2) - c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) + s\alpha_{13}\,s\theta_{23}, & d + d_3\,(c\theta_{12} - c\theta_2\,s\alpha_1\,s\theta_2) + d_2\,c\theta_1 + a_3\,s\alpha_3 \\ (c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) + a_2\,s\alpha_{12} + a_3\,c\theta_3\,s\alpha_1\,s\theta_2 & \end{bmatrix} \quad (11)$$

$$\begin{bmatrix} 0, & 0, & 0, & 1 \end{bmatrix} \quad (12)$$

4DOF =

$$\begin{bmatrix} -c\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - s\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) \\ - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})), \\ s\alpha_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) \\ + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + c\alpha_4\,s\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) \\ - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - c\alpha_4\,c\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) \\ + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})), \\ c\theta_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) \\ + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - s\alpha_4\,s\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + c\theta_4\,s\theta_4\,(c\alpha_3\,c\theta_3 \\ (c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})), \\ d_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) \\ + a\,c\theta_1 + d_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) - a_4\,s\alpha_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 \\ + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + a_2\,c\theta_{12} + d_2\,s\alpha_1\,s\theta_1 - a_3\,s\alpha_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) \\ - a_4\,c\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + a_3\,c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}) - a_2\,c\alpha_1\,s\alpha_2\,s\theta_1 & \end{bmatrix} \quad (13)$$

$$[- c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - s\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)), s\alpha_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - c\alpha_4 c\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)), c\theta_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\theta_4 s\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - s\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)), d_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + a\, s\alpha_1 - d_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + a_2 c\theta_2 s\theta_1 - d_2 c\theta_1 s\theta_1 - a_3 s\alpha_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - a_4 c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - a_4 s\alpha_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + a_3 c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) + a_2 c\alpha_1 c\theta_1 s\alpha_2] \quad (14)$$

$$[c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2) + s\theta_4 (s\alpha_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23}), s\alpha_4 (c\theta_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) - c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + s\alpha_{13} s\theta_{23}) - c\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2) + c\alpha_4 c\theta_4 (s\alpha_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23}), c\theta_4 (c\theta_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) - c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + s\alpha_{13} s\theta_{23}) + s\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2) - c\theta_4 s\theta_4 (s\alpha_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23}), d + d_4 (c\theta_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) - c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + s\alpha_{13} s\theta_{23}) + d_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + d_2 c\theta_1 + a_3 s\alpha_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + a_2 s\alpha_{12} + a_4 c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2) + a_4 s\alpha_4 (s\alpha_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23}) + a_3 c\theta_3 s\alpha_1 s\theta_2] \quad (15)$$

$$[\quad 0, \quad 0, \quad 0, \quad 1] \quad (16)$$

5DOF =

$$[s\theta_5 (s\alpha_4 (c\theta_3 (c\theta_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\theta_{12}) + c\theta_3 s\theta_3 (c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) + s\alpha_3 s\theta_3 (c\theta_{12} - c\alpha_1 s\theta_{12})) + c\alpha_4 s\theta_4 (s\theta_3 (c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) - c\theta_3 (c\theta_{12} - c\alpha_1 s\theta_{12})) - c\alpha_4 c\theta_4 (c\alpha_3 c\theta_3 (c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) - s\alpha_3 (c\theta_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_{12} - c\alpha_1 s\theta_{12}))) - c\theta_5 (c\theta_4 (s\theta_3 (c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) - c\theta_3 (c\theta_{12} - c\alpha_1 s\theta_{12})) + s\theta_4 (c\alpha_3 c\theta_3 (c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1$$

$+ \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))),\ s\alpha_5\, (c\theta_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1$
$+ c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - s\alpha_4\, s\theta_4$
$(s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + c\theta_4\, s\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1)$
$- s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) + c\alpha_5\, s\theta_5\, (c\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 +$
$c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + s\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2$
$+ c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) + c\alpha_5\, c\theta_5\, (s\alpha_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3$
$(c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + c\alpha_4\, s\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1)$
$- c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - c\alpha_4\, c\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12})$
$+ c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))),\ c\theta_5\, (c\theta_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1$
$+ \ c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - s\alpha_4\, s\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))$
$+ c\theta_4\, s\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3$
$(c\theta_{12} - c\alpha_1\, s\theta_{12}))) - s\alpha_5\, s\theta_5\, (c\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + s\theta_4\, (c\alpha_3\, c\theta_3$
$(c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) -$
$c\theta_5\, s\theta_5\, (s\alpha_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3$
$(c\theta_{12} - c\alpha_1\, s\theta_{12})) + c\alpha_4\, s\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - c\alpha_4\, c\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2$
$- s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))),\ d_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1$
$+ c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + a\, c\theta_1 + d_5$
$(c\theta_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3$
$(c\theta_{12} - c\alpha_1\, s\theta_{12})) - s\alpha_4\, s\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + c\theta_4\, s\theta_4\, (c\alpha_3\, c\theta_3$
$(c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})))$
$+ d_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) - a_4\, s\alpha_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1$
$+ c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - a_5\, c\theta_5\, (c\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3$
$(c\theta_{12} - c\alpha_1\, s\theta_{12})) + s\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12})$
$+ c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) + a_2\, c\theta_{12} + a_5\, s\alpha_5\, (s\alpha_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2$
$- s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + c\alpha_4\, s\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3$
$(c\theta_{12} - c\alpha_1\, s\theta_{12})) - c\alpha_4\, c\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12})$
$+ c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) + d_2\, s\alpha_1\, s\theta_1 - a_3\, s\alpha_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - a_4\, c\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 -$
$s\alpha_{12}\, s\theta_1 + \ c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + a_3\, c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}) - a_2\, c\alpha_1\, s\alpha_2\, s\theta_1]$ (17)

