DOKUZ EYLÜL UNIVERSITY
GARADUATE SCHOLL OF NATURAL AND APPLIED SCIENCES

DESIGN AND ANALYSIS OF ROBOT MANIPULATORS BY INTEGRATED CAE PROCEDURES

by
MURAT AKDAĞ

February, 2008
İZMİR

DESIGN AND ANALYSIS OF ROBOT MANIPULATORS BY INTEGRATED CAE PROCEDURES

A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy in
Machine Theory and Dynamics Program

by

MURAT AKDAĞ

February, 2008
İZMİR

Ph.D. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"DESING AND ANALYSIS OF ROBOT MANIPULATORS BY INTEGRATED CAE PROCEDURES"** completed by **MURAT AKDAĞ** under supervision of **ASSIST. PROF. DR. ZEKİ KIRAL** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Doctor of Philosophy.

.....

Assist. Prof.Dr. Zeki KIRAL

Supervisor

.....	.....
Prof.Dr. Hira KARAGÜLLE	Assist.Prof.Dr. Ahmet ÖZKURT
Thesis Committee Member	Thesis Committee Member
.....	.....
Examining Committee Member	Examining Committee Member

Prof.Dr. Cahit HELVACI

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGEMENTS

I would like to my supervisor, Assist. Prof. Dr. Zeki KIRAL for his very valuable guidance, his support and his critical suggestions throughout my doctoral studies. It was a privilege to study under his supervision.

I am grateful to the members of my doctoral committee, Prof. Dr. Hira KARAGÜLLE and Assist Prof. Dr. Ahmet ÖZKURT, for their careful review and advice during the research.

I would also like to thank my colleagues, Research Assistant Dr. Levent MALGACA, retired instructor Kemal VAROL, Ms.C Thesis student Uğur ERTURUN for their inspiration.

Finally, I wish to express special thanks to my dear wife, GÜLŞAH for her encouragement, patience and love during this doctoral work.

I wish to dedicate this thesis to my parents who have always supported to me.

Murat AKDAĞ

İzmir 2008

DESIGN AND ANALYSIS OF ROBOT MANIPULATORS BY INTEGRATED CAE PROCEDURES

ABSTRACT

Robots are the basic components of production sector. Computer aided engineering methods must be used effectively in the design process since the robot design has various parameters. In this study, for industrial robots, a design process in which the integrated engineering methods are employed, is considered and three different robots are designed following this process. Two of the designed robots have been manufactured and are ready for operation.

The design of the trajectory is of great importance in order to use the robots efficiently. Great attention must be paid on trajetory design in order to reduce the end point deflections which occur at the stoppage, especially in high speed movements. In this study, the importance of the trajectory design is presented by experimental and numerical investigation of strains induced on an industrial robot.

The robots have different structural stiffnesses for different positions since they are movable machines. In this study, a new concept which presents the changes in the stiffness for different positions of the robot within its workspace is presented. A new concept named as “Rigidity Workspace” is investigated according to the end point static deflections and modal behaviour of the robot. A new proposal is made for the method of positional accuracy compansation which is performed for every industrial robot. Besides, the influence of the joint flexibility definition on real end point deflections and modal behavior of robot systems is investigated experimentally and numerically.

Keywords: Robot design, Computer aided engineering, The finite element method, Rigidity workspace, Accuracy compensation.

ROBOT MANİPULATÖRLERİNİN ENTEGRE BİLGİSAYAR DESTEKLİ MÜHENDİSLİK YÖNTEMLERİ İLE TASARIMI VE ANALİZİ

ÖZ

Robotlar, üretim sektöründeki temel bileşenlerdendir. Robot tasarımı pek çok parametreye sahip olduğu için, bilgisayar destekli mühendislik yöntemleri tasarım sürecinde etkili bir şekilde kullanılmadır. Bu çalışmada, endüstriyel robotlar için entegre bilgisayar destekli mühendislik yöntemlerinin kullanıldığı bir tasarım süreci ele alınmış ve bu süreç takip edilerek üç farklı robot tasarımı yapılmıştır. Tasarımı tamamlanan bu robotlardan ikisinin üretimi tamamlanarak çalışır hale getirilmiştir.

Robotların verimli kullanılabilmeleri için yörünge tasarımının önemi büyüktür. Özellikle yüksek hızlı hareketlerde durma anında ortaya çıkan titreşimlerin ve uç nokta sapmalarının azaltılabilmesi için yörünge tasarımına dikkat edilmelidir. Bu çalışmada, yörünge tasarımının önemi, farklı yörüngeler için endüstriyel bir robot üzerinde oluşan zorlanmaların deneysel ve sayısal olarak incelenmesi ile ortaya konulmuştur.

Robotlar hareketli makinalar oldukları için farklı pozisyonlarda farklı yapısal direngenliğe sahiptirler. Bu çalışmada, robotun çalışma uzayı içerisindeki farklı pozisyonları için söz konusu direngenlik değişimlerini ortaya koyan yeni bir kavram sunulmuştur. “Direngenlik Uzayı” olarak adlandırdığımız bu yeni kavram, robotun uç nokta statik çökmesi ve frekans davranışına göre incelenmiştir. Bu kavram kullanılarak, üretilen her endüstriyel robot için yapılan konumsal hassasiyet kompanzasyonu yöntemi için yeni bir öneri yapılmıştır. Ayrıca mafsal esnekliği tanımlamasının robot sistemlerinin gerçek uç nokta sapması ve frekans davranışının belirlenebilmesi üzerindeki etkisi deneysel ve sayısal olarak incelenmiştir.

Anahtar sözcükler: Robot tasarımı, Bilgisayar destekli mühendislik, Sonlu elemanlar yöntemi, Direngenlik çalışma uzayı, Pozisyon hassasiyet kompanzasyonu.

CONTENTS

Page

CHAPTER ONE
INTRODUCTION

1.1 Introduction

The adventure of industrial robots begins with rudimentary robot arms and extends as far as humonoid robots. The speeds and the sensitivities of robot manipulators are increasing as a result of the advances in robot technology. The Computer Aided Engineering (CAE) approach plays an important role on these advances in the robot technology.

1.2 History of Industrial Robots

An industrial robot is officially defined by ISO (ISO Standard 1994) as an "automatically controlled, reprogrammable, multipurpose manipulator programmable in three or more axes". Typical applications of robots include welding, painting, ironing, assembly, pick and place, packaging and palletizing, product inspection, and testing, all accomplished with high endurance, speed, and precision.

George Devol applied for the first robotics patent in 1954 (granted in 1961). The first company to produce a robot was Unimation, founded by George Devol and Joseph F. Engelberger in 1956, and was based on Devol's original patents. Unimation robots were also called "programmable transfer machines" since their main use at first was to transfer objects from one point to another, less than a dozen feet or so apart. First industrial robot Unimate is seen in Figure 1.1. Their robot used hydraulic actuators and was programmed in joint coordinates, i.e. the angles of the various joints were stored during a teaching phase and replayed in operation. They were accurate to within 1/10,000 of an inch. Unimation later licensed their technology to Kawasaki Heavy Industries and Guest-Nettlefolds, manufacturing Unimates in Japan and England, respectively. For some time Unimation's only competitor was Cincinnati Milacron Inc. of Ohio. This changed radically in the late 1970s when several big Japanese conglomerates began producing similar industrial robots.

The octopus-like wall mounted tentacle arm was developed by Marvin Minsky in 1968. Its twelve joints enabled the arm to go around corners. A PDP-6 computer controls the arm, powered by hydraulic fluids. The arm could lift the weight of a person. The robot arm is seen in Figure 1.2.

Victor Scheinman, a mechanical engineering student working in the Stanford Artificial Intelligence Lab (SAIL), created the Stanford Arm in 1969. Figure 1.3 shows the first design of the Stanford Arm. The arm's design becomes a standard and is still influencing the design of robot arms today. Victor Scheinman's Stanford Arm made a breakthrough as the first successful electrically powered, computer-controlled robot arm. By 1974, the Stanford Arm could assemble a Ford Model T water pump, guiding itself with optical and contact sensors. The Stanford Arm led directly to commercial production. Scheinman went on to design the PUMA series of industrial robots for Unimation which are used for automobile assembly and other industrial tasks.

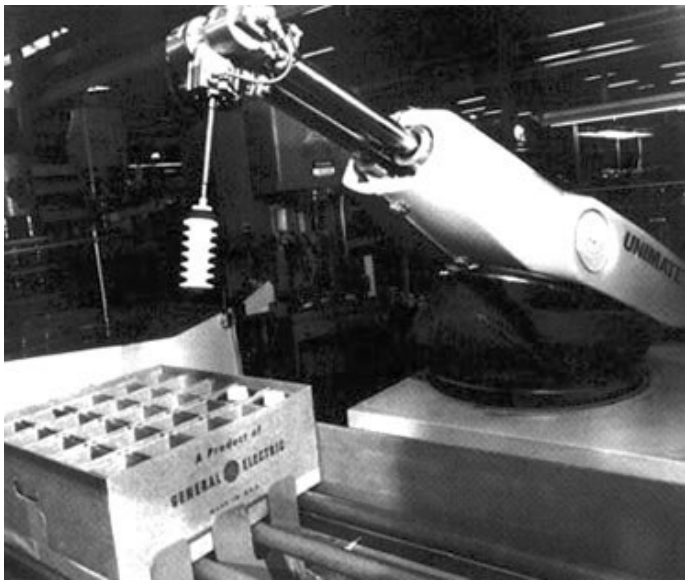


Figure 1.1 UNIMATE, first industrial robot.

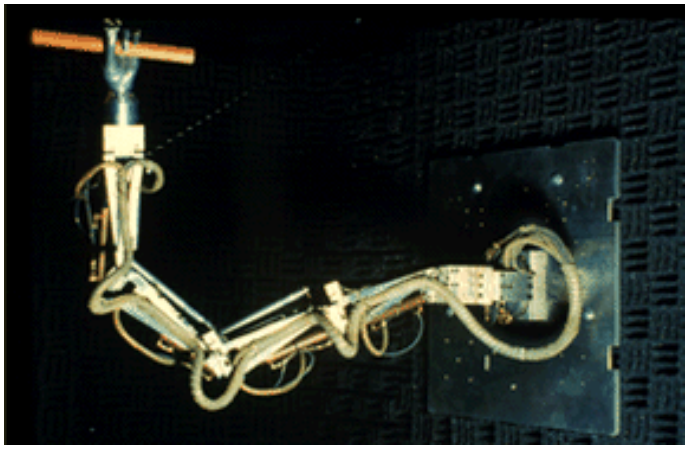


Figure 1.2 Tentacle Arm.

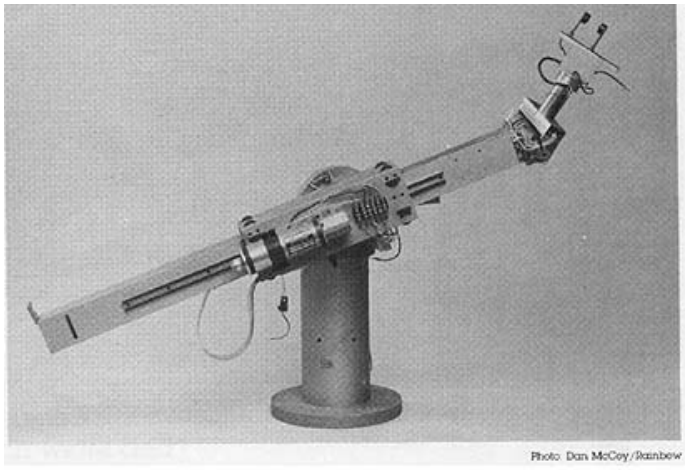


Figure 1.3 First design of the Stanford Arm.

Professor Victor Scheinman of Stanford University designed the Standard Arm in 1970. Today, its kinematic configuration remains known as the Standard Arm.

Cincinnati Milacron Corporation released the T3 in 1973, (The Tomorrow Tool), the first commercially available minicomputer-controlled industrial robot (designed by Richard Hohn) as seen in Figure 1.4.

Figure 1.4 Cincinati Milacron T3.

Victor Scheinman formed his own company and started marketing the Silver Arm in 1974. It was capable of assembling small parts together using feedback from touch and pressure sensors. It is seen in Figure 1.5.

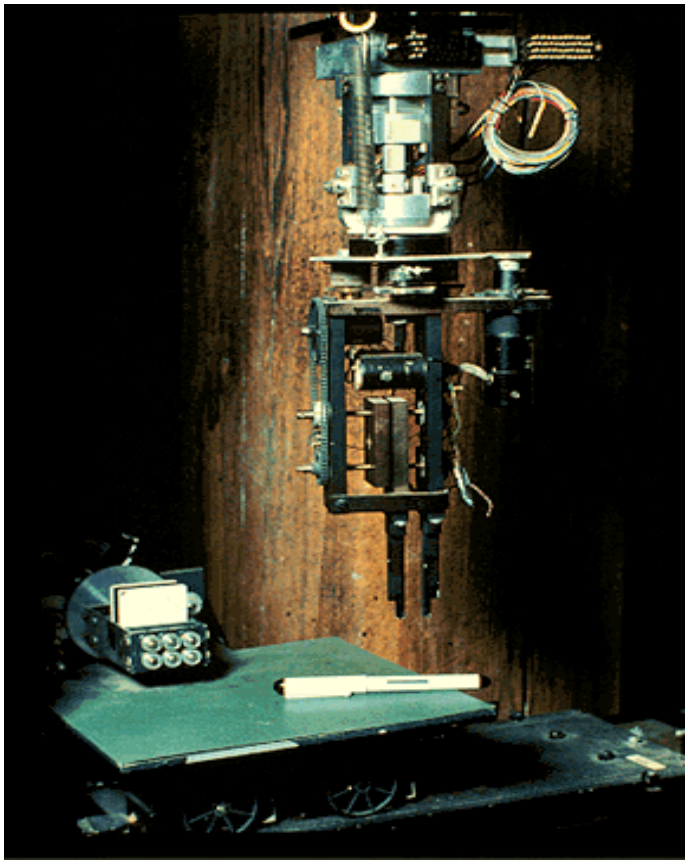


Figure 1.5 Silver Arm.

Victor Scheinman developed the Programmable Universal Manipulation Arm (PUMA), which are widely used as industrial robots. PUMA is seen in Figure 1.6. Unimation purchased Vicarm Inc. in 1977 and using technology from Vicarm, they developed the PUMA robot to be marketed in 1979.

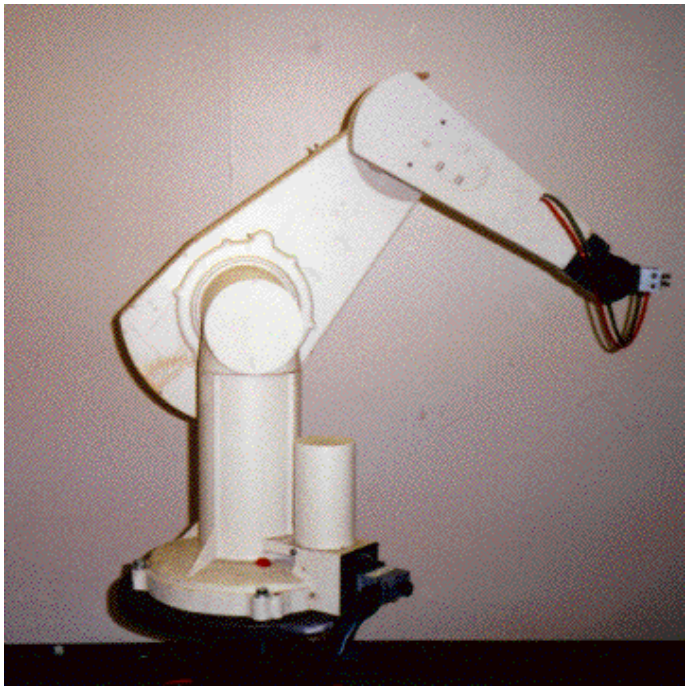


Figure 1.6 Programmable universal manipulation arm (PUMA)

Sankyo and IBM marketed the SCARA (Selective Compliant Articulated Robot Arm) developed at Yamanashi University in Japan. First SCARA robot is seen in Figure 1.7.

In 1981 Takeo Kanade built the direct drive arm. It is the first robot to have motors installed directly into the joints of the arm. This development made the joints faster and much more accurate than previous robotic arms.

Figure 1.7 First selective compliant articulated robot arm (SCARA)

Fanuc of Japan and General Motors formed a joint venture GM Fanuc in 1983. The new company has begun to market robots in North America.

In 1986 Honda began a robot research program that starts with the premise that the robot "should coexist and cooperate with human beings, by doing what a person cannot do and by cultivating a new dimension in mobility to ultimately benefit society."