$[\ s\theta_5\, (s\alpha_4\, (c\theta_3\, s\theta_3\, (c\theta_1\, s\alpha_2\, s\theta_1 + c\alpha_2\, s\theta_{12} - \ c\alpha_{12}\, c\theta_{12}) - c\theta_3\, (c\theta_{12}\, s\theta_1 - s\alpha_2\, s\theta_{12} + c\alpha_1\, c\theta_{12}\, s\theta_2) + s\alpha_3\, s\theta_3\, (c\theta_2\, s\theta_1$
$+ c\alpha_1\, c\theta_1\, s\theta_2)) - c\alpha_4\, c\theta_4\, (s\alpha_3\, (c\theta_{12}\, s\theta_1 - s\alpha_2\, s\theta_{12} + c\alpha_1\, c\theta_{12}\, s\theta_2) + c\alpha_3\, c\theta_3\, (c\theta_1\, s\alpha_2\, s\theta_1 + c\alpha_2\, s\theta_{12} - \ c\alpha_{12}\, c\theta_{12}) +$
$c\alpha_3\, s\theta_3\, (c\theta_2\, s\theta_1 + c\alpha_1\, c\theta_1\, s\theta_2)) + c\alpha_4\, s\theta_4\, (s\theta_3\, (c\theta_1\, s\alpha_2\, s\theta_1 + c\alpha_2\, s\theta_{12} - \ c\alpha_{12}\, c\theta_{12}) - c\theta_3\, (c\theta_2\, s\theta_1 + c\alpha_1\, c\theta_1\, s\theta_2))) -$

$c\theta_5\,(c\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + s\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))), s\alpha_5\,(c\theta_4\,(c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + s\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + c\theta_4\,s\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - s\alpha_4\,s\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))) + c\alpha_5\,s\theta_5\,(c\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + s\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))) + c\alpha_5\,c\theta_5\,(s\alpha_4\,(c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + s\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - c\alpha_4\,c\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + c\alpha_4\,s\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))), c\theta_5\,(c\theta_4\,(c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + s\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + c\theta_4\,s\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - s\alpha_4\,s\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))) - s\alpha_5\,s\theta_5\,(c\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + s\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))) - c\theta_5\,s\theta_5\,(s\alpha_4\,(c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + s\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - c\alpha_4\,c\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + c\alpha_4\,s\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))), d_4\,(c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + s\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + d_5\,(c\theta_4\,(c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + s\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + c\theta_4\,s\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - s\alpha_4\,s\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))) + a\,s\alpha_1 - d_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + a_5\,s\alpha_5\,(s\alpha_4\,(c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + s\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - c\alpha_4\,c\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + c\alpha_4\,s\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))) + a_2\,c\theta_2\,s\theta_1 - d_2\,c\theta_1\,s\theta_1 - a_3\,s\alpha_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - a_4\,c\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - a_4\,s\alpha_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - a_5\,c\theta_5\,(c\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) + s\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2))) + a_3\,c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2) + a_2\,c\alpha_1\,c\theta_1\,s\alpha_2]$ (18)

$[\ c\theta_5\,(c\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) + c\theta_3\,s\alpha_1\,s\theta_2) + s\theta_4\,(s\alpha_3\,(c\theta_{12} - c\theta_2\,s\alpha_1\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) - c\alpha_3\,s\alpha_1\,s\theta_{23}\,)) + s\theta_5\,(s\alpha_4\,(c\theta_3\,(c\theta_{12} - c\theta_2\,s\alpha_1\,s\theta_2) - c\theta_3\,s\theta_3\,(c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) + s\alpha_{13}\ s\theta_{23}\,) - c\alpha_4\,s\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) + c\theta_3\,s\alpha_1\,s\theta_2) + c\alpha_4\,c\theta_4\,(s\alpha_3\,(c\theta_{12} - c\theta_2\,s\alpha_1\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2 + c\alpha_2\,c\theta_2\,s\alpha_1) - c\alpha_3\,s\alpha_1\,s\theta_{23}\,)),$

$s\alpha_5\ (c\theta_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) + s\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) +$
$c\theta_3\ s\alpha_1\ s\theta_2) - c\theta_4\ s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) - c\alpha_5\ s\theta_5\ (c\theta_4\ (s\theta_3$
$(c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ ))$
$+ c\alpha_5\ c\theta_5\ (s\alpha_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) - c\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1)$
$+ c\theta_3\ s\alpha_1\ s\theta_2) + c\alpha_4\ c\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )),$
$c\theta_5\ (c\theta_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) + s\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1)$
$+ c\theta_3\ s\alpha_1\ s\theta_2) - c\theta_4\ s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + s\alpha_5\ s\theta_5\ (c\theta_4\ (s\theta_3$
$(c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ ))$
$- c\theta_5\ s\theta_5\ (s\alpha_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) - c\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1)$
$+ c\theta_3\ s\alpha_1\ s\theta_2) + c\alpha_4\ c\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )),$
$d + d_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) + d_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + d_5\ (c\theta_4\ (c\theta_3$
$(c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) + s\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2)$
$- c\theta_4\ s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + d_2\ c\theta_1 + a_3\ s\alpha_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1)$
$+ a_5\ s\alpha_5\ (s\alpha_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) - c\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1)$
$+ c\theta_3\ s\alpha_1\ s\theta_2) + c\alpha_4\ c\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + a_2\ s\alpha_{12} + a_4\ c\theta_4\ (s\theta_3$
$(c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + a_5\ c\theta_5\ (c\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2)$
$+ c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + a_4\ s\alpha_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1)$
$- c\alpha_3\ s\alpha_1\ s\theta_{23}\ ) + a_3\ c\theta_3\ s\alpha_1\ s\theta_2]$ (19)