At MIT Rodney Brooks and A. M. Flynn published the paper in 1989 entitled "Fast, Cheap and Out of Control: A Robot Invasion of the Solar System" in the Journal of the British Interplanetary Society. The paper changed rover research from building the one, big, expensive robot to building lots of little cheap ones. The paper also made the idea of building a robot somewhat more accessible to the average

person. Academics started to concentrate on small, smart useful robots rather than simulated people.

ABB of Switzerland acquired Cincinnati Milacron (creator of PUMA) in 1990. Most small robot manufacturers went out of business leaving only a few that now produce well developed industrial units.

In 1997 the first node of the International Space Station was placed in orbit. Over the next several years more components joined it, including a robotic arm designed by Canadian company MD Robotics.

Honda debuted a new humanoid robot ASIMO in 2000, the next generation of its series of humanoid robots. ASIMO is seen in Figure 1.8.

In October 2000, the UN estimated that there are 742,500 industrial robots in use worldwide. More than half of these are being used in Japan.

Built by MD Robotics of Canada, the Space Station Remote Manipulator System (SSRMS) was successfully launched into orbit and began operations to complete assembly of International Space Station in 2001. The SSRMS is shown in Figure 1.9.

In recent years, robot manufacturers have increased research activities on the light weight robots like Motoman and Kuka. In Figure 1.10, Motoman's light weight robot SDR10 and Kuka's research light weight robot are shown.

Figure 1.8 Honda's ASIMO, next generation of humanoid robots.

Figure 1.9 Space station remote manipulator system (SSRMS)

Figure 1.10 Light weight robot (LWR) applications by Motoman and Kuka

1.3 Literature Survey

This section is arranged as two main parts. The studies related to the robot manipulators are given for design and structural accuracy compensation of robot manipulators.

1.3.1 Robot Design

The optimization of the design of a robot is still an important research subject. Today, the expected abilities of robot manipulators which are the inevitable components in the flexible manufacturing concept are increasing. The robot manipulators have complex structures including different parameters and geometrical constraints. Therefore, the optimal robot design is a comprehensive task. So, it becomes almost an obligation to use Computer Aided Engineering (CAE) procedures in order to reach to an optimal robot design.

One of the basis study related to robot design is presented by Thomson (1984). In his study, he investigated the requirements of the designers and users of such equipment and attempted to evaluate current work in this field. Vukobratovic, Potkonjak, Inoue & Takano (2002) discussed kinds of robot driving systems and described CAD systems for industrial robots. They avoided giving specific constructive solutions and discussed the impact of the actuators on the robot design. They explained the principles of advanced robot design.

Mir-Nasiri (2004) suggested new design of robotic arm with a parallel structure, but with a functionality or geometry similar to the serial structure of a SCARA robot. His new design has a number of advantages compared to a SCARA robot and to other conventional manipulators with parallel structures. This paper and related research aimed at overcoming the problems encountered in the design, modeling and application of such robotic arms.

Mrozek (2003) described two approaches towards designing interdisciplinary mechatronic systems: the first is visual modeling with the UML (Unified Modeling Language), the second is physical modeling with MODELICA. The advantages of UML are investigated on the modeling, understanding and modification of graphical diagrams of mechatronic systems.

Clark & Lin (2007) proposed a CAD-based integration method for analyzing and verifying the design of robotic mechanisms. The unique capabilities of two popular engineering CAD packages, namely, the mathematical computations in MATLAB and the virtual prototyping functions in Pro/Mechanica are blended in their method. Their proposed method realized design automation of robotic mechanisms and provided design flexibility by allowing the designers to change designated critical design parameters rather easily and quickly.

Myung & Han (2001) described the parametric modeling process of machine tool, and proposed a framework which parametrically models a machine tool assembly based on a design expert system which is also applicable for robot manipulators.

Lucchetta, Bariani & Knight (2005) presented a systematic methodology for product simplification through an integration of Design for Manufacture and Assembly (DFMA) with the Theory of Invention Problem Solving (TRIZ). A new functional model is combined with a selection of TRIZ problem solving tools that are identified as effective in product structure simplification. They evaluated alternative concepts by using DFMA analysis.

Bhatia, Thirunarayanan & Dave (1998) presented an expert system-based approach for designing a SCARA robot. Their expert system with the help of its knowledge base carried out the static and dynamic analyses and arrived at the dimensions of the individual parts of the robot. They aimed to reduce the design cycle time of the current four-six month down to a few days for custom-designing a SCARA robot to user specifications.

Morozov & Angeles (2007) focused a novel Schönflies-motion generator (SMG). Schönflies motions are characterized by four degrees of freedom: three independent translations and one rotation about one axis of fixed orientations. The design philosophy and the layout of the SMG are discussed, along with the design procedure, which included: (i) part-modeling and visualization, with animation of the device; (ii) evaluation of the main parameters and the characteristics of the different structural realizations, as well as the selection of one single structure meeting best the design specifications; (iii) the design of the main components for the selected variant of the structure; (iv) the structural and modal analyses and determination of the inertial and elastic parameters of the device and its components; and (v) the production of detailed manufacturing drawings.

Park, Kim, Kim & Park (2007) developed new mid-sized humanoid robot hardware. They focused on the use of an integrated application of CAD/CAM/CAE and rapid prototyping (RP) for the rapid development of the robot's outer body parts. In their study, most parts are designed three-dimensionally with 3D CAD softwares which enable effective connection with CAE analyses, the basis of which laid in kinematic simulation and structural analysis. They applied integrated analyses successfully on the humanoid robot prototype Bonobo.

O'Halloran, Wolf & Choset (2005) presented design, construction and testing of a two-wheeled low cost mobile robot platform. They developed drive transmission system design and integrated suspension system. They optimized design parameters for desired vibration characteristics.

Ouyang, Li & Zhang (2003) described an integrated approach to design a Real Time Controllable (RTC) mechanism considering force balancing and trajectory tracking, simultaneously. They called a new approach as Adjusting Kinematic Parameter (AKP) for the force balancing of RTC mechanisms. Their study demonstrated that the force balancing mechanism by the AKP approach is more promising than those by other approaches in terms of the reduction of joint forces and torques in servomotors, and improvement of the trajectory tracking performance.

Lee & Mavroidis (2003) studied the geometric design of spatial open loop prismatic–revolute–revolute (PRR) robot manipulators. Four spatial positions and orientations are defined and the dimensions of the geometric parameters of the PRR manipulator are computed.

1.3.2 Structural Accuracy Compensation

The number of research and application on flexible robot manipulators is increasing continuously. The end point deflections in robot manipulators increase due to the demand on reducing the weight of the robot. The increase in the end point deflections due to the flexibility negatively affects the accuracy of the robot manipulator. Increasing the accuracy of even today's rigid industrial robots keeps its importance. Different compensation techniques have been developed and new methods will be developed especially for flexible robots.

A comprehensive literature review releated to dynamic analyses of flexible robotic manipulators is presented by Dwivedy & Eberhard (2006). The review of the published papers is classified as modeling, control and experimental studies. In case of modeling, they are subdivided depending on the method of analysis and number of links involved in the analysis. In this work both link and joint flexibilities are considered. Total of 433 papers presented between the years 1974–2005 have been reviewed in this work.

Wang, Xu, Tso & Zhang (1997) investigated path error compensation of a two link flexible robot arm. A planar two-link flexible robot is used as the macro-robot and an integrated laser-PSD transducer is used for measuring the position of the end-effector. A micro robot consisting of translational and rotational joints for compensating the dynamic path errors of the macro flexible robot in real time is used. Simulation and experimental results are given in their investigation.

A new adaptive control method with a learning ability in the repetitive tasks, called Adaptive-Learning (A-L) is described by Sun & Mills (1999). Their method is

based on the proposed theory of two operational modes; the single operational mode and the repetitive operational mode. The advantage of the A-L scheme is discussed. The effectiveness of the A-L scheme in controlling an industrial robot manipulator was demonstrated by theoretical analysis and experimental evaluation on a commercial robot.

Young & Pickin (2000) conducted a trial on three modern serial linkage robots to assess and compare robot accuracy. Laser interferometry measurement system is used for each robot and measurements are done in a similar area of its working range. Their trial is limited only static measurements. The results and conclusions from this trial show that compared to older robots the accuracy can be remarkably good though it is dependent on a calibration process.

Yang, Xu & Tso (2001) considered tip trajectory tracking control problem for multi-link manipulators. They utilized an integrated optical laser sensor system to measure the tip deformations of the flexible links. The Lagrangian assumed-mode method incorporating the measured linear displacements and angular deflections of flexible links is used to derive the dynamic model of the flexible manipulator. The simulation results demonstrated the effectiveness of the proposed approach.

Drouet, Dubowsky & Mavroidis (1998) developed a model to compensate the end-effector error. This method is applied to a high accuracy 6 DOF medical robot that positions patient for cancer therapy. The method explicitly decomposes measured end-point error data into generalized geometric and elastic errors. The experimental results showed that a very good accuracy is obtained enabling this system to meet its design specifications.

Xu, Tso & Wang (1998) proposed a sensor based technique for the modeling and control of the manipulator positioning errors caused by structural defections of the flexible links. A laser-optical sensor system is specially designed for measuring the deflections of each flexible link, and the robot positioning inaccuracies are thus

deduced through the link deflections measured and, finally, compensated for in real time by adjustment of the joint variables.

Alici & Shirinzadeh (2005) presented a systematic approach for representing and estimating the Cartesian positioning errors of robot manipulators with analytical functions such as Fourier polynomials and ordinary polynomials. A Motoman SK 120 robot manipulator is employed as an experimental system to evaluate the efficacy of the approach. The position data needed throughout this study are provided by a laser-based dynamic measurement system. The proposed approximation and estimation approach is verified experimentally for three exemplary Cartesian space trajectories, which describe different configurations of the manipulator.

Abderrahim & Whittaker(1999) aimed to improve the off-line programming capability of industrial robots by improving their accuracy. They proposed a method of identifying kinematic parameters specific to each individual robot. The method is validated on both simulated data and real measured data for a Puma 560 robot, showing an improvement in positioning accuracy of around 80%.

Chin & Lin (1997) constructed a closed loop path precompensation method for a flexible arm robot. A concept of partial deformation compensation is subsequently proposed to improve the torque profiles and the trajectory fidelity. The advantage of this concept is shown by examples of planar trajectory. The torque method and partial deformation compensation are incorporated to track the spatial trajectory. Numerical simulations are given to show the usefulness of the proposed concept and method.

Zhang & Goldberg (2005) developed a fast, low cost and easy-to-operate calibration system for wafer-handling robots. The system is defined by a fixture and a simple compensation algorithm. Given robot repeatability, end-effector uncertainties, and the tolerance requirements of wafer placement points, they derived fixture design and placement specifications based on a statistical tolerance model.

They developed fixture design criteria and a simple compensation algorithm to satisfy calibration requirements.

Shirinzadeh, Teoh, Foong & Liu (1999)proposed laser interferometry-based sensing (LIS) technique and established recently to track and perform dynamic measurements on a moving end-effector of a robot manipulator. The LIS system is used as a sensor to guide the end-effector of a robot manipulator. The structure and various components within the system and the control strategy are presented

Hashimoto & Kiyosawa (1998) proposed the practical torque sensor which utilizes elasticity of harmonic drives. They examined experimentally the characteristics of joint torque control using this torque sensor. Three types of torque control laws are implemented with a one-link arm to find a control method which provides excellent friction reduction and dynamic response in joint torque. They discussed the experimental results of joint torque control and showed that the torque sensor is very useful to compensate the nonlinear friction. They improved the accuracy of the motion control at low velocity and suppressed the vibration caused by the joint flexibility.

Jang, Kim & Kwak (2001) proposed a new approach of calibration which deals with joint angle dependent errors to compensate inaccurate positioning of the robot end-effector caused by joint deformation as well as geometric errors. Robot workspace is divided into several local regions to implement this method and a calibration equation is built by generating the constraint conditions of the end-effector's motion in each local region using a three-dimensional position measurement system. They used this technique to improve the performance of a six DOF industrial robot used for arc welding. Distance accuracy and path accuracy on the workspace is evaluated. The distance accuracy showed a significant improvement from a mean value of 0.920% to 0.154%. Despite this improvement, there is still 3 mm deviation for a movement of 1000 mm.

Karagülle & Malgaca (2004) studied the effect of flexibility on the trajectory of a planar two link manipulator by using integrated computer-aided design/analysis (CAD/CAE) procedures. I-DEAS program is used to create solid models and the finite element models of the parts of the manipulator. The end point vibrations and the deviations from the rigid-body trajectory are analyzed for different types of end point acceleration curves. It is observed that the precision of the manipulator can be increased by testing different end point acceleration curves.

Feliu & Ramos (2005) presented a new control scheme for single-link flexible very light weight robots. The method is strain gauge based and consists of two nested loops: an internal loop that controls the motor dynamics; and an external loop that allows the tip to be positioned in space. They showed that their control scheme is robust to error in parameter estimation and motor parameter changes.

Albu-Schaffer, Haddadin, Ott, Stemmer, Wimböck & Hirzinger (2007) presented a new generation of torque-controlled light-weight robots developed at the Institute of Robotics and Mechatronics of the German Aerospace Center. In their robot concept joint torque sensing plays a central role. Designed robot distinguished from the classical robots such as: load-to-weight ratio of 1:1, torque sensing in the joints, active vibration damping, sensitive collision detection and compliant control on joint and Cartesian level. The first systematic experimental evaluation of possible injuries during robot-human crashes using standardized testing facilities is presented.

1.4 Scope of the Thesis

The design of a robot may become a challenge due to the structural complexities and geometrical restrictions. It is a requirement for an industrial robot to work at high speed and sensitivity together with having a wide range of workspace. This causes to consider lots of parameter simultaneously. So, the robot design is a onerous task even for professional designers. Nowadays, the usage of the CAE procedures in order to manage to accomplish a proper robot design is inevitable.

In the scope of this thesis, an integrated design scheme for robot design including modern CAE procedures is proposed. Three different robot manipulators are designed following the proposed design scheme. The manufacturing process of two robot manipulator is completed and then robots are assembled. The control units of these robots are formed and then the robots are tested.

The dynamic strain and stress behaviors of an industrial robot manipulator are investigated for different trajectories. The effect of the trajectory selection on the dynamic behavior is presented both numerically and experimentally. The ABAQUS finite element package is used effectively for dynamic calculations.

A new workspace concept for manipulator rigidity is introduced for the static end point displacements and the natural frequencies of the considered robot manipulators in their kinematic workspaces. The new workspace is named as the Rigidity Workspace. The Rigidity Workspaces of two manipulators are obtained by the finite element method using the ABAQUS software. The numerical results for the static end-point deflections and the natural frequencies are verified by laser displacement measurements.

1.5 Organization of the Thesis

This thesis consists of six chapters (including the introduction and the conclusions) and the appendices.

Chapter 1 presents the history of industrial robots, literature survey, scope and organization of the thesis. A comprehensive literature survey for related studies is given for robot design and structural accuracy compensation.

Chapter 2 presents the integrated design procedure. The detailed explanations of all stages in the integrated design scheme are given.

Chapter 3 presents the design procedures of three different robot manipulators which are accomplished following the integrated design procedure given in Chapter 2. The parametric solid model of the first robot is created by using ABAQUS. A computer code is developed by MATLAB for inverse kinematic analyses. The kinematic and kinetic analyses are performed by CosmosMotion in order to calculate the joint torques. The detailed models of the robot parts and associated technical drawings are obtained by I-DEAS. The solid models of the second robot are obtained by SolidWorks 2007 software. The kinematic and kinetic analyses are performed by CosmosMotion. The static and frequency analyses are performed by CosmosWorks software. The solid models of the third robot are obtained by SolidWorks 2007 software. The results of the kinematic and kinetic analyses obtained by CosmosMotion are presented. The results of the static and frequency analyses obtained by CosmosWorks software are presented.

Chapter 4 presents the theoretical background of the dynamic analyses employed in the ABAQUS implicit dynamics. The results of the dynamic strain and stress analyses are presented for a real industrial robot both numerically and experimentally.

Chapter 5 presents the new workspace definition called as Rigidity Workspace. The rigidity workspaces of the first and the second robot manipulators are obtained in terms of the static end point deflections and the natural frequencies of the robot assemblies. The static and frequency analyses are verified by experimental measurements for the second robot.

Finally, in chapter 6, the conclusions and the suggestions for the future works for robot design are presented.

The MATLAB code related to the inverse kinematic analyses for the first robot and the script code written in ABAQUS for the parametric design and analysis of the first robot are given in the Appendices.

CHAPTER TWO
INTEGRATED ANALYSIS OF ROBOT DESIGN

2.1 Introduction

Machine manufacturers have to manufacture machine which posses convenient design for the customer's requirements due to increasing competition between the manufacturers in recent years. It is not an effective solution to respond to this demand by producing several machine models. It is important that the design of the machine is carried out so as to satisfy the customer's specific requirements to decrease the manufacturing cost and increase the quality of the product. The design process must be completed rapidly in order to proceed with the manufacturing of the machines which fulfils the customer's requirements. Necessity for quick design has created the flexible design concept. Flexible design can be defined as accomplishing the whole design which fulfils all the requirements in a quick and reliable manner. Flexible design is carried out by using integrated CAE analysis effectively.

The steps of the integrated analysis of design are given as a flow-chart in Figure 2.1. This process begins with job definition for designing machine and finishes with the manufacturing. This process is used by large-scale manufacturers but methods, programs and conclusions are not shared clearly. It is very important to apply the integrated design steps in a clear manner that every manufacturer can follow.

Integrated analysis of design steps are explained for an industrial robot design as follows.

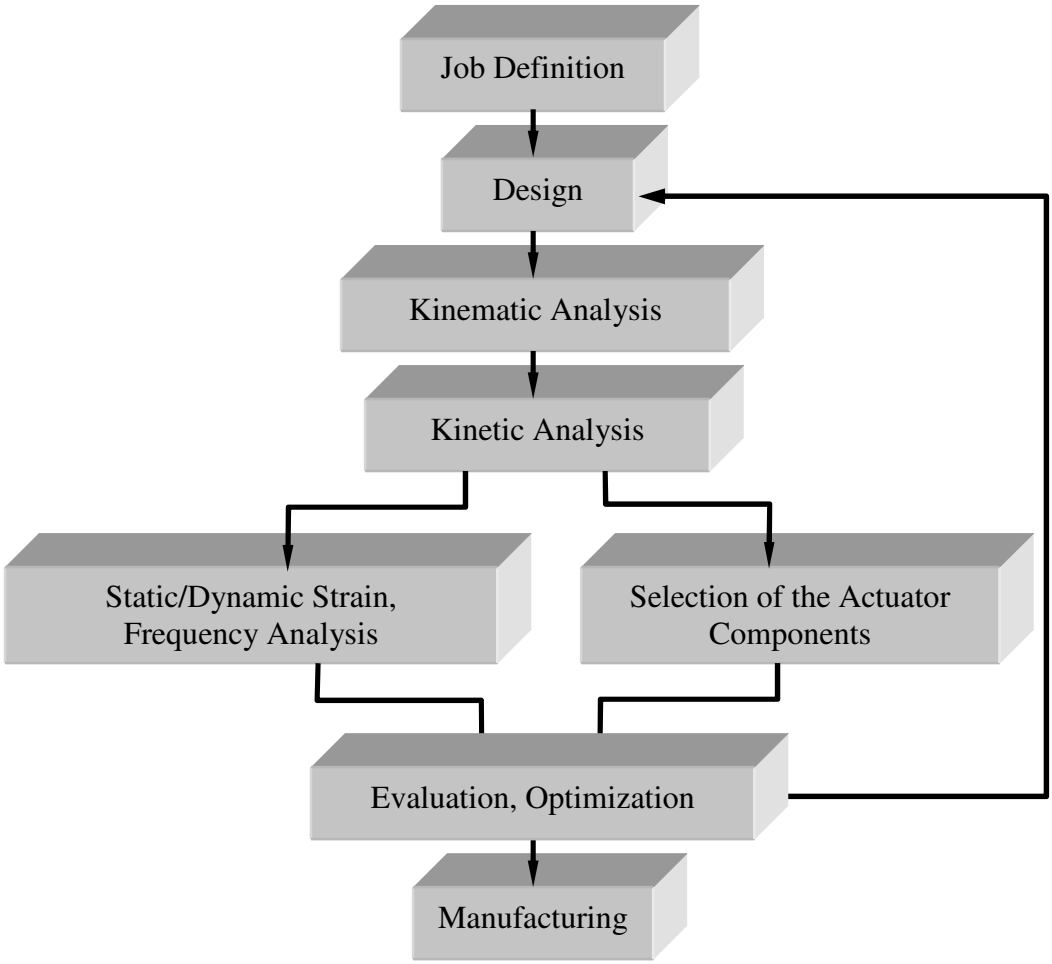


Figure 2.1 Integrated analysis of design scheme

2.2 Job Definition

First of all, job definition should be done very well to determine the whole characteristic parameters for manufactured machine. All details for the job definition are taken into consideration. A machine, for which the job definition is not done properly, does not perform the desired jobs completely or make the jobs with over performance meaning high cost.

Industrial robot usage increases gradually as the flexible manufacturing concept becomes prevalent. Flexible manufacturing concept requires rapid adaptation for different manufacturing conditions. Consequently, only the job definition is not enough for an industrial robot. The job definition for the considered robot must be

done in order to get the desired maximum performance. The characteristic parameters of the industrial robots are payload, reach distance and axes velocities. These parameters are fundamental datum for job definition. Additionally the workspace volume of the robot is desired as large as possible. The workspace of a robot is a measure of its movement ability. Workspace volume of a robot is determined by using selected limb dimensions. It is desired that an industrial robot should reach as far as possible. At the same time, it should work at the points near the robot. So working range can be maximized. Figure 2.2 shows an example workspace of the robot manufactured by ABB.

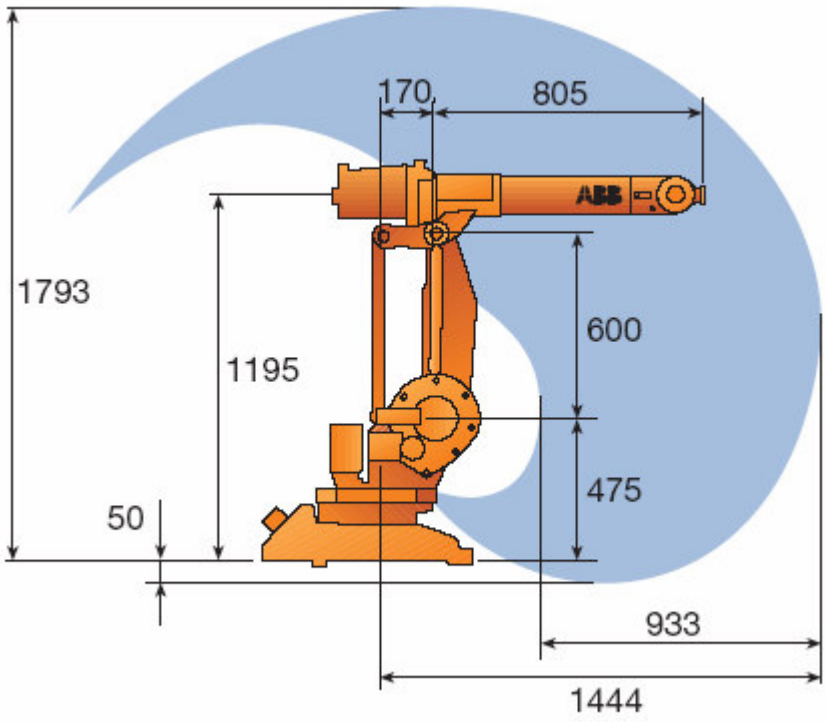


Figure 2.2 Workspace of an ABB robot (ABB, 2007)

2.3 Design

2.3.1 Figural Design

This step begins with drawing the parts of the robot. Drawing helps to determine the general dimensions of the robot limbs. Approximate dimensions of equipments on the robot such as motors and gears are taken into consideration in this level. In the

figural design, the geometrical limitations on the axis freedoms should be taken into consideration. The workspace of the robot becomes larger as the axes movement limits are reduced. Firstly, solid model design is created by using a solid modeling program by taking into account all of these design criteria. In the figural design, there are no details like motor connection holes, shaft steps, rolling element bearings, bolts, nuts and holes on the parts. Shape design is finished after obtaining the overall shape of the robot.

2.3.2 Parametric Design

In this design step, all dimensions on the solid models determined in shape design stage are selected as a parameter. Many kinds of solid modeling programs have a parametric modeling approach. In these programs parameters are defined via script codes or parametric design modules. The parametric design model is very important because changes in dimensions by using the user interfaces for successive analyses are very difficult. In addition it may cause some problems on the model.

2.3.3 Detailed Design

In this design step whole parts of the robot are added in the solid model. All details on the parts are modeled and technical drawings of the whole parts are prepared. In this step if dimensions are changed, these changes must be done in parametric design model and analyses must be repeated. Detailed design is done generally after the analyses and actuator selection.

2.4 Kinematic Analysis

In this step, forward and inverse kinematic calculations are performed for the manufactured robot. Angular displacements and angular velocities of the robot limbs are calculated when the end point of the robot moves on the predefined trajectories. These calculations can be performed by using the commercial FE packages such as CosmosWorks, Working Model and ABAQUS. Working Model or CosmosWorks

softwares are more suitable programs due to the advantages in the solution time and usage simplicity. These programs perform the solutions by using numerical integration methods. For that reason integration steps are selected as small as possible. The results obtained from these programs are quite close to exact solutions. This is very important because the improper selection of time step in the numerical analysis causes the unreal end point deflections.

In the kinematic analysis, it is possible that the end point location or the end point velocity can be given as kinematic inputs. These inputs are calculated by a program which is written by using Visual Basic programming language. The path constructed via this program is a line between two points. Figure 2.3 shows the end point velocity profile.

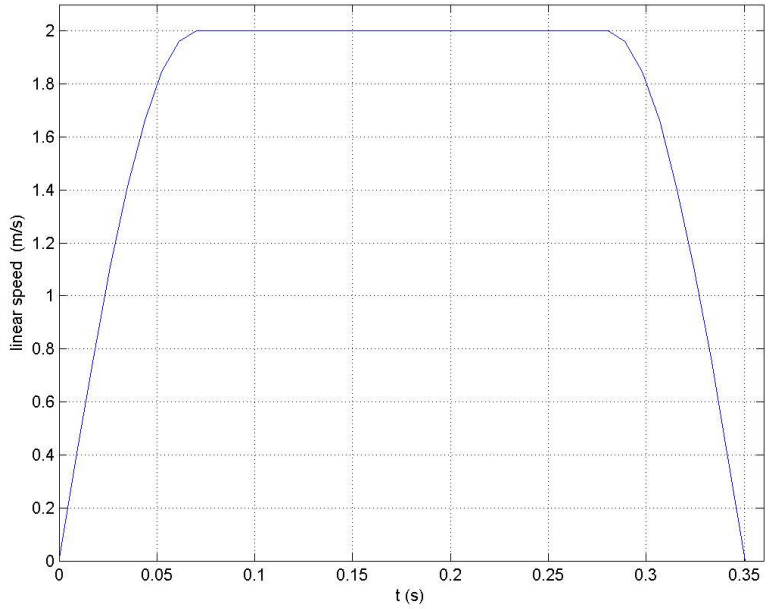


Figure 2.3 Robot end point linear velocity profile

2.5 Kinetic Analysis

In this step, motor moments are calculated for the revolute joints of the robot. After calculating the motor moments, the selection of the motors and gearboxes is possible. Parametric model is used for this analysis. Approximate mass values which are the equivalent masses of the motors and gears must be added to the geometric

model. Constructional changes for the selected parts on the robot must be done. Motor selection is very critical step in the design stage because the motors must be powerful enough to deal with the instantaneous loads. At the same time, they must be as small as possible because of the total weight and constructional considerations of the robot assembly. CosmosWorks program is very useful in the kinetic analysis. Angular velocities which are calculated from the kinematical analysis are used as inputs. In the kinetic analysis, the maximum payload is attached on the end point of the robot and the trajectory which is passed through the point on maximum reach distance is defined.

2.6 Static/Dynamic Strain, Stress and Frequency Analyses

In this step, the static and dynamic strain and stress values are calculated by using the finite element method. The end point deflections can not be calculated by simply defining the start and end points of the trajectory. So the deflections aren't seen at the end point of the robot during the movement. To overcome this difficulty, angular velocities of the joint axes must be entered as movement inputs in order to perform a precise analysis for the end point deflections. These input values are obtained from the kinetic analyses.

In the frequency analysis, different configurations of robot arms are considered. For different configurations of the robot assembly the results of the frequency analyses give information about the rigidity of the robot manipulator in different directions in the workspace. The robot assembly is created by using the parametric modelling technique. For this aim, ABAQUS package is used due to the superiority of the program in the dynamic finite element solutions. In the analyses, the effect of gravity is taken into account and the maximum payload is used. Materials which are used in the robot model are assigned correctly.

In the static analysis, the rotational movement of the axes is restricted by locking two parts associated with the axis. Motor breaks hold position is similar to the situation simulated in the static analysis. Static analysis can be performed for

different axis angles by using parametric models of the robot. So stress distribution on the parts of the robot is investigated for different configurations. Some design changes can be carried out on the robot parts depending on the stress results which are obtained from static analyses. In these analyses, the joint elements are modelled as perfectly rigid components. But in the real model, joint flexibilities must be taken into consideration. This analysis is performed to define only the overall structural rigidity. It is not aimed to calculate the real natural frequencies of the real structure.

The main reason for the selection of ABAQUS program is its real time simulation capability. The effect of the inertia forces on the stress and strain values on the parts can be clearly seen from the results. In the dynamic analyses, axis movements are set to free. The end point of the robot tracks the trajectory defined as a movement input.

2.7 Selection of the Actuator Components

There are two main parameters affecting the selection of the actuator components; first is required moment, second is the axes velocity. Industrial robots have generally the servo motors due to their sensitiveness, precise control and quietness. An example of power curve for a servo motor is shown in Figure 2.4. The continuous usage curve is taken into consideration for the required motor moment which is calculated from the kinetic analysis. Servo motor moment tends to reduce after the known angular speed value. For production purpose, the maximum velocities of the robot axes are calculated by using this known angular speed values. The critical angular speed value is 3000 rpm for a sample servo motor shown in Figure 2.4.

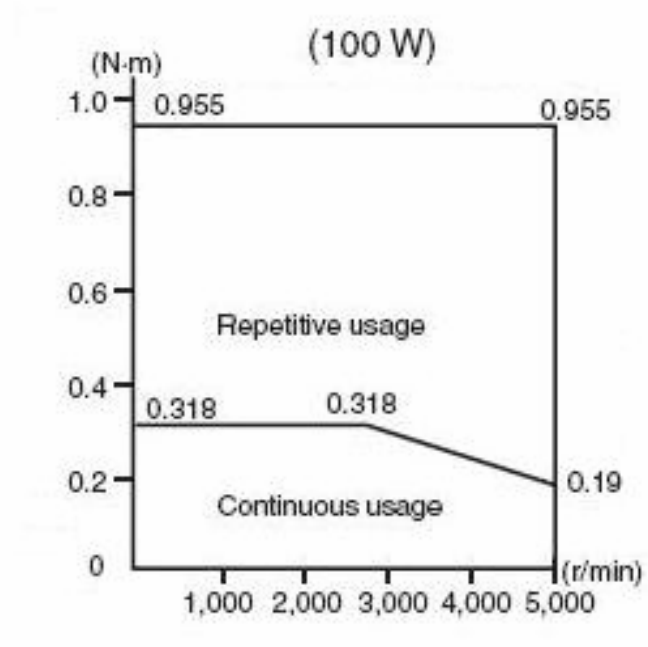


Figure 2.4 Example of servo motor power curve
(Omron, 2006)

Gear ratio must be determined in order to calculate the required axis moment and axis angular velocity. Mathematical expression for the axis moments and velocities are given as

Gear Ratio x Continuous Motor Moment ≥ Required Axis Moment
Gear Ratio x Motor Angular Velocity ≥ Axis Angular Velocity

2.8 Evaluation and Optimization

This step is the final step before the manufacturing. Having evaluated all the studies the necessary modifications are done in the design. After a modification, all design steps should be repeated. There is no distinct limit to finish the optimization studies. It must reach to a decision depending on the design goal. For example, the largest acceptable end point deflection or the smallest acceptable natural frequency must be obtained. Optimization loop is run until the desired aim has realized. The manufacturing stage begins after the modifications have completed.

2.9 Manufacturing

Manufacturing process of the robot parts is very important. Especially the axis parts of the robot must be manufactured within very low tolerances. The bearings on the robot arm must be machined on the CNC machine. The five-axis CNC machines must be used if it is needed.

CHAPTER THREE
APPLICATION OF INTEGRATED ANALYSIS OF DESIGN

3.1 Three Axis Serial Manipulator (DEU-3X2-550)

First design study performed in the scope of this thesis is introduced in this chapter by using integrated analysis of design method.

3.1.1 Job Definition

A desktop robot which has three axes and 2 kg payload is considered in this design study. It is intended that the robot has 500 mm maximum reach distance and 2000 mm/s maximum end point velocity. Generally small class industrial robots have less payloads and high end point velocities.