[0, 0, 0, 1] (20)

APPENDIX C

COMPUTATIONS FOR 2 DOF TO 5 DOF

The computations for 2 DOF to 5 DOF were formulated as follows for programming language:

$$\mathbf{T}_1 = {}^0\mathbf{A}_5 = {}^0\mathbf{A}_1\ {}^1\mathbf{A}_2 = \begin{pmatrix} nx & sx & ax & px \\ ny & sy & ay & py \\ nz & sz & az & pz \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Computation for 2 DOF

$$nx = c\theta_{12} - c\alpha_1 s\theta_{12} \quad (1)$$

$$sx = s\alpha_{12} s\theta_1 - c\alpha_2 c\theta_1 s\theta_2 - c\alpha_{12} c\theta_2 s\theta_1 \quad (2)$$

$$\alpha x = c\alpha_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\alpha_2 s\theta_1 \quad (3)$$

$$px = a_1 c\theta_1 + a_2 c\theta_{12} + d_2 s\alpha_1 s\theta_1 - a_2 c\alpha_1 s\theta_1 \quad (4)$$

$$ny = c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2 \quad (5)$$

$$sy = c\alpha_{12} c\theta_{12} - c\alpha_2 s\theta_{12} - c\theta_1 s\alpha_{12} \quad (6)$$

$$\alpha y = s\alpha_2 s\theta_{12} - c\alpha_2 c\theta_1 s\alpha_1 - c\alpha_1 c\theta_{12} s\alpha_2 \quad (7)$$

$$py = a_1 s\theta_1 + a_2 c\theta_2 s\theta_1 - c\theta_1 d_2 s\alpha_1 + a_2 c\alpha_1 c\theta_1 s\theta_2 \quad (8)$$

$$nz = s\alpha_1 s\theta_2 \quad (9)$$

$$sz = c\alpha_{12} s\alpha_{21} c\theta_2 \quad (10)$$

$$\alpha z = c\alpha_{12} - c\theta_2 s\alpha_{12} \quad (11)$$

$$pz = d_1 + c\alpha_1 d_2 + a_2 s\alpha_1 s\theta_2 \quad (12)$$

$$\mathbf{T}_1 = {}^0\mathbf{A}_5 = {}^0\mathbf{A}_1\ {}^1\mathbf{A}_2\ {}^2\mathbf{A}_3 = \begin{pmatrix} nx & sx & ax & px \\ ny & sy & ay & py \\ nz & sz & az & pz \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Computation for 3 DOF

$$nx = c\theta_3\,(c\theta_{12} - c\alpha_1\, s\theta_{12}) - s\theta_3\,(c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) \quad (13)$$

$$sx = s\alpha_3(c\alpha_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\alpha_2 s\theta_1) - c\alpha_3 s\theta_3(c\theta_{12} - c\alpha_1 s\theta_{12}) - c\alpha_3 c\theta_3(c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) \quad (14)$$

$$\alpha x = c\alpha_3(c\alpha_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\alpha_2 s\theta_1) + s\alpha_3 s\theta_3(c\theta_{12} - c\alpha_1 s\theta_{12}) + c\theta_3 s\alpha_3(c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_1 c\alpha_2 c\theta_2 s\theta_1) \quad (15)$$

$$px = a_1 c\theta_1 + d_3(c\alpha_2 s\alpha_1 s\theta_1 + c\theta_1 s\alpha_2 s\theta_2 + c\alpha_1 c\theta_2 s\alpha_2 s\theta_1) + a_3 c\theta_3(c\theta_{12} - c\alpha_1 s\theta_{12}) + a_2 c\theta_{12} + d_2 s\alpha_1 s\theta_1 - a_3 s\theta_3(c\alpha_2 c\theta_1 s\theta_2 - s\alpha_{12} s\theta_1 + c\alpha_{12} c\theta_2 s\theta_1) - a_2 c\alpha_1 s\theta_1 s\theta_2 \quad (16)$$

$$ny = c\theta_3(c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) - s\theta_3(c\theta_1 s\alpha_{12} + c\alpha_2 s\theta_1 s\theta_2 - c\alpha_1 c\alpha_2 c\theta_{12}) \quad (17)$$

$$sy = - s\alpha_3(c\alpha_2 c\theta_1 s\alpha_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\alpha_2) - c\alpha_3 s\theta_3(c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) - c\alpha_3 c\theta_3(c\theta_1 s\alpha_{12} + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) \quad (18)$$

$$\alpha y = s\alpha_3 s\theta_3(c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) - c\alpha_3(c\alpha_2 c\theta_1 s\alpha_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\alpha_2) + c\theta_3 s\alpha_3(c\theta_1 s\alpha_{12} + c\alpha_2 s\theta_1 s\theta_2 - c\alpha_{12} c\theta_{12}) \quad (19)$$

$$py = a_1 s\theta_1 - d_3(c\alpha_2 c\theta_1 s\alpha_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\alpha_2) + a_3 c\theta_3(c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) + a_2 c\theta_2 s\theta_1 - c\theta_1 d_2 s\alpha_1 - a_3 s\theta_3(c\theta_1 s\alpha_{12} + c\alpha_2 s\theta_1 s\theta_2 - c\alpha_{12} c\theta_{12}) + a_2 c\alpha_1 c\theta_1 s\theta_2 \quad (20)$$