3.1.2 Design

3.1.2.1 Figural Design

Figural design of the robot begins with free hand drawing. First drawing of the design is seen in Figure 3.1. Main parts of the robot are modelled by using this first drawing. The lengths of the robot arms are dimensioned according to the desired maximum reach distance. Different alternatives are examined for the structural design of the robot arms. Some different alternatives of the second arm structure are seen in Figure 3.2. All these alternatives are evaluated according to rigidity, manufacturability, weight and aesthetic. Final structural design of the second arm is seen in Figure 3.3. These evaluations are done for all robot parts one by one.

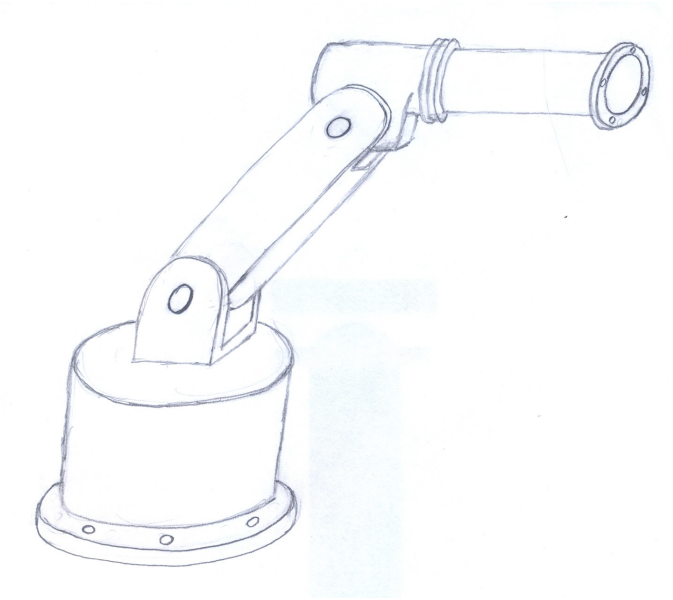


Figure 3.1 First design by free hand drawing

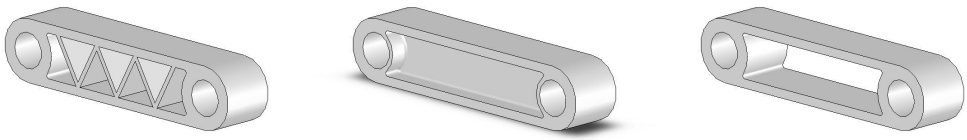


Figure 3.2 Different design alternatives for second arm of the robot

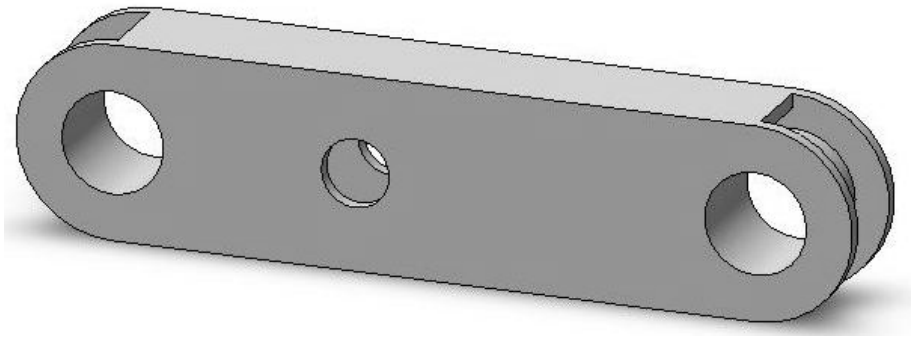
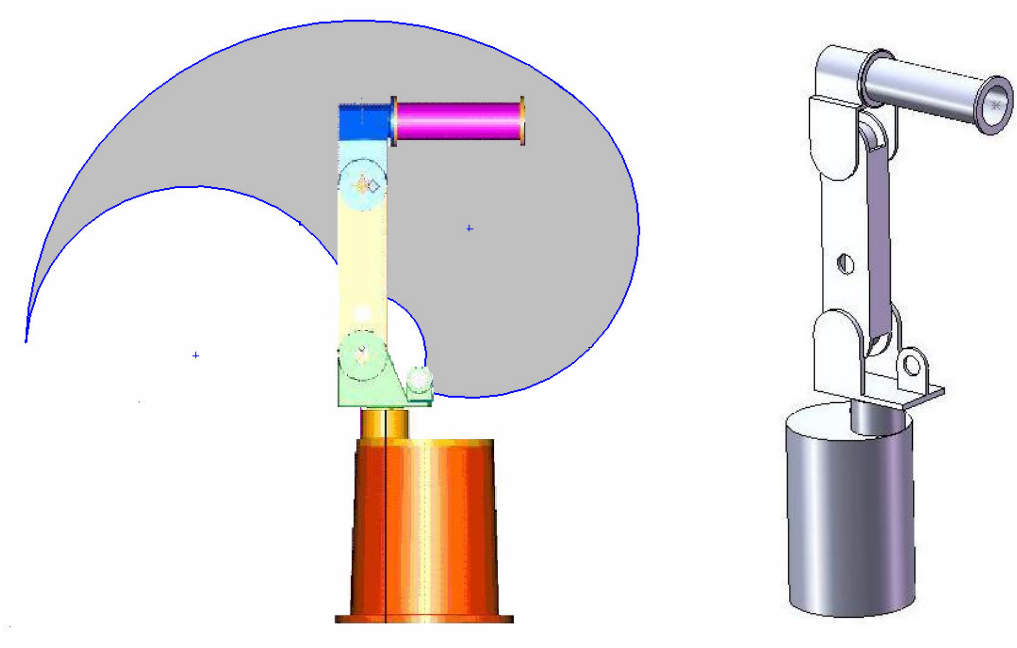


Figure 3.3 Final design for the second arm of the robot

The workspace of the robot manipulator is obtained and examined after the whole main robot parts have modelled. Some modifications are done on the robot structure design to enlarge the workspace of the robot if it is needed. The workspace of the robot DEU-3X2-550 is seen in Figure 3.4.a. Figural design of the robot manipulator is seen in Figure 3.4.b.



(a) (b)

Figure 3.4 a)Workspace of the robot, b) figural design of the robot

3.1.2.2 Parametric Design

Whole dimensions of the robot parts which are taken from the figural design are assigned to a parameter. ABAQUS program is chosen to perform the static and dynamic analysis of the robot due to the some advantages in the modelling and time-dependent analyses. The ABAQUS program is used in modelling of the parametric robot model. The whole solid model of the robot is modelled with a script code written in ABAQUS. This script code is given in the Appendix A2. Dimensional parameters of the robot parts are seen in Figure 3.5.

In the scope of this thesis, kinematic, kinetic, static and dynamic analyses are performed by this parametric design model. Structural and dimensional modifications are done in the parametric model according to results of the analyses. Rigidity of the robot is determined using the results of the maximum end point displacement and natural frequencies of the robot at different position. Required improvements are done by altering the dimensional parameters. In addition, this parametric model is used in the kinematic and kinetic analyses.

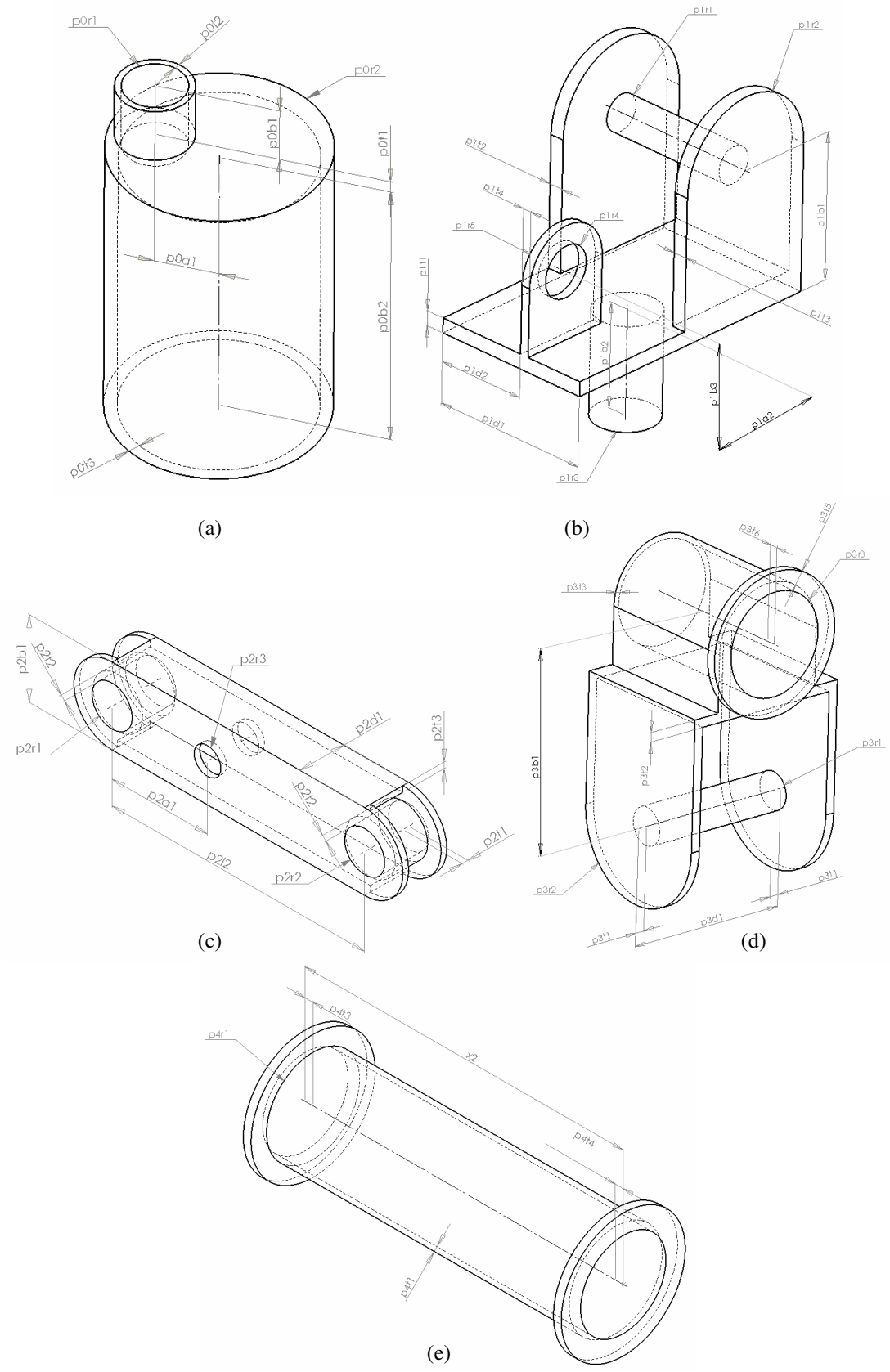


Figure 3.5 Dimensional parameters for the robot parts, a) part P0, b) part P1, c) part P2, d) part P3, e) part P4

3.1.2.3 Detailed Design

In this design step, the detailed model of the robot model created in the parametric design step is carried out. Required details for manufacturing process are determined. Detailed design model of the robot is seen in Figure 3.6.

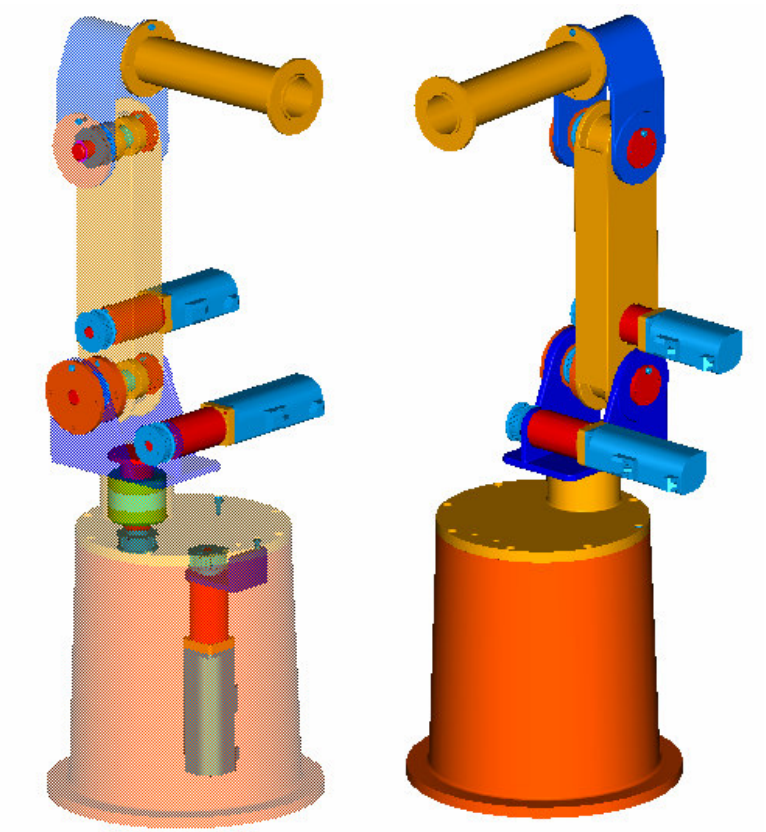


Figure 3.6 Detailed design model of the robot

3.1.3 Kinematic Analysis

3.1.3.1 Path Generation and the Inverse Kinematic Analysis

In this section, the inverse kinematic equations are derived by using the Denavit-Hartenberg notation. A computer code written in MATLAB is developed to calculate the axis displacements for a defined end point trajectory. The code is given in the Appendix A1. The dimensional parameters used in the inverse kinematic calculations are given in Figure 3.7. The link parameters are given in Table 3.1. The animation

view of the robot during the robot is followed the defined path is also obtained by the code. A sample view of the robot movement for a linear path is given in Figure 3.8.

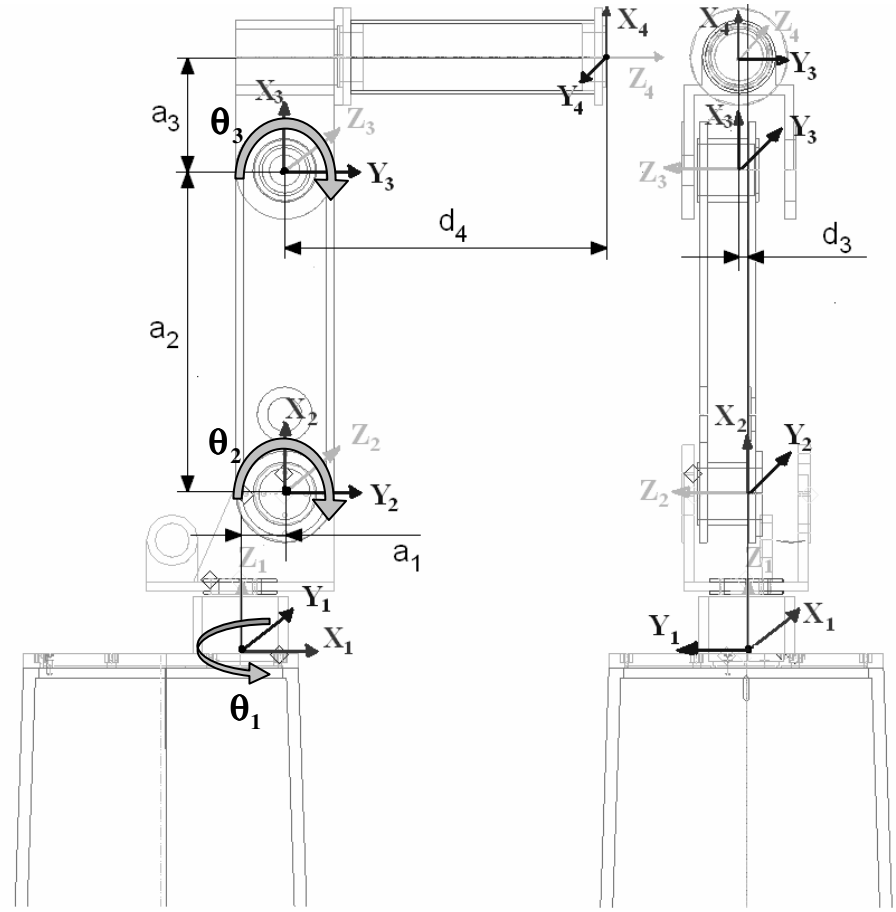


Figure 3.7 Kinematic parameters and axis assignments for the DEU-3X2-550

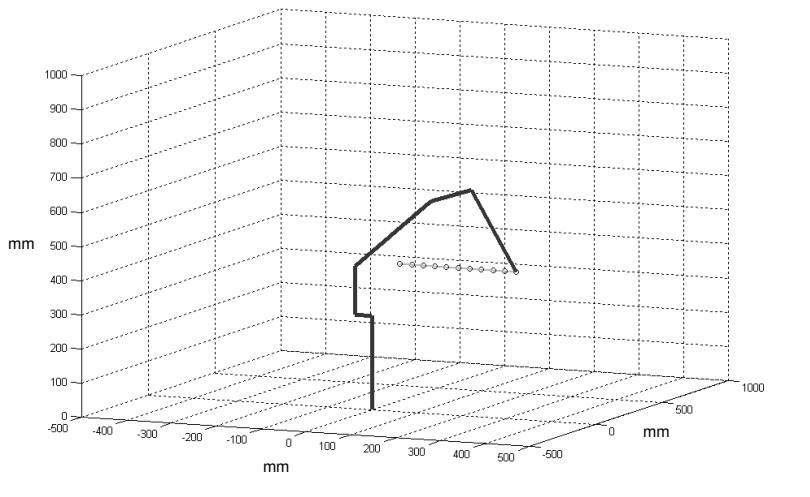


Figure 3.8 A sample view during the robot motion.

Table 3.1 Link parameters of the DEU-3X2-550

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0°	0	0	θ_1
2	-90°	a_1	0	θ_2
3	0°	a_2	d_3	θ_3
4	-90°	a_3	d_4	θ_4

The transformation matrices between the axes are given as (Craig, 1986)

$$ {}^{0}_{1}T = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.1} $$

$$ {}^{1}_{2}T = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2 & 0 & a_1 \\ 0 & 0 & 1 & 0 \\ -\sin\theta_2 & -\cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2} $$

$$ {}^{2}_{3}T = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3 & 0 & a_2 \\ \sin\theta_3 & \cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3} $$

$$ {}^{3}_{4}T = \begin{bmatrix} \cos\theta_4 & -\sin\theta_4 & 0 & a_3 \\ 0 & 0 & 1 & d_4 \\ -\sin\theta_4 & -\cos\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.4} $$

where $c_n = \cos\theta_n$, $s_n = \sin\theta_n$, $c_{23} = c_2c_3 - s_2s_3$ and $s_{23} = c_2s_3 + s_2c_3$

The overall transformation between the base and the end point of the robot is

$$ {}^{0}_{4}\mathrm{T} = \begin{bmatrix} c_1c_{23}c_4 + s_1s_4 & -c_1c_{23}s_4 - s_1c_4 & -c_1s_{23} & c_1(c_{23}a_3 - s_{23}d_4 + c_2a_2 + a_1) - s_1d_3 \\ s_1c_{23}c_4 - c_1s_4 & -s_1c_{23}s_4 + c_1c_4 & -s_1s_{23} & s_1(c_{23}a_3 - s_{23}d_4 + c_2a_2 + a_1) + c_1d_3 \\ -s_{23}c_4 & -s_{23}s_4 & -c_{23} & -s_{23}a_3 - c_{23}d_4 - s_2a_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.5) $$

The last column of Equation (3.5) includes p_x, p_y and p_z, respectively which denotes the end point location.

$$ \left[{}^{1}_{0}\mathrm{T}\right]^{-1} \cdot {}^{0}_{4}\mathrm{T} = {}^{1}_{4}\mathrm{T} \quad (3.6) $$

Equating the (2, 4) elements from both sides of Equation 3.6 we have

$$ -s_1p_x + c_1p_y = d_3 \quad (3.7) $$

Finally using Equation (3.7), the solution for θ_1 may be written as

$$ \theta_1 = \tan^{-1}(p_y, p_x) - \tan^{-1}\left(d_3, \pm\sqrt{p_x^2 + p_y^2 - d_3^2}\right) \quad (3.8) $$

Note that we find two possible solutions for θ_1 corresponding to the plus-or-minus sign in Equation (3.8). Now that θ_1 is known, the left hand side of Equation 3.6 is known. If we equate the (1,4) elements from both side of Equation (3.6) and also the (3,4) elements, we obtain

$$ \begin{aligned} c_1p_x + s_1p_y &= c_{23}a_3 - s_{23}d_4 + c_2a_2 + a_1 \\ -p_z &= s_{23}a_3 + c_{23}d_4 + s_2a_2 \end{aligned} \quad (3.9) $$

If we take the squares of the Equation (3.7) and Equation (3.9) and take the summation of the resulting equations we obtain

$$a_3 c_3 - d_4 s_3 = K \tag{3.10}$$

where K is

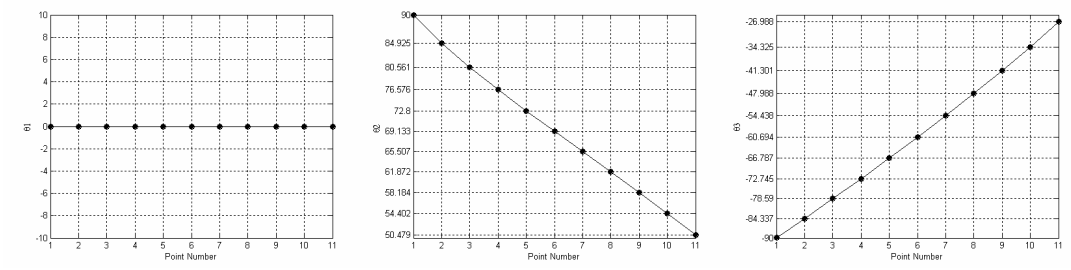
$$K = \frac{p_x^2 + p_y^2 + p_z^2 - a_2^2 - a_3^2 - d_3^2 - d_4^2}{2a_2} \tag{3.11}$$

Note that dependence on θ_1 is removed from Equation (3.10). Equation (3.10) is of the same form as Equation (3.7) and so may be solved by the same kind of trigonometric substitution to yield a solution for θ_3:

$$\theta_3 = \tan^{-1}(a_3, d_4) - \tan^{-1}\left(K, \pm\sqrt{a_3^2 + d_4^2 - K^2}\right) \tag{3.12}$$

The plus or minus sign in Equation (3.12) leads to two different solutions for θ_3. The solution of the Equation (3.9) by putting the θ_1 and θ_3 gives the θ_2. Note that there are four possible solutions according to the four possible combinations of θ_1 and θ_3.

The variations of the axis displacements θ_1, θ_2 and θ_3 are given in Figure 3.9 for the linear path given in Figure 3.8.



(a) (b) (c)

Figure 3.9 The variations of the axis displacements a) θ_1, b) θ_2 and c) θ_3

3.1.3.2 Kinematic Analysis by CosmosMotion

Kinematic analyses of the robot manipulator are carried out by Cosmos Motion software. Axis angular velocities are calculated for desired maximum end point velocity. These are the necessary inputs for the selection of motors and gears. Desired end point velocity is chosen as 2000 mm/s for this robot. Kinematic analyses are performed for different trajectories. These trajectories pass through the maximum reach distance points. Initial and final positions of the robot end point on a sample trajectory are seen in Figure 3.10. The total movement time is decreased until the end point velocity has reached the desired value. The end point velocity profile is seen in Figure 3.11. Angular velocities of the robot axes are calculated according to desired end point velocity. These angular velocities are seen in Figure 3.12.

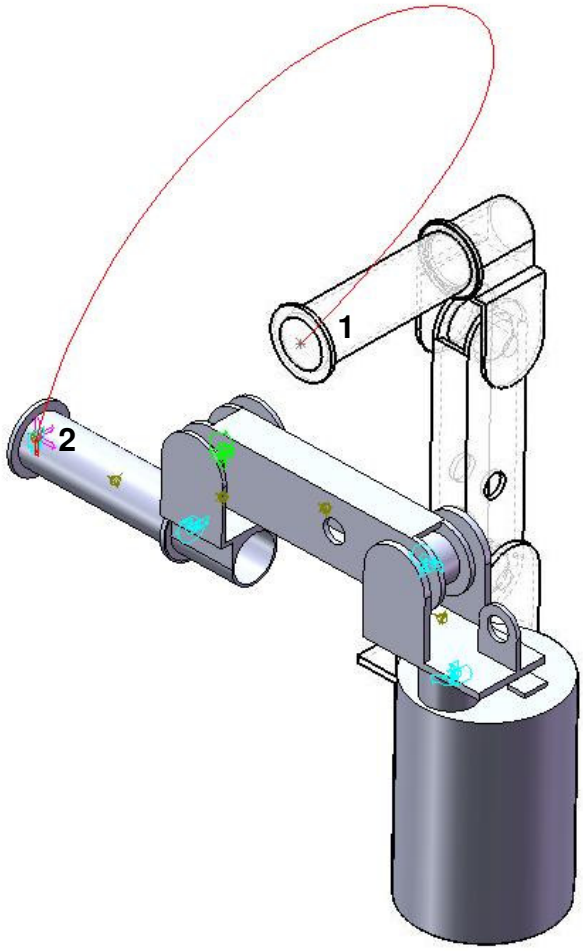


Figure 3.10 Initial and end position of the robot on a defined path for the kinematic analysis

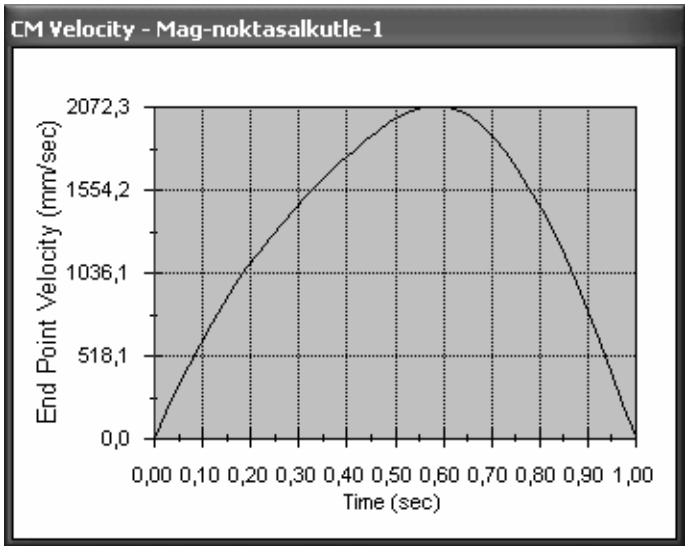
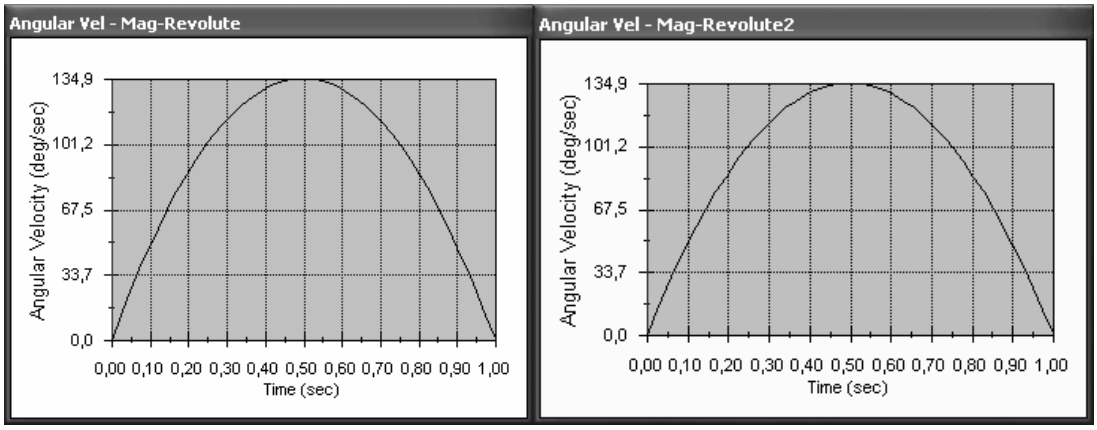
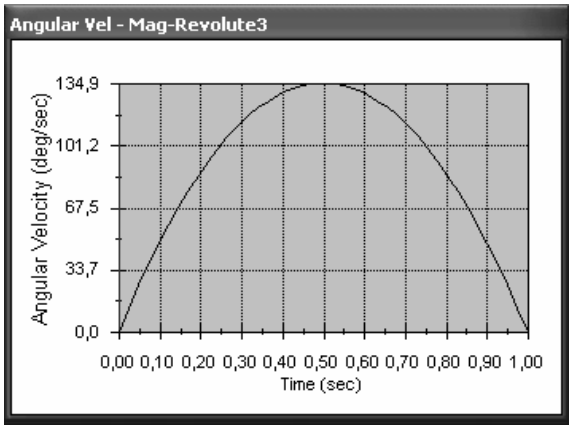


Figure 3.11 End point velocity of the robot



(a) (b)



(c)

Figure 3.12 Angular velocity of the robot axes, (a) first axis, (b) second axis, (c) third axis

3.1.4 Kinetic Analysis

Required motor torqueses are calculated for the movement of the robot. Angular velocity data which is obtained from kinematic analysis is used for kinetic calculations. Kinetic analysis is performed for the same trajectories by angular velocities are considered as input values. Axes torqueses required for desired motion are seen in Figure 3.13.

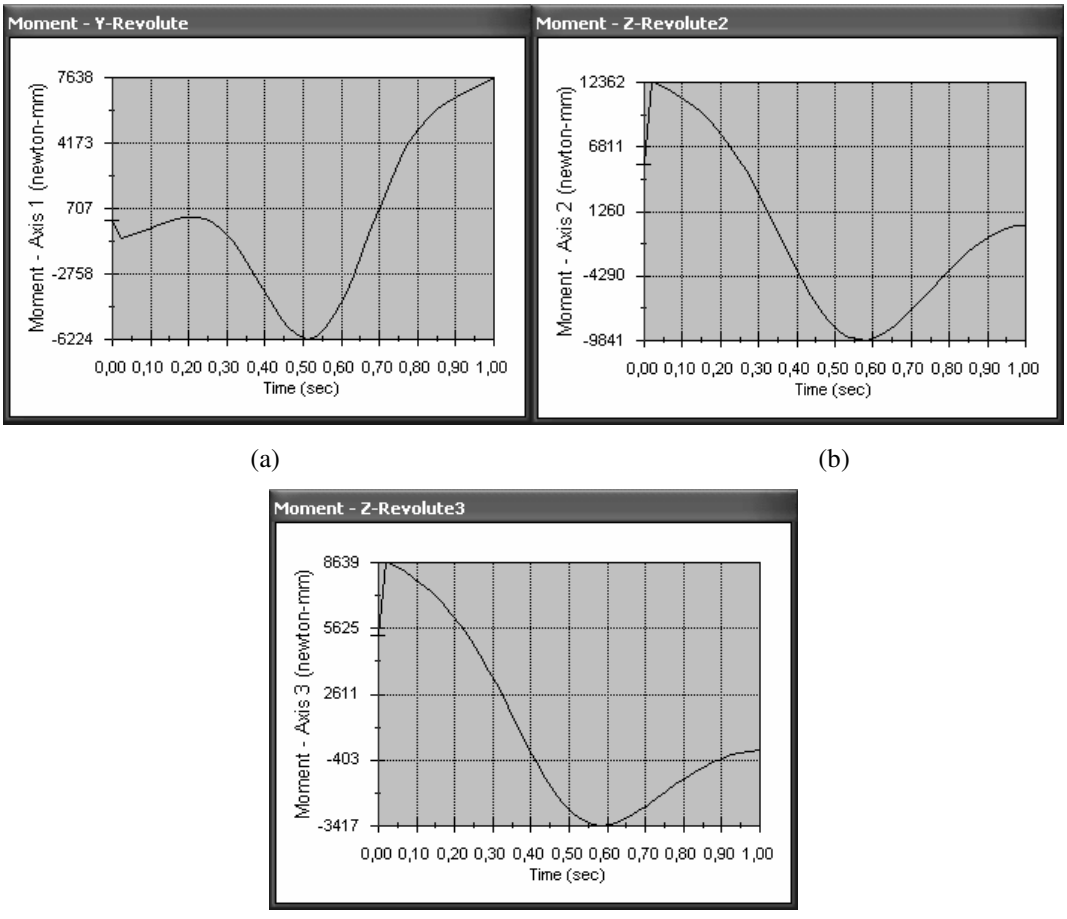


Figure 3.13 Required moments of the robot axes, (a)first axis, (b)second axis, (c)third axis

3.1.5 Static/Dynamic Strain and Frequency Analysis

In this step, finite element model of the robot model is obtained. For this aim, ABAQUS program is used since the finite element solutions of this program are very effective. The finite element model of the robot is seen in Figure 3.14.