$$nz = s\theta_3(c\alpha_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2 \quad (20)$$

$$sz = s\alpha_3(c\alpha_1 c\alpha_2 - c\theta_2 s\alpha_{12}) + c\alpha_3 c\theta_3(c\alpha_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23} \quad (22)$$

$$\alpha z = c\alpha_3(c\alpha_{12} - c\theta_2 s\alpha_{12}) - c\theta_3 s\alpha_3(c\alpha_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + s\alpha_{13} * s\theta_{23} \quad (23)$$

$$pz = d_1 + c\alpha_1 d_2 + d_3(c\alpha_{12} - c\theta_2 * s\alpha_{12}) + a_3 s\theta_3(c\alpha_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + a_2 s\alpha_1 s\theta_2 + a_3 c\theta_3 s\alpha_1 s\theta_2 \quad (24)$$

$$\mathbf{T}_1 = {}^0\mathbf{A}_5 = {}^0\mathbf{A}_1\ {}^1\mathbf{A}_2\ {}^2\mathbf{A}_3\ {}^3\mathbf{A}_4 = \begin{pmatrix} nx & sx & ax & px \\ ny & sy & ay & py \\ nz & sz & az & pz \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Computation for 4 DOF

$$nx = -c\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - s\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) \quad (25)$$

$$sx = s\alpha_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + c\alpha_4\,s\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - c\alpha_4\,c\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) \quad (26)$$

$$ax = c\theta_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - s\alpha_4\,s\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + c\theta_4\,s\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) \quad (27)$$

$$px = d_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + a\,c\theta_1 + d_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) - a_4\,s\alpha_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + a_2\,c\theta_{12} + d_2\,s\alpha_1\,s\theta_1 - a_3\,s\alpha_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - a_4\,c\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + a_3\,c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}) - a_2\,c\alpha_1\,s\alpha_2\,s\theta_1 \quad (28)$$

$$ny = -c\theta_4\,(s\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) - c\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) - s\theta_4\,(s\alpha_3\,(c\theta_{12}\,s\theta_1 - s\alpha_2\,s\theta_{12} + c\alpha_1\,c\theta_{12}\,s\theta_2) + c\alpha_3\,c\theta_3\,(c\theta_1\,s\alpha_2\,s\theta_1 + c\alpha_2\,s\theta_{12} - c\alpha_{12}\,c\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_2\,s\theta_1 + c\alpha_1\,c\theta_1\,s\theta_2)) \quad (29)$$

$$sy = s\alpha_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - c\alpha_4 c\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) \quad (30)$$

$$\alpha y = c\theta_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\theta_4 s\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - s\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) \quad (31)$$

$$py = d_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + a\, s\alpha_1 - d_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + a_2 c\theta_2 s\theta_1 - d_2 c\theta_1 s\theta_1 - a_3 s\alpha_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - a_4 c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - a_4 s\alpha_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + a_3 c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2) + a_2 c\alpha_1 c\theta_1 s\alpha_2 \quad (32)$$

$$nz = c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2) + s\theta_4 (s\alpha_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23}) \quad (33)$$

$$sz = sa_4(ca_3(ca_{12} - c\theta_2 sa_{12}) - c\theta_3 sa_3(ca_1 sa_2 + ca_2 c\theta_2 sa_1) + sa_{13} s\theta_{23}) - ca_4 s\theta_4 (st_3(ca_1 sa_2 + ca_2 c\theta_2 sa_1) + c\theta_3 sa_1 s\theta_2) + ca_4 c\theta_4 (sa_3(ca_{12} - c\theta_2 sa_{12}) + ca_3 c\theta_3 (ca_1 sa_2 + ca_2 c\theta_2 sa_1) - ca_3 sa_1 s\theta_{23}) \quad (34)$$

$$\alpha z = c\theta_4 (c\theta_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) - c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + s\alpha_{13} s\theta_{23}) + s\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2) - c\theta_4 s\theta_4 (s\alpha_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23}) \quad (35)$$

$$pz = d + d_4 (c\theta_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) - c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + s\alpha_{13} s\theta_{23}) + d_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + d_2 c\theta_1 + a_3 s\alpha_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + a_2 s\alpha_{12} + a_4 c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) + c\theta_3 s\alpha_1 s\theta_2) + a_4 s\alpha_4 (s\alpha_3 (c\theta_{12} - c\theta_2 s\alpha_1 s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 + c\alpha_2 c\theta_2 s\alpha_1) - c\alpha_3 s\alpha_1 s\theta_{23}) + a_3 c\theta_3 s\alpha_1 s\theta_2 \quad (36)$$

$$T_1 = {}^0A_5 = {}^0A_1\ {}^1A_2\ {}^2A_3\ {}^3A_4\ {}^4A_5 = \begin{pmatrix} nx & sx & ax & px \\ ny & sy & ay & py \\ nz & sz & az & pz \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Computation for 5 DOF

$$nx = s\theta_5\,(s\alpha_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + c\alpha_4\,s\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - c\alpha_4\,c\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}))) - c\theta_5\,(c\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + s\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})))$$ (37)

$$sx = s\alpha_5\,(c\theta_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - s\alpha_4\,s\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + c\theta_4\,s\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}))) + c\alpha_5\,s\theta_5\,(c\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + s\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}))) + c\alpha_5\,c\theta_5\,(s\alpha_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + c\alpha_4\,s\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - c\alpha_4\,c\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})))$$ (38)