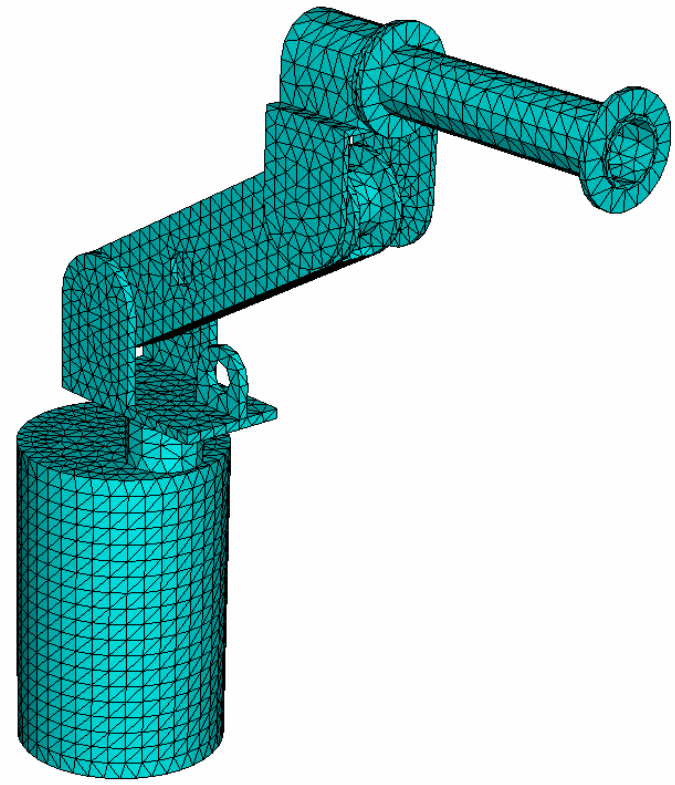
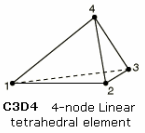
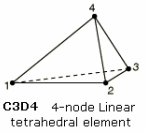
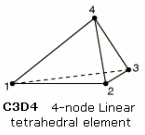
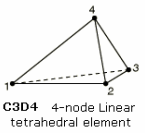
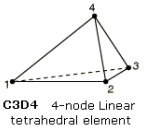


Figure 3.14 Mesh model of the robot by using ABAQUS program

Gravity effects are taken into consideration and the end-point load value is taken as 20 N in the downward direction. The point masses which are equivalent to masses of the associated motors and gears are added on the robot model. Four-node linear tetrahedral elements (C3D4) (ABAQUS[1] 6.5, 2004) are used in the mesh model of the robot parts. Mesh properties of the robot parts are seen in Table 3.2.

Aluminium materials are assigned to the whole robot parts without except shafts and rolling element bearing tabs due to lightness. Axes are locked in the static analysis. Motor breaks hold position is similar situation of the static analysis. Static analysis can be repeated for different axes angles by using parametric models of the robot. So stress distribution on the parts of the robot is investigated for different configurations. The static deflection is obtained as 198 µm from the first static analysis in case of the maximum reach distance position of the robot. After changing the parametric dimensions of the robot model, this deflection value is decreased to 77.5 µm. The effect of the parametric dimension p3t1 on the end point deflection is seen in Figure 3.15.

Table 3.2 Mesh properties of the robot parts

	Element Type	Element Size	Element Number	Node Number
P0 		0.01 m	16285	5041
P1 		0.01 m	4339	1284
P2 		0.01 m	4931	1710
P3 		0.01 m	3268	1088
P4 		0.01 m	3316	1101

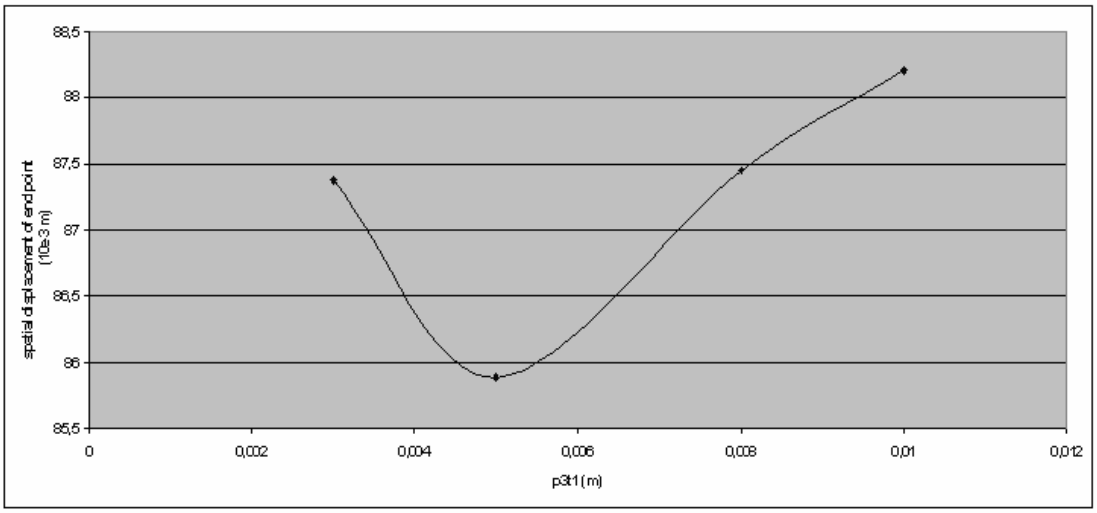


Figure 3.15 The effect of parametric dimension p3t1on the end point deflection

As seen from the figure that increasing the thickness of the part P4 decreases the static displacement of the end point until a specific value. Beyond this value, greater thickness values of P4 increases the displacement of the end point due to the increasing weight. This study is performed for the many parameters on the robot

parts and the end point displacement is decreased. Static displacement distribution for the maximum reach distance position of the robot is seen in Figure 3.16. Figure 3.17 shows the stress distribution on the robot manipulator for the same position.

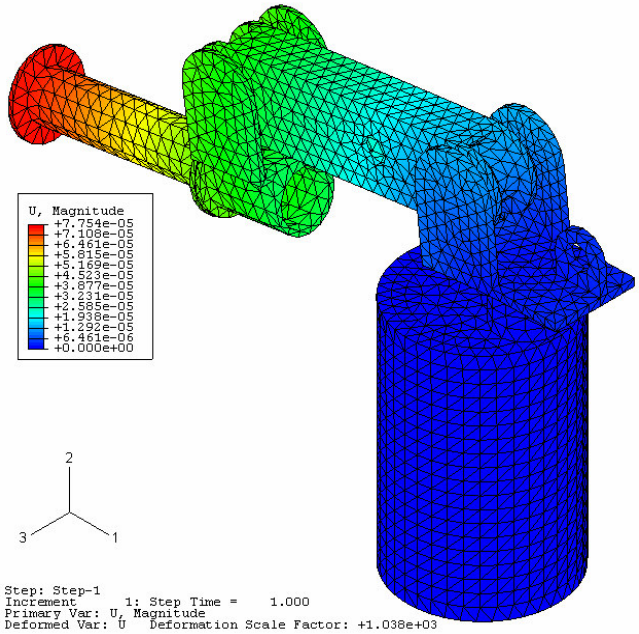


Figure 3.16 Displacement results obtained from the final static analysis for the maximum reach distance position

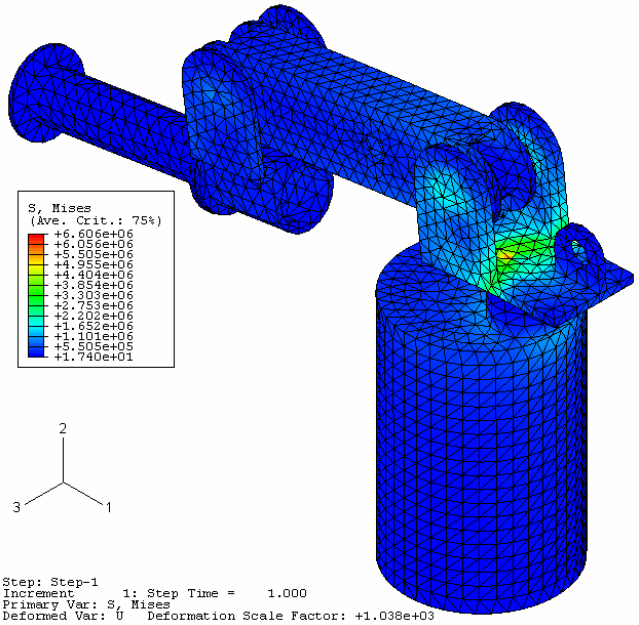


Figure 3.17 Stress results obtained from the final static analysis for the maximum reach distance position

The main reason for selection the ABAQUS program is its real time simulation capability of the mechanisms. The effect of the inertia forces on the strain and stress values of the parts can be seen. In the real time dynamic analyses, the axis movements are set to free. The end point of robot tracks the path with given movement input. End point location information must not be entered as movement input. It is an important point because displacement occurs during the analysis and joint angles change in order to compensate this displacement value. So the deflections aren't seen at the end point of the robot during the movement. The axis angular moments must be given in order to perform a correct analysis. Figure 3.18 shows the eight increments of the analysis.

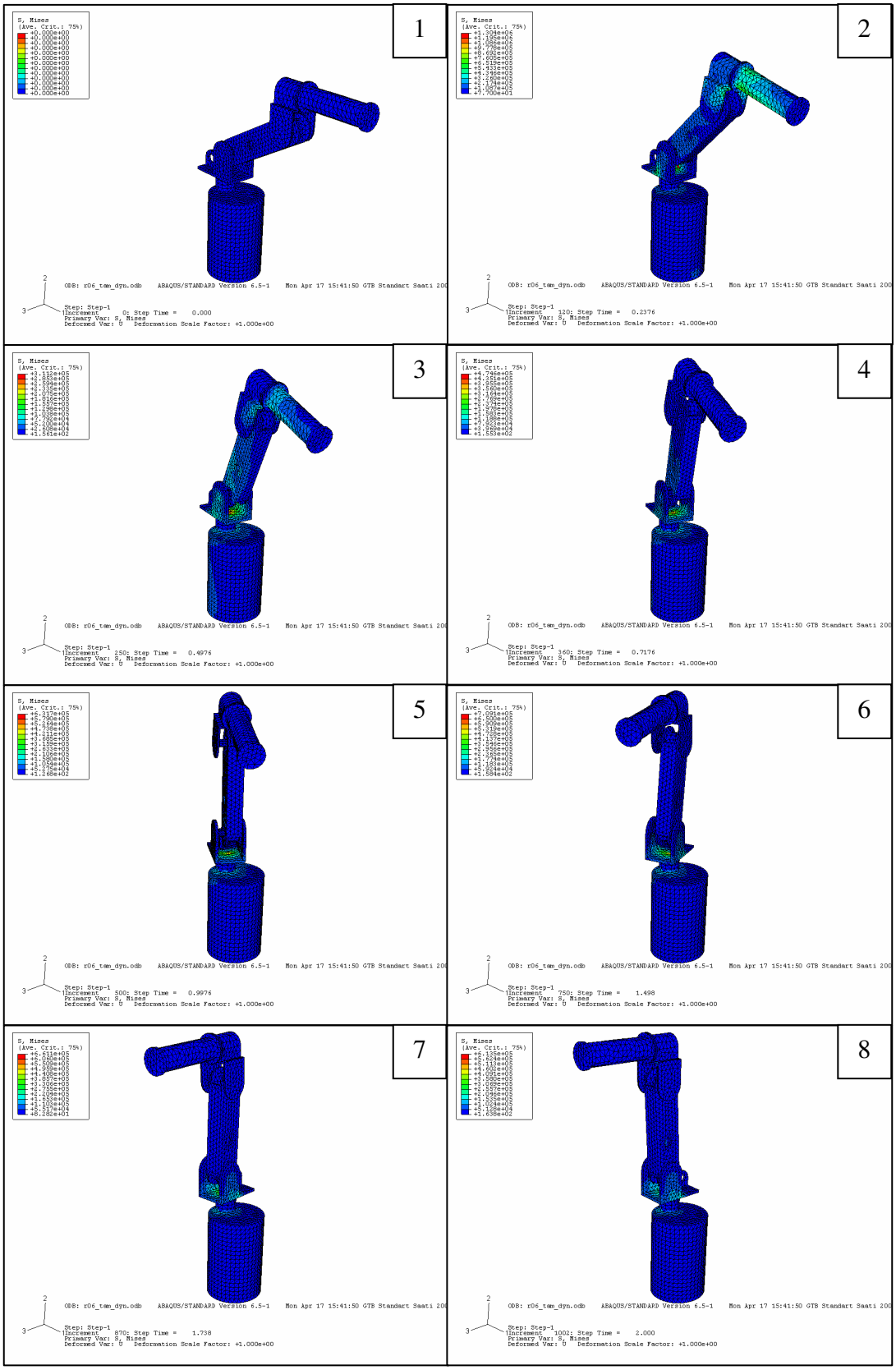


Figure 3.18 Dynamic analysis results of the robot for eight instant of the movement

The natural frequencies of a robot manipulator give information about the manipulator's rigidity. Natural frequencies of the robot are calculated for different positions by using ABAQUS program. Mode shapes and related natural frequency values of the first three modes for the maximum reach distance position of the robot are seen in Figure 3.19.

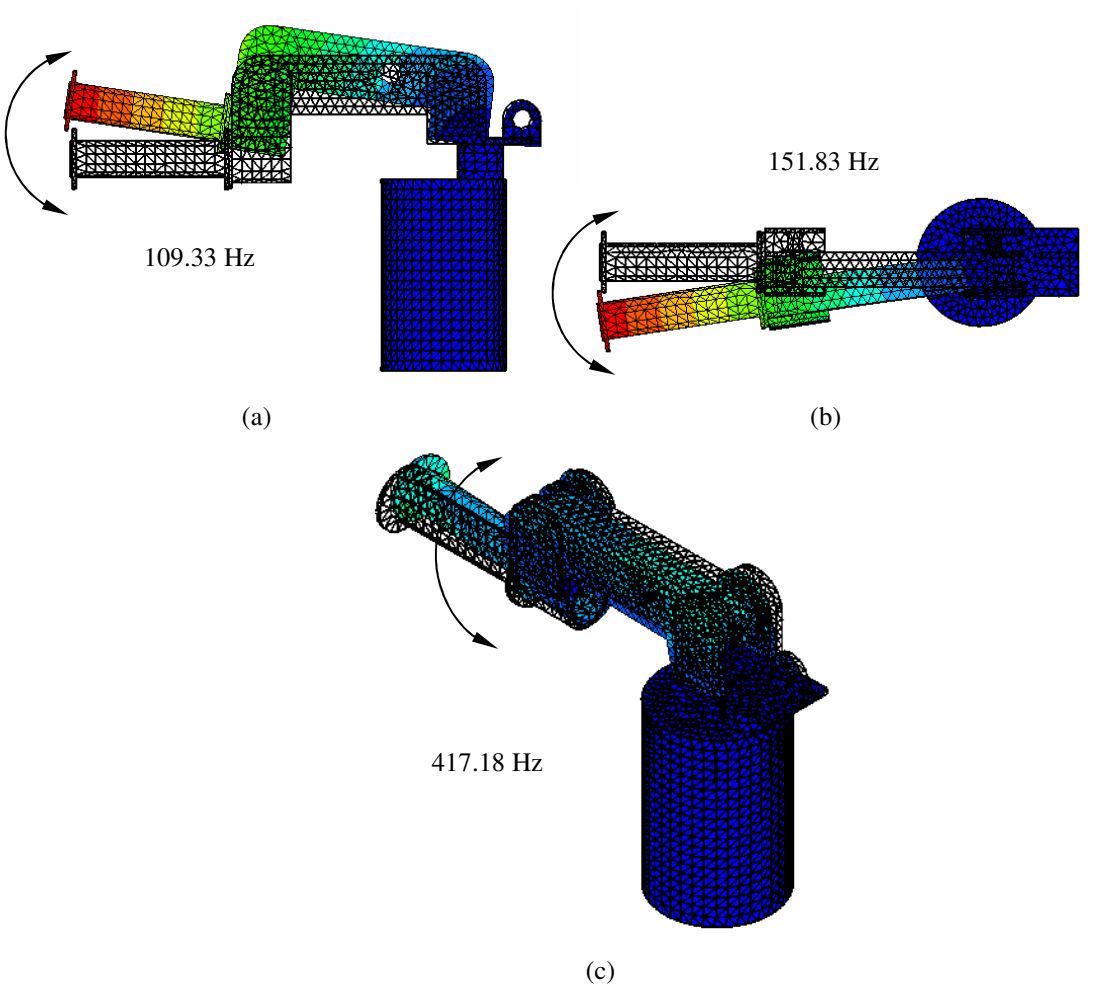


Figure 3.19 Natural frequencies and mode shapes of the robot a) first, b) second, c) third.

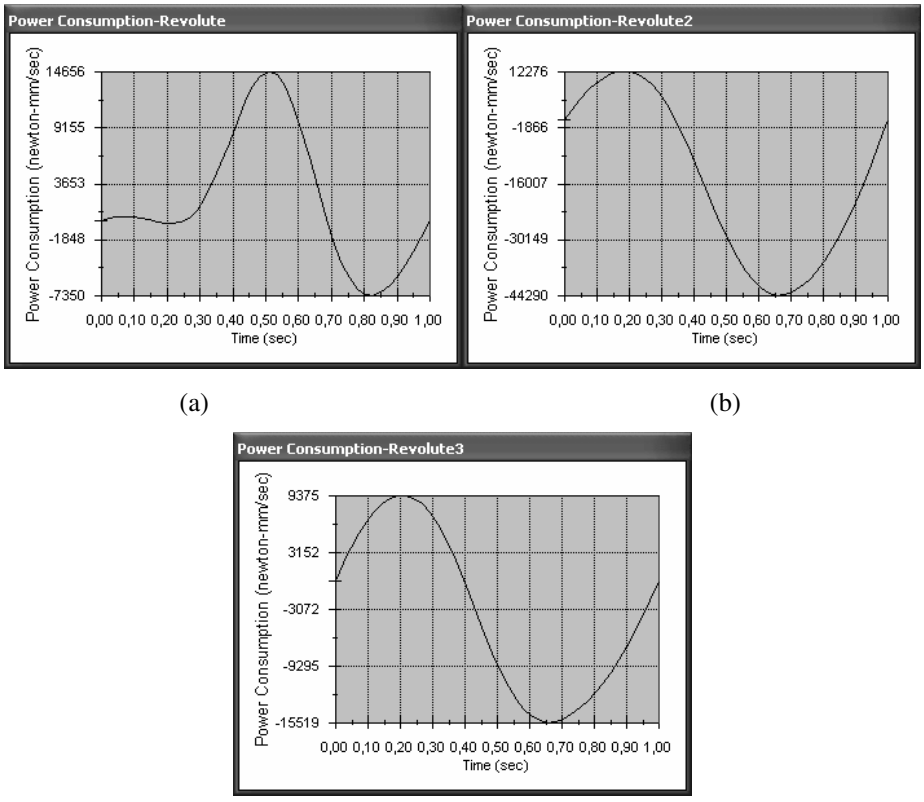
3.1.6 Selection of the Actuator Components

Gear ratios can be calculated by using the maximum angular velocities of the axes which are obtained from the kinematic analyses. Maximum angular velocity at maximum power of the selected servo motor should be known for doing this calculation. It is shown form the Omron product catalogue (Omron, 2006) that the speed value is 3000 rpm for the motors with suitable torque ranges. Maximum

angular velocity of robot axis is calculated as 135 deg/s (22.5 rpm) by the kinematic analyses. Maximum ratio for the selected gear is calculated as 133 by Equation (3.13).

$$\text{Maximum gear reduction ratio} = \frac{\text{Angular velocity of motor on maximum power}}{\text{Maximum angular velocity obtained from kinematic analysis}} \tag{3.13}$$

A gear couple which has a ratio under 133 satisfies our torque requirement. However required torqueses which are calculated by the kinetic analyses are important for this selection. For that reason, first of all motors should be selected. Power consumptions for the axes of the robot are seen in Figure 3.20. This calculation is performed done for a trajectory which is shown in the kinematic analysis section. The greatest power consumption is calculated for the second joint as seen in the figure.



(a) (b)

(c)

Figure 3.20 Power consumptions of the robot axis for the trajectory given in Figure 3.10 a) first, b) second and c) third axis

Selected motors for the axes are listed in Table 3.3. They are selected based on the power consumptions for different trajectories. Torque profiles of the motors are seen in Figure 3.21.

Table 3.3 Selected motors for the robot axes

Axis number	**Motor power**
1. Axis	30 W Omron AC servo motor
2. Axis	100 W Omron AC servo motor
3. Axis	100 W Omron AC servo motor

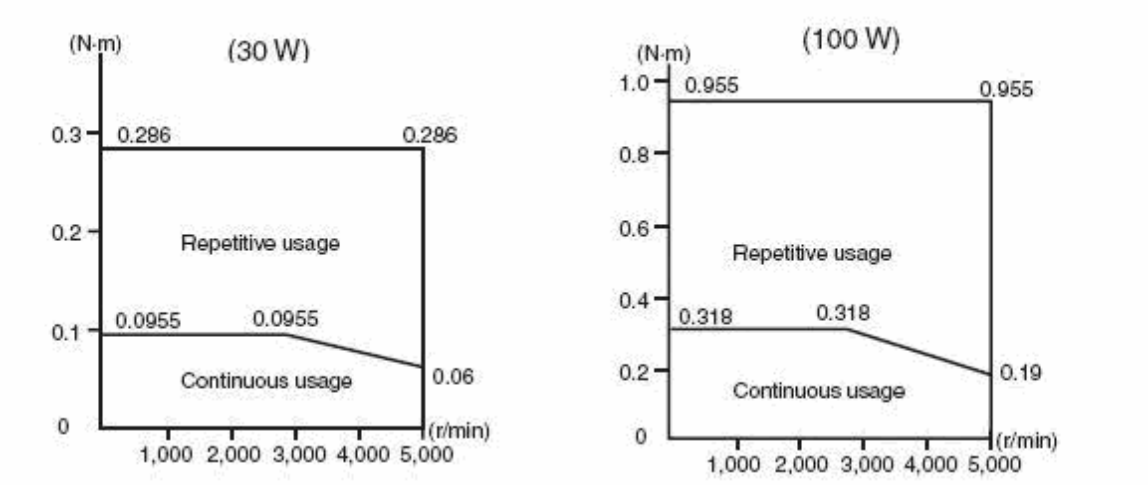


Figure 3.21 Motor torque graphics for selected motors (Omron, 2006)

In Figure 3.21, continuous usage motor torques should be taken as the motor torque. Equation (3.14) is used for calculating minimum gear ratio.

$$\text{Minimum gear reduction ratio} = \frac{\text{Required axis moment obtained from kinetic analysis}}{\text{Continuous usage motor moment}} \qquad (3.14)$$

Minimum gear reduction ratios are calculated for three axes of the robot as follows,

$$\text{Min gear reduction ratio} = \frac{7\ \text{Nm}}{0.0955\ \text{Nm}} = 73.29 \approx 74 \qquad \text{for Axis 1}$$

$$\text{Min gear reduction ratio} = \frac{13\ \text{Nm}}{0.318\ \text{Nm}} = 40.88 \approx 41 \qquad \text{for Axis 2}$$

$$\text{Min gear reduction ratio} = \frac{13\ \text{Nm}}{0.318\ \text{Nm}} = 40.88 \approx 41 \qquad \text{for Axis 3}$$

We have Maxon planetary gears which has the reduction ratio 74. These gears are used because both reduction ratios are suitable with our calculations and they have very compact structure. Details of the actuator selection are shown in Table 3.4

Table 3.4 Details of actuators selection

		Axis 1	Axis 2	Axis 3
MOTOR	Motor	Omron 30W AC servo motor	Omron 100W AC servo motor	Omron 100W AC servo motor
	Max continuous moment (Nm)	0.0955	0.318	0.318
	Max. repetitive moment(Nm)	0.296	0.955	0.955
	Max. angular velocity at max. power (rpm)	3000	3000	3000
GEAR	Gear	Maxon Planetary gear	Maxon Planetary gear	Maxon Planetary gear
	Reduction Ratio	1/74	1/74	1/74
AXES	Max angular velocity (rpm) [deg/s]	40.54 [243.24]	40.54 [243.24]	40.54 [243.24]
	Max continuous moment (Nm)	7.067	23.532	23.532
	Max repetitive moment (Nm)	21.904	70.67	70.67

3.1.7 Evaluation and Optimization

Before the manufacturing stage, modifications are done on the robot parts successively and the effects of the modifications are examined. Evaluation and optimization process are completed when the desired values have been obtained.

3.1.8 Manufacturing

In the manufacturing process, tolerances between adjacent parts and homocentricity between the axes of the robot are very important. For these reasons especially robot main parts are manufactured by using CNC machines. P1 and P3 parts are manufactured by using 5-axis CNC machine with single fixation. Parts connected to each other by dowel pin. Manufactured robot is seen in Figure 3.22.

Figure 3.22 Manufactured robot assembly

3.2 .Macro Positioning SCARA Manipulator (DEU S45-900)

3.2.1 Job Definition

In this section, a robot is designed which performs the macro positioning for the hexapod which is designed and developed in the scope of a project. In this study, firstly the SolidWorks and CosmosWorks softwares are used for modelling and analyzing. A two-axis macro-positioning robot with high payload capacity and long reach distance is preferred. First model of the robot is designed and analyzed. When designing this robot, the hexapod is considered because the purpose of this design is macro positioning of the hexapod. Payload of the robot (45 kg) is calculated while considering the weight of the hexapod (15 kg) and possible 3^{rd} and 4^{th} axes (approx. 30 kg). In addition, dimensions of the robot are calculated while considering the workspace of the hexapod operations. Approximately 900 mm reach distance is desired for the hexapod workspace.

3.2.2 Figural and Parametric Design

A basic model was constituted for the two-axis SCARA robot which has relatively a simple geometry. It is used for testing different positions of the robot and finding the characteristics of actuators. Figure 3.23 shows 3D view of this basic solid model of the SCARA robot.

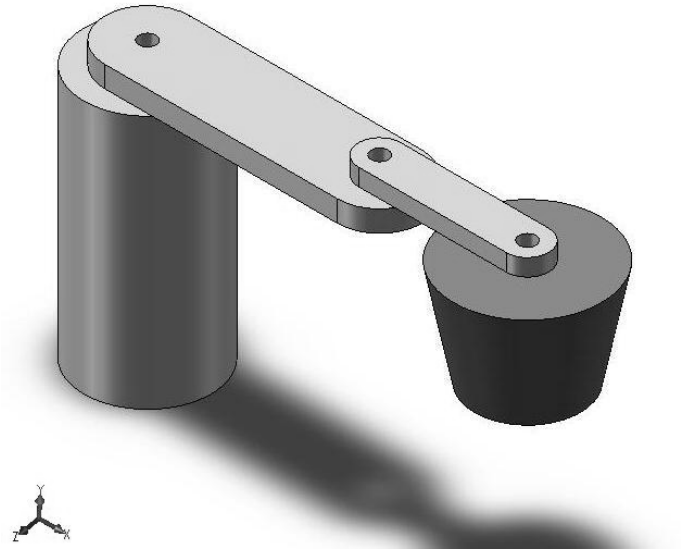


Figure 3.23 Parametric model of the SCARA robot

The basic model consists of a cylindrical body, a long arm, a short arm and a basic hexapod model. The dimensions of these parts are calculated considering the required workspace for the macro-positioning of the hexapod. The main goal of this initial model is to select the optimal actuators for the robot.

3.2.3 Selection of Actuators

For the parametric model of the macro-positioning robot, simulations and inverse kinematic analyses are made in CosmosMotion software to find required actuator torques. In a demo simulation of the basic model, 200 mm displacement both in x and z directions for first 5 seconds and than same displacement in opposite directions for last 5 seconds are given to the hexapod. At the end of this simulation, the axes return to their initial positions. Rotations and speed of the actuators for this demo motion are calculated by considering the macro positioning of the hexapod. Required moments for actuators are calculated by using inverse kinematic method in CosmosMotion software. Figure 3.24 shows the required moments of first and second axes for the demo motion.

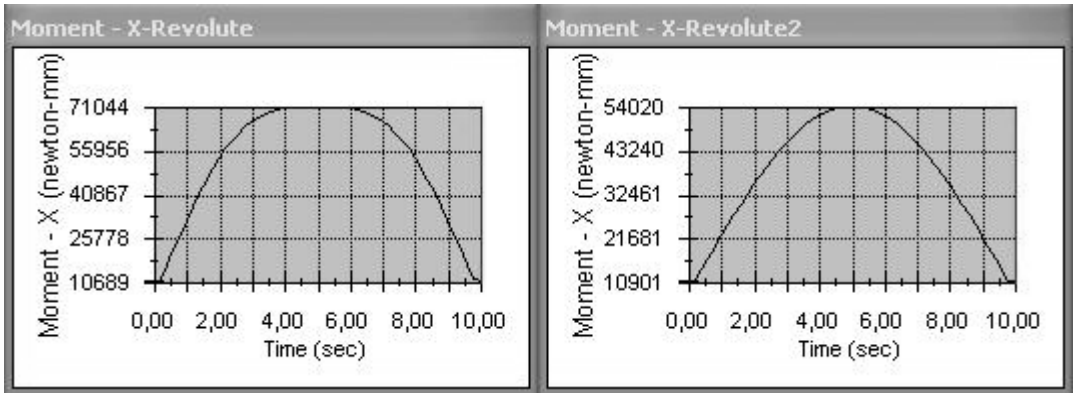


Figure 3.24 Required moments of first and second axis for the demo motion.

High motor torqueses are required due to the high inertia of the system and 45 kg payload with 900 mm reach distance creates high static moments. In addition, macro positioning of the hexapod requires high precision and repeatability. Actuators and bearings of the robot should guarantee all these requirements.

Having searched and examined various model and types of AC servo actuators, it is decided to use CHA series hollow shaft AC servo actuators of Harmonic Drive AG with HIPERFACE encoders. CHA-32A-100 type of actuator is chosen for first axis and CHA-20A-100 type is chosen for second axis of the robot. Both actuators come with integrated Harmonic Drive zero backlash precision gear set (100:1 ratio). Figure 3.25 shows real and section views of Harmonic Drive CHA-20A-100 type actuator and Figure 3.26 shows performance curves of both actuators.

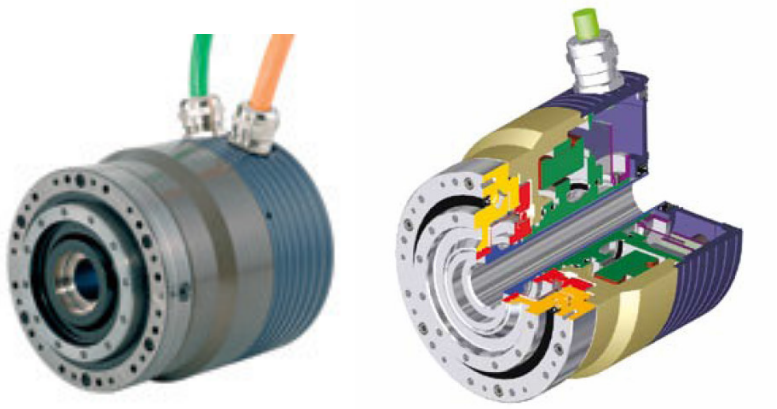


Figure 3.25 Real and section views of a Harmonic Drive servo actuator (CHA-20A-100)(Harmonic Drive, 2006).

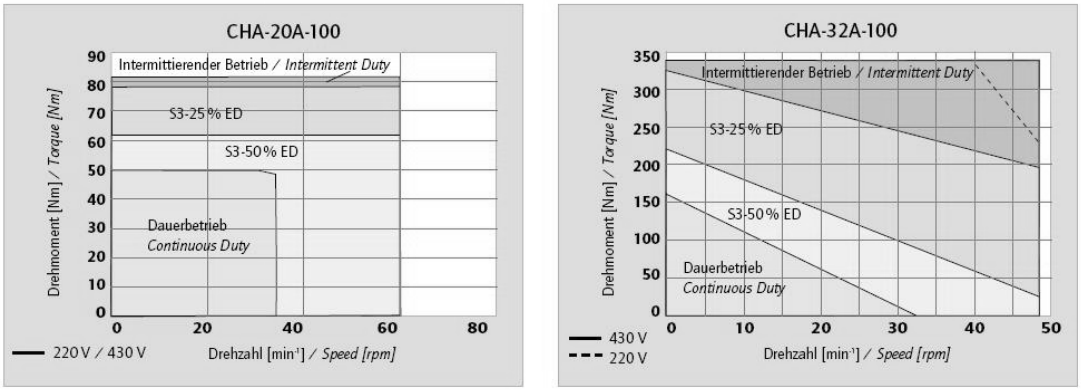


Figure 3.26 Performance curves of the CHA series servo actuators (Harmonic Drive, 2006).

The Harmonic Drive CHA series digital AC hollow shaft actuators with precision Harmonic Drive gearing set have many advantages for industrial servo systems and robotic applications. They provide precision motion control and high torque capacity in a very compact package. In addition, they come with integrated high load capacity bearings. For many applications there is no need to use additional bearings because of these specially developed bearings. For the bearing of CHA-32A-100, permissible static tilting moment is 2425 Nm and for CHA-20A-100 it is 578 Nm. More detailed technical data about the CHA series actuators are shown in Table 3.5.