$$\alpha x = c\theta_5\,(c\theta_4\,(c\theta_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\theta_3\,s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) + s\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) - s\alpha_4\,s\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + c\theta_4\,s\theta_4\,(c\alpha_3\,c\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - s\alpha_3\,(c\theta_2\,s\alpha_1\,s\theta_1 + c\theta_1\,s\alpha_2\,s\theta_2 + c\alpha_1\,c\theta_2\,s\theta_{12}) + c\alpha_3\,s\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12}))) - s\alpha_5\,s\theta_5\,(c\theta_4\,(s\theta_3\,(c\alpha_2\,c\theta_1\,s\theta_2 - s\alpha_{12}\,s\theta_1 + c\alpha_{12}\,c\theta_2\,s\theta_1) - c\theta_3\,(c\theta_{12} - c\alpha_1\,s\theta_{12})) + s\theta_4\,(c\alpha_3\,c\theta_3$$

$$(c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) - c\theta_5\, s\theta_5\, (s\alpha_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + c\alpha_4\, s\theta_4$$

$$(s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - c\alpha_4\, c\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) \quad (39)$$

$$px = d_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + a\, c\theta_1 + d_5\, (c\theta_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - s\alpha_4\, s\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + c\theta_4\, s\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) + d_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) - a_4\, s\alpha_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - a_5\, c\theta_5\, (c\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + s\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) + a_2\, c\theta_{12} + a_5\, s\alpha_5\, (s\alpha_4\, (c\theta_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\theta_3\, s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) + s\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + c\alpha_4\, s\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) - c\alpha_4\, c\theta_4\, (c\alpha_3\, c\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - s\alpha_3\, (c\theta_2\, s\alpha_1\, s\theta_1 + c\theta_1\, s\alpha_2\, s\theta_2 + c\alpha_1\, c\theta_2\, s\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}))) + d_2\, s\alpha_1\, s\theta_1 -$$

$$a_3\, s\alpha_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - a_4\, c\theta_4\, (s\theta_3\, (c\alpha_2\, c\theta_1\, s\theta_2 - s\alpha_{12}\, s\theta_1 + c\alpha_{12}\, c\theta_2\, s\theta_1) - c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12})) + a_3\, c\theta_3\, (c\theta_{12} - c\alpha_1\, s\theta_{12}) - a_2\, c\alpha_1\, s\alpha_2\, s\theta_1 \quad (40)$$

$$ny = s\theta_5\, (s\alpha_4\, (c\theta_3\, s\theta_3\, (c\theta_1\, s\alpha_2\, s\theta_1 + c\alpha_2\, s\theta_{12} - c\alpha_{12}\, c\theta_{12}) - c\theta_3\, (c\theta_{12}\, s\theta_1 - s\alpha_2\, s\theta_{12} + c\alpha_1\, c\theta_{12}\, s\theta_2) + s\alpha_3\, s\theta_3\, (c\theta_2\, s\theta_1 + c\alpha_1\, c\theta_1\, s\theta_2)) - c\alpha_4\, c\theta_4\, (s\alpha_3\, (c\theta_{12}\, s\theta_1 - s\alpha_2\, s\theta_{12} +$$

$$c\alpha_1\, c\theta_{12}\, s\theta_2) + c\alpha_3\, c\theta_3\, (c\theta_1\, s\alpha_2\, s\theta_1 + c\alpha_2\, s\theta_{12} - c\alpha_{12}\, c\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_2\, s\theta_1 + c\alpha_1\, c\theta_1\, s\theta_2)) + c\alpha_4\, s\theta_4\, (s\theta_3\, (c\theta_1\, s\alpha_2\, s\theta_1 + c\alpha_2\, s\theta_{12} - c\alpha_{12}\, c\theta_{12}) - c\theta_3\, (c\theta_2\, s\theta_1 + c\alpha_1\, c\theta_1\, s\theta_2))) - c\theta_5\, (c\theta_4\, (s\theta_3\, (c\theta_1\, s\alpha_2\, s\theta_1 + c\alpha_2\, s\theta_{12} - c\alpha_{12}\, c\theta_{12}) - c\theta_3\, (c\theta_2\, s\theta_1 + c\alpha_1\, c\theta_1\, s\theta_2)) + s\theta_4\, (s\alpha_3\, (c\theta_{12}\, s\theta_1 - s\alpha_2\, s\theta_{12} + c\alpha_1\, c\theta_{12}\, s\theta_2) + c\alpha_3\, c\theta_3\, (c\theta_1\, s\alpha_2\, s\theta_1 + c\alpha_2\, s\theta_{12} - c\alpha_{12}\, c\theta_{12}) + c\alpha_3\, s\theta_3\, (c\theta_2\, s\theta_1 + c\alpha_1\, c\theta_1\, s\theta_2))) \quad (41)$$

$$sy = s\alpha_5 (c\theta_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\theta_4 s\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - s\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2))) + c\alpha_5 s\theta_5 (c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + s\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2))) + c\alpha_5 c\theta_5 (s\alpha_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - c\alpha_4 c\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2))) \quad (42)$$

$$\alpha y = c\theta_5 (c\theta_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\theta_4 s\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - s\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2))) - s\alpha_5 s\theta_5 (c\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + s\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2))) - c\theta_5 s\theta_5 (s\alpha_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - c\alpha_4 c\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2))) \quad (43)$$

$$py = d_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + d_5 (c\theta_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 +$$

$$c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) + c\theta_4 s\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + c\alpha_3 c\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) + c\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - s\alpha_4 s\theta_4 (s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2))) + a s\alpha_1 - d_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + a_5 s\alpha_5 (s\alpha_4 (c\theta_3 s\theta_3 (c\theta_1 s\alpha_2 s\theta_1 + c\alpha_2 s\theta_{12} - c\alpha_{12} c\theta_{12}) - c\theta_3 (c\theta_{12} s\theta_1 - s\alpha_2 s\theta_{12} + c\alpha_1 c\theta_{12} s\theta_2) + s\alpha_3 s\theta_3 (c\theta_2 s\theta_1 + c\alpha_1 c\theta_1 s\theta_2)) - c\alpha_4 c\theta_4 (s\alpha_3 (c\theta_{12} s\theta_1 –$$

$s\alpha_2\ s\theta_{12} + c\alpha_1\ c\theta_{12}\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2\ s\theta_1 + c\alpha_2\ s\theta_{12} - c\alpha_{12}\ c\theta_{12}) + c\alpha_3\ s\theta_3$
$(c\theta_2\ s\theta_1 + c\alpha_1\ c\theta_1\ s\theta_2)) + c\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2\ s\theta_1 + c\alpha_2\ s\theta_{12} - c\alpha_{12}\ c\theta_{12}) - c\theta_3$
$(c\theta_2\ s\theta_1 + c\alpha_1\ c\theta_1\ s\theta_2))) + a_2\ c\theta_2\ s\theta_1 - d_2\ c\theta_1\ s\theta_1 - a_3\ s\alpha_3\ (c\theta_1\ s\alpha_2\ s\theta_1 + c\alpha_2\ s\theta_{12} -$

$c\alpha_{12}\ c\theta_{12}) - a_4\ c\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2\ s\theta_1 + c\alpha_2\ s\theta_{12} - c\alpha_{12}\ c\theta_{12}) - c\theta_3\ (c\theta_2\ s\theta_1 + c\alpha_1\ c\theta_1\ s\theta_2)) -$
$a_4\ s\alpha_4\ (s\alpha_3\ (c\theta_{12}\ s\theta_1 - s\alpha_2\ s\theta_{12} + c\alpha_1\ c\theta_{12}\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2\ s\theta_1 + c\alpha_2\ s\theta_{12} - c\alpha_{12}\ c\theta_{12}) +$
$c\alpha_3\ s\theta_3\ (c\theta_2\ s\theta_1 + c\alpha_1\ c\theta_1\ s\theta_2)) - a_5\ c\theta_5\ (c\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2\ s\theta_1 + c\alpha_2\ s\theta_{12} - c\alpha_{12}\ c\theta_{12}) -$
$c\theta_3\ (c\theta_2\ s\theta_1 + c\alpha_1\ c\theta_1\ s\theta_2)) + s\theta_4\ (s\alpha_3\ (c\theta_{12}\ s\theta_1 - s\alpha_2\ s\theta_{12} + c\alpha_1\ c\theta_{12}\ s\theta_2) + c\alpha_3\ c\theta_3$
$(c\theta_1\ s\alpha_2\ s\theta_1 + c\alpha_2\ s\theta_{12} - c\alpha_{12}\ c\theta_{12}) + c\alpha_3\ s\theta_3\ (c\theta_2\ s\theta_1 + c\alpha_1\ c\theta_1\ s\theta_2))) + a_3\ c\theta_3$
$(c\theta_2\ s\theta_1 + c\alpha_1\ c\theta_1\ s\theta_2) + a_2\ c\alpha_1\ c\theta_1\ s\alpha_2$ (44)

$nz = c\theta_5\ (c\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) +$
$c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + s\theta_5\ (s\alpha_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) -$
$c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) - c\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) +$
$c\theta_3\ s\alpha_1\ s\theta_2) + c\alpha_4\ c\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) -$
$c\alpha_3\ s\alpha_1\ s\theta_{23}\ ))$ (45)

$sz = s\alpha_5\ (c\theta_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) +$
$s\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) - c\theta_4\ s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) +$
$c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) - c\alpha_5\ s\theta_5\ (c\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) +$
$c\theta_3\ s\alpha_1\ s\theta_2) + s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) -$
$c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + c\alpha_5\ c\theta_5\ (s\alpha_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) +$
$s\alpha_{13}\ \ s\theta_{23}\ ) - c\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + c\alpha_4\ c\theta_4$
$(s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ ))$ (46)

$\alpha z = c\theta_5\ (c\theta_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) +$
$s\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) - c\theta_4\ s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) +$
$c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + s\alpha_5\ s\theta_5\ (c\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 +$
$c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) -$
$c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) - c\theta_5\ s\theta_5\ (s\alpha_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) +$
$s\alpha_{13}\ \ s\theta_{23}\ ) - c\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + c\alpha_4\ c\theta_4\ (s\alpha_3\ (c\theta_{12} -$
$c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ ))$ (47)

$pz = d + d_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ \ s\theta_{23}\ ) +$
$d_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + d_5\ (c\theta_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) +$

$s\alpha_{13}\ s\theta_{23}\ ) + s\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) - c\theta_4\ s\theta_4\ (s\alpha_3$
$(c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + d_2\ c\theta_1 + a_3\ s\alpha_3$
$(c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + a_5\ s\alpha_5\ (s\alpha_4\ (c\theta_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) - c\theta_3\ s\theta_3\ (c\theta_1\ s\alpha_2 +$
$c\alpha_2\ c\theta_2\ s\alpha_1) + s\alpha_{13}\ s\theta_{23}\ ) - c\alpha_4\ s\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) +$
$c\alpha_4\ c\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) - c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) +$
$a_2\ s\alpha_{12} + a_4\ c\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + a_5\ c\theta_5\ (c\theta_4\ (s\theta_3\ (c\theta_1\ s\alpha_2 +$
$c\alpha_2\ c\theta_2\ s\alpha_1) + c\theta_3\ s\alpha_1\ s\theta_2) + s\theta_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) -$
$c\alpha_3\ s\alpha_1\ s\theta_{23}\ )) + a_4\ s\alpha_4\ (s\alpha_3\ (c\theta_{12} - c\theta_2\ s\alpha_1\ s\theta_2) + c\alpha_3\ c\theta_3\ (c\theta_1\ s\alpha_2 + c\alpha_2\ c\theta_2\ s\alpha_1) -$
$c\alpha_3\ s\alpha_1\ s\theta_{23}\ ) + a_3\ c\theta_3\ s\alpha_1\ s\theta_2$ (48)

APPENDIX D

COMPUTATIONS FOR PROGRAMMING

1DOF =

[cosJointAngle1 -cosLinkTwistAngle1*sinJointAngle1 sinLinkTwistAngle1*sinJointAngle1linkLength1*cosJointAngle1]

[sinJointAngle1 cosLinkTwistAngle1*cosJointAngle1 -sinJointAngle1*cosJointAngle1linkLength1*sinLinkTwistAngle1]

[0 sinLinkTwistAngle1 cosJointAngle1 jointDistance1]

[0+ 0 +0+ 1]

2DOF =

[cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2+
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 - cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1+ cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2+linkLength1*cosJointAngle1 +
linkLength2*cosJointAngle1*cosJointAngle2 + jointDistance2*sinLinkTwistAngle1*sinJointAngle1 -
linkLength2*cosLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1]

[cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2+
cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 -
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1+ sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 -

cosJointAngle1*cosJointAngle2*sinJointAngle1 -
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2+linkLength1*sinLinkTwistAngle1 +
linkLength2*cosJointAngle2*sinJointAngle1 - jointDistance2*cosJointAngle1*sinJointAngle1 +
linkLength2*cosLinkTwistAngle1*cosJointAngle1*sinLinkTwistAngle2]

[sinLinkTwistAngle1*sinJointAngle2+ cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1+ cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2+ jointDistance1 + jointDistance2*cosJointAngle1 +
linkLength2*sinLinkTwistAngle1*sinLinkTwistAngle2]

[0+ 0+ 0+ 1]

3DOF =

[cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2) -
sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 - sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1)+
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) -
cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)+
cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -

sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 -
cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)+linkLength1*cosJointAngle1 +
jointDistance3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) + linkLength2*cosJointAngle1*cosJointAngle2 +
jointDistance2*sinLinkTwistAngle1*sinJointAngle1 -
linkLength3*sinLinkTwistAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
linkLength3*cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2) -
linkLength2*cosLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1]

[cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2) -
sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 -
cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2)+ -
sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) -
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)+
cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 +
cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)+linkLength1*sinLinkTwistAngle1 -

jointDistance3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) + linkLength2*cosJointAngle2*sinJointAngle1 - jointDistance2*cosJointAngle1*sinJointAngle1 - linkLength3*sinLinkTwistAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) + linkLength3*cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2) + linkLength2*cosLinkTwistAngle1*cosJointAngle1*sinLinkTwistAngle2]

[sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2+ sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3+ cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3 + jointDistance1 + jointDistance3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + jointDistance2*cosJointAngle1 + linkLength3*sinLinkTwistAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + linkLength2*sinLinkTwistAngle1*sinLinkTwistAngle2 + linkLength3*cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2]

[0+ 0+ 0+ 1]

4DOF =

[- cosJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 - sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -

cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))+
sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
cosLinkTwistAngle4*cosJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))+
cosJointAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -

sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosJointAngle4*sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))+
jointDistance4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))
+linkLength1*cosJointAngle1 + jointDistance3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) -
linkLength4*sinLinkTwistAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
linkLength2*cosJointAngle1*cosJointAngle2 + jointDistance2*sinLinkTwistAngle1*sinJointAngle1 -
linkLength3*sinLinkTwistAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
linkLength4*cosJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -

sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) - cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) + linkLength3*cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2) - linkLength2*cosLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1]

[- cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) - cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) - sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) + cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))+ sinLinkTwistAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) - cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) + sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) - cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) + cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) + cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -

cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))+
cosJointAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
cosJointAngle4*sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))+
jointDistance4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))
+linkLength1*sinLinkTwistAngle1 - jointDistance3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
linkLength2*cosJointAngle2*sinJointAngle1 - jointDistance2*cosJointAngle1*sinJointAngle1 -
linkLength3*sinLinkTwistAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -

linkLength4*cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) - cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) - linkLength4*sinLinkTwistAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) + cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) + linkLength3*cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2) + linkLength2*cosLinkTwistAngle1*cosJointAngle1*sinLinkTwistAngle2]

[cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) + sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)+ sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) - cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)+ cosJointAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) +

sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) +
sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) -
cosJointAngle4*sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)+
jointDistance1 + jointDistance4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3)
+ jointDistance3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) +
jointDistance2*cosJointAngle1 + linkLength3*sinLinkTwistAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + linkLength2*sinLinkTwistAngle1*sinLinkTwistAngle2 +
linkLength4*cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) +
linkLength4*sinLinkTwistAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)
+ linkLength3*cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2]

[0+ 0+ 0+ 1]

5DOF =

[sinJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
cosLinkTwistAngle4*cosJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))) -
cosJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)))+
sinLinkTwistAngle5*(cosJointAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -

sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosJointAngle4*sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))) +
cosLinkTwistAngle5*sinJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))) +
cosLinkTwistAngle5*cosJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -

sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
cosLinkTwistAngle4*cosJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)))+
cosJointAngle5*(cosJointAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosJointAngle4*sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))) -
sinLinkTwistAngle5*sinJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -

sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))) -
cosJointAngle5*sinJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
cosLinkTwistAngle4*cosJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)))+
jointDistance4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))
+linkLength1*cosJointAngle1 + jointDistance5*(cosJointAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1
+ cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +

cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosJointAngle4*sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))) +
jointDistance3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) -
linkLength4*sinLinkTwistAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
linkLength5*cosJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
sinJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +

cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))) +
linkLength2*cosJointAngle1*cosJointAngle2 +
linkLength5*sinLinkTwistAngle5*(sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 +
cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosJointAngle3*sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) -
cosLinkTwistAngle4*cosJointAngle4*(cosLinkTwistAngle3*cosJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
sinLinkTwistAngle3*(cosJointAngle2*sinLinkTwistAngle1*sinJointAngle1 + cosJointAngle1*sinLinkTwistAngle2*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle2*sinJointAngle1*sinJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2))) +
jointDistance2*sinLinkTwistAngle1*sinJointAngle1 -
linkLength3*sinLinkTwistAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
linkLength4*cosJointAngle4*(sinJointAngle3*(cosLinkTwistAngle2*cosJointAngle1*sinJointAngle2 -
sinLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle2*sinJointAngle1) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2)) +
linkLength3*cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosLinkTwistAngle1*sinJointAngle1*sinJointAngle2) -
linkLength2*cosLinkTwistAngle1*sinLinkTwistAngle2*sinJointAngle1]

[sinJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))) -
cosJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)))+
sinLinkTwistAngle5*(cosJointAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +

cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
cosJointAngle4*sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))) +
cosLinkTwistAngle5*sinJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))) +
cosLinkTwistAngle5*cosJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJo
intAngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 -
cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -

cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)))+
cosJointAngle5*(cosJointAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
cosJointAngle4*sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))) -
sinLinkTwistAngle5*sinJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +

sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))) -
cosJointAngle5*sinJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAn
gle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 -
cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)))+
jointDistance4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +

jointDistance5*(cosJointAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
cosJointAngle4*sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)))
+linkLength1*sinLinkTwistAngle1 - jointDistance3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
linkLength5*sinLinkTwistAngle5*(sinLinkTwistAngle4*(cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointA
ngle1 + cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 -
cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 - sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 +
cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
sinLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +

cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))) +
linkLength2*cosJointAngle2*sinJointAngle1 - jointDistance2*cosJointAngle1*sinJointAngle1 -
linkLength3*sinLinkTwistAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
linkLength4*cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
linkLength4*sinLinkTwistAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) -
linkLength5*cosJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) -
cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2)) +
sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2*sinJointAngle1 -
sinLinkTwistAngle2*sinJointAngle1*sinJointAngle2 + cosLinkTwistAngle1*cosJointAngle1*cosJointAngle2*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2*sinJointAngle1 +
cosLinkTwistAngle2*sinJointAngle1*sinJointAngle2 - cosLinkTwistAngle1*cosLinkTwistAngle2*cosJointAngle1*cosJointAngle2) +
cosLinkTwistAngle3*sinJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2))) +

linkLength3*cosJointAngle3*(cosJointAngle2*sinJointAngle1 + cosLinkTwistAngle1*cosJointAngle1*sinJointAngle2) + linkLength2*cosLinkTwistAngle1*cosJointAngle1*sinLinkTwistAngle2]

[cosJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) + sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)) + sinJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) - cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3))+ sinLinkTwistAngle5*(cosJointAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) + sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle4*sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3))

- cosLinkTwistAngle5*sinJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) +
sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1)
- cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)) +
cosLinkTwistAngle5*cosJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) -
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) +
cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) -
cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3))+
cosJointAngle5*(cosJointAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3)
+ sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) -
cosJointAngle4*sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3))
+ sinLinkTwistAngle5*sinJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) +

sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)) - cosJointAngle5*sinJointAngle5*(sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) - cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3))+jointDistance1 + jointDistance4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) + jointDistance3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + jointDistance5*(cosJointAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) + sinLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle4*sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3))

+ jointDistance2*cosJointAngle1 + linkLength3*sinLinkTwistAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) +
linkLength5*sinLinkTwistAngle5*(sinLinkTwistAngle4*(cosJointAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) - cosJointAngle3*sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + sinLinkTwistAngle1*sinLinkTwistAngle3*sinJointAngle2*sinJointAngle3) -
cosLinkTwistAngle4*sinJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) +
cosLinkTwistAngle4*cosJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3))
+ linkLength2*sinLinkTwistAngle1*sinLinkTwistAngle2 +
linkLength4*cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) +
linkLength5*cosJointAngle5*(cosJointAngle4*(sinJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) + cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2) +
sinJointAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 - cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) +
cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 + cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1)
- cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)) +
linkLength4*sinLinkTwistAngle4*(sinLinkTwistAngle3*(cosJointAngle1*cosJointAngle2 -
cosJointAngle2*sinLinkTwistAngle1*sinJointAngle2) + cosLinkTwistAngle3*cosJointAngle3*(cosJointAngle1*sinLinkTwistAngle2 +
cosLinkTwistAngle2*cosJointAngle2*sinLinkTwistAngle1) - cosLinkTwistAngle3*sinLinkTwistAngle1*sinJointAngle2*sinJointAngle3)
+ linkLength3*cosJointAngle3*sinLinkTwistAngle1*sinJointAngle2]

[0+ 0+ 0+ 1]

APPENDIX E

CD WITH THE SYSTEM DEVELOPED