Other important advantages of the Harmonic Drive CHA series hollow shaft AC servo actuators are as followings (Harmonic Drive, 2006):

- Radial and axial run out < 10 μm thanks to integrated precision bearing,
- Improved smoothness at low speeds thanks to high pole numbers of the motors,
- Higher mechanical resonance frequencies thanks to higher torsional stiffness,
- No additional support bearing necessary thanks to high load capacity and high stiffness of output bearing of the actuators,
- Higher gear precision thanks to improved smoothness and surface quality,
- Zero backlash and <6 arcsec repeatability of gear component set.

Table 3.5 Detailed technical data about the CHA series actuators

Actuator	**Unit**	**CHA-20A**	**CHA-32A**
Ratio		100	100
Maximum output torque	Nm	82	333
Maximum output speed	min^{-1}/rpm	60	48
Continuous stall torque	Nm	49	150
Maximum current	Arms	2.8	7.1
Continuous stall current	Arms	1.6	3.2
No load starting current	Arms	0.14	0.20
No load current constant (30 °C)	10^{-3}A/rpm	7.7	21.1
No load current constant (80 °C)	10^{-3}A/rpm	3.4	6.8
Demagnetization current	Arms	7.0	15
Torque constant (at output)	Nm/A	33.4	49.7
Torque constant (Motor)	Nm/A	0.36	0.57
AC-voltage constant (L-L, 20 °C)	Vrms/1000rpm	23	37
Motor terminal voltage (fundamental wave only)	Vrms	220-430	220-430
Mechanical time constant (20 °C)	ms	6.7	7.1
Actuator	**Unit**	**CHA-20A**	**CHA-32A**
Electrical time constant (20 °C)	ms	1.4	1.6
Moment of inertia without brake (at output)	kgm^2	1.12	4.9
Moment of inertia with brake (at output)	kgm^2	1.39	5.88
Moment of inertia at motor (with brake)	$kgm^2 \times 10^{-4}$	1.12 (1.39)	4.90 (5.88)
Motor moment of inertia without WG (with brake)	$kgm^2 \times 10^{-4}$	0.94 (1.21)	3.2 (4.16)
Maximum motor speed	min^{-1}/rpm	6000	4800
Rated motor speed	min^{-1}/rpm	4500	3500
Resistance (L-L, 20 °C)	Ω	5.9	3.7
Inductance (L-L)	mH	8.0	6.0
Number of pole pairs		5	6
Brake voltage	VDC	24 ± 10 %	24 ± 10 %
Brake holding torque	Nm	82	180
Brake current to open	A	0.6	0.9
Brake current to hold	A	0.25 (10V)	0.4 (10V)
Number of brake cycles at n=0 rpm		100000	100000
Emergency brake cycles		200	200
Weight without brake	kg	3.2	6.6
Weight with brake	kg	3.9	7.8

3.2.4 Detailed Design of the First Model

Firstly a cylindrical body part is designed. After that, long arm (550 mm length between joint axes) and short arm (350 mm length between joint axes) are designed. Than a housing and bearing are designed in order to mount the first AC servo motor (CHA-100L-32A) to the body. However, the main purpose of the bearing design is not just mounting of the motor; it is especially for eliminating of forces and moments which are transmitted from robot arms. Moreover, a special shaft is designed for transmission of torque from the motor to the arm. Same type of a bearing and shaft are designed also for second motor (CHA-100L-20A). The second motor and its bearing are mounted to the long arm instead of the body. Figure 3.27 shows 3D model assembly and section view of the first robot model.

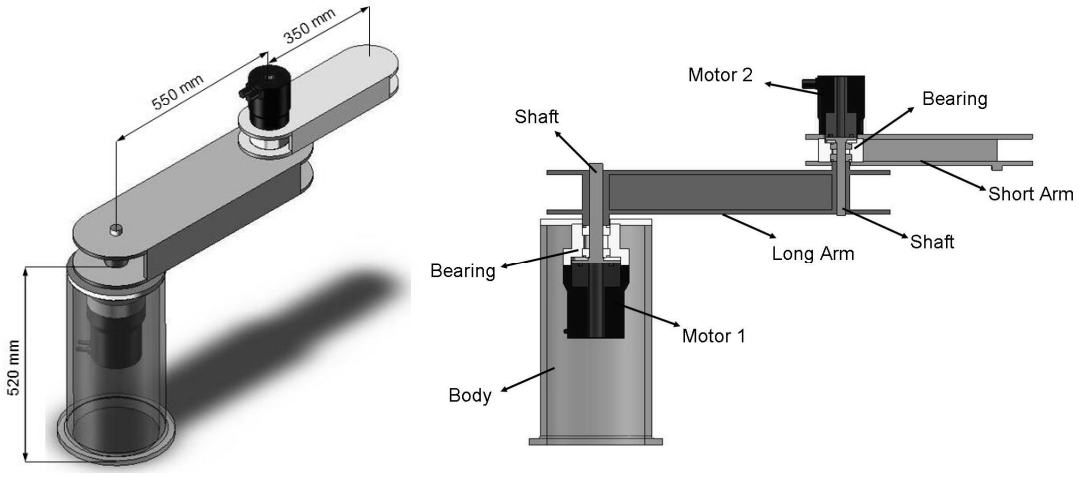


Figure 3.27 Solid model and section view of the first model.

3.2.5 Static and Frequency Analyses

The first model of the SCARA robot is analyzed for maximum distance from first axis. All the finite element model analyses are done in Cosmos Works software and 450 N normal forces is applied to the short arm to demonstrate real working conditions (45 kg payload; considering weight of the hexapod and possible weight of the 3rd and 4th axes) of the robot.

With FEM analyses; maximum displacement, maximum stress for Von Mises, equivalent strain and minimum natural frequency of the first model were analyzed. Mesh properties for these FEM analyses are given as follow; mesh type: solid mesh, element size: 17.230 mm, total nodes: 65223, total elements: 35.260. Figure 3.28 shows the maximum displacements of the SCARA robot as the results of these FEM analyses. Table 3.6 shows results of FEM analyses for maximum reach position.

FEM analyses results of the first robot model point out that the design should be improved. The lowest natural frequency of the model (30.629 Hz) is sufficient for our design goal. However, the maximum displacement at the tip (2.950 mm) is quite large. The macro-positioning robot is supposed to have precise movement capability. Therefore the maximum displacement value at the tip should be lower than 0.4 mm which is determined pursuant to our design goal.

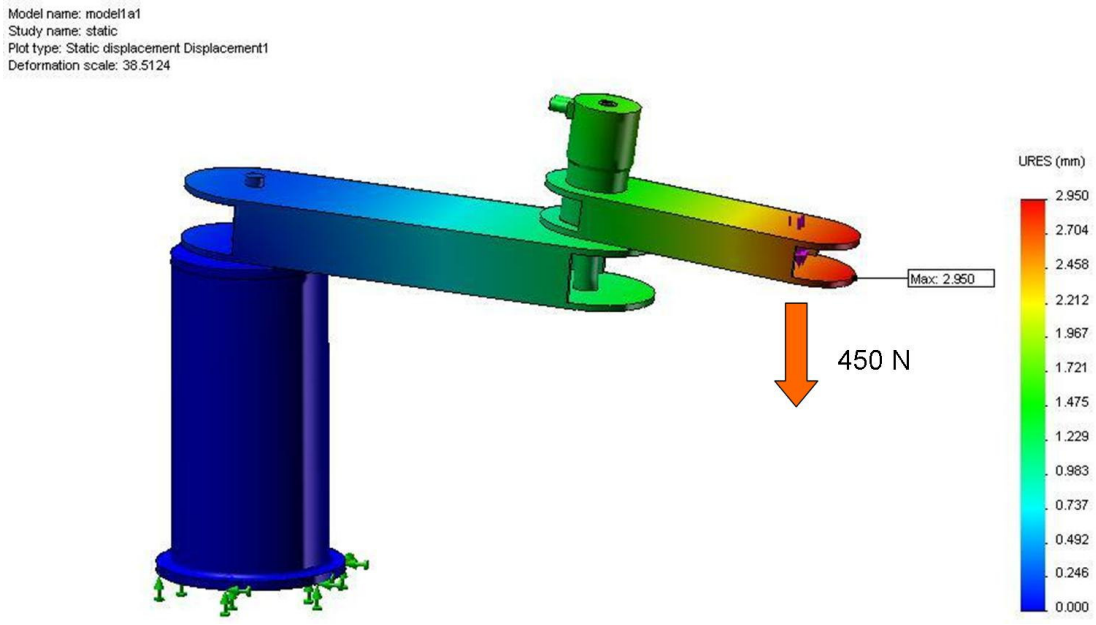


Figure 3.28 Maximum displacements of the first model for position M (deformation scale: 38.51).

Table 3.6 Results of FEM Analyses for the first model.

Static Analyses	
Displacement (max.)	2.950 mm
Stress (vonMises, max.)	71.984 MPa
Strain (Equivalent, max.)	0.00096 ESTRN
Natural Frequency Analyses	
1st mode (min.)	30.629 Hz
2nd mode	38.269 Hz
3rd mode	111.49 Hz

3.2.6 Evaluation and Final Model of SCARA Robot

First model of the robot cannot ensure the static analysis requirements. Therefore, the first model should be improved by making some modifications. If it cannot ensure the requirements even after these modifications, it should be completely redesigned. In this study, the first model is improved and analyzed for each improvement several times according to the evaluation/optimization processes of integrated analysis of design.

To improve analysis results of the macro-positioning robot some modifications are made on the first model. First of all, the maximum displacement of the end point with 45 kg payload should be decreased. After that, natural frequencies of the robot should be increased. Therefore, sections and thicknesses of the arms are changed. Figure 3.29 shows section views of the final model and its long arm.

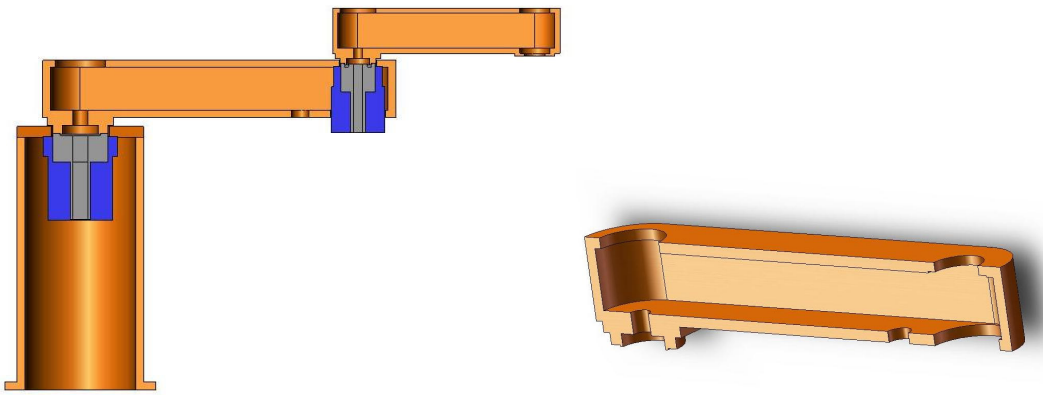


Figure 3.29 Section views of the final model and long arm part.

After analysing the first model of the robot, it can be seen that there is no need for the extra joint bearings, because bearings of Harmonic Drive servo motors (CHA-100L-32A, CHA-100L-20A) are capable of carrying all loads and moments of the robot. Hence, the model is modified to remove these extra joint bearings and shaft parts. Special flange parts are designed and integrated to the arms for connecting the arms to the motor shafts directly. As a result of this improvement, the arms can be directly driven by the motor shafts. This connection type also increases the resistance and decreases the maximum displacements of the arms. As material of parts, steel is preferred instead of aluminium because of its higher resistance capability than aluminium.

The final solid model view of the macro-positioning robot with the hexapod and the console is shown in Figure 3.30.

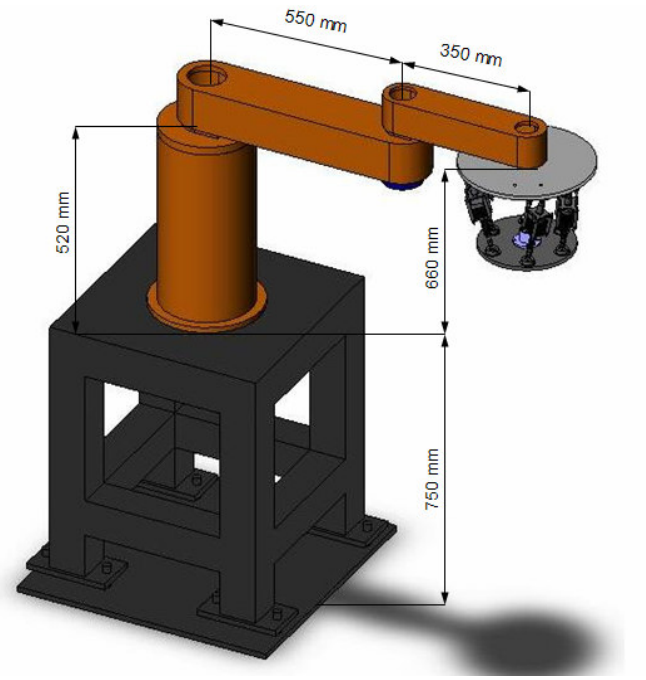


Figure 3.30 Final model of the robot with the hexapod

When all the modifications are finished, the final model of the robot is analysed for pre-defined positions which are position M (maximum reach distance) and position R (reference point, x=200mm, z=200 mm). All the FEM analyses are done

in CosmosWorks software. Figure 3.31 shows the maximum displacements with 450 N load to demonstrate 45 kg payload.

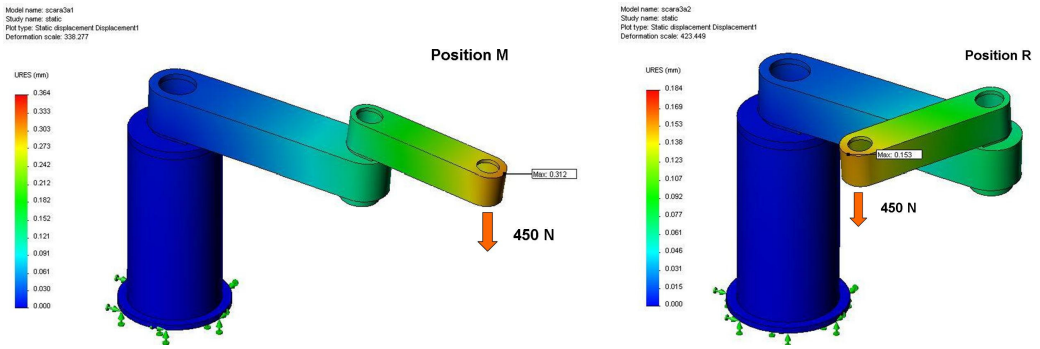


Figure 3.31 Displacements of the final model for position M and R (deformation scale: 338 and 423).

With FEM analyses; maximum displacement, maximum stress for Von Mises, equivalent strain and minimum natural frequency of the final robot model are calculated. Table 3.7 shows the results of analyses for position M and position R. Mesh properties for these FEM analyses are given as followings; mesh type: solid mesh, element size: 17 mm, total nodes: approx. 70.000, total elements: approx. 38.000.

Table 3.7 FEM Analyses results of the final model for position M and R.

	Position M	**Position R**
Static Analyses		
Displacement (max.)	0.312 mm	0.153 mm
Stress (vonMises, max.)	37.238 MPa	36.611 MPa
Strain (Equivalent, max.)	0.000090 ESTRN	0.000086 ESTRN
Natural Frequency Analyses		
1st mode (min.)	31.084 Hz	42.408 Hz
2nd mode	41.643 Hz	61.774 Hz
3rd mode	88.558 Hz	74.358 Hz

The modal analyses are performed in order to calculate the approximate structural rigidity of the manipulator. The influence of the joint flexibility on the exact modal behaviour of the complex systems such as robot manipulators are examined in Chapter 5.

3.2.7 Design of a Console for the SCARA Robot

Micro positioning operations by the hexapod are required to be done on a table which has approximately 750 mm height. The macro-positioning robot should be installed on a special console which has approximately the same height with this table. Moreover, the console should have highly rigid structure and high natural frequencies to reduce possible vibrations. After designing and analysing some console models, final model of the console is designed and manufactured. It is made by square section steel profiles which have 100x100 mm section dimensions and 5 mm thicknesses. Table 3.8 shows the results of natural frequency analyses of the final console model.

This console should be joined to the ground to increase natural frequency and working safety of the system. Therefore a base platform part is designed for the console. The base platform is joined to the ground and than the console is joined to it with adjustable screw-nut joints. Upper surface of the console is made parallel to the ground by adjusting the nuts. The console and the base platform are shown in Figure 3.32 with their main dimensions.

Table 3.8 Frequency analysis results of the console.

Natural Frequency Analysis	
1st mode (min.)	118.51 Hz
2nd mode	144.62 Hz
3rd mode	218.38 Hz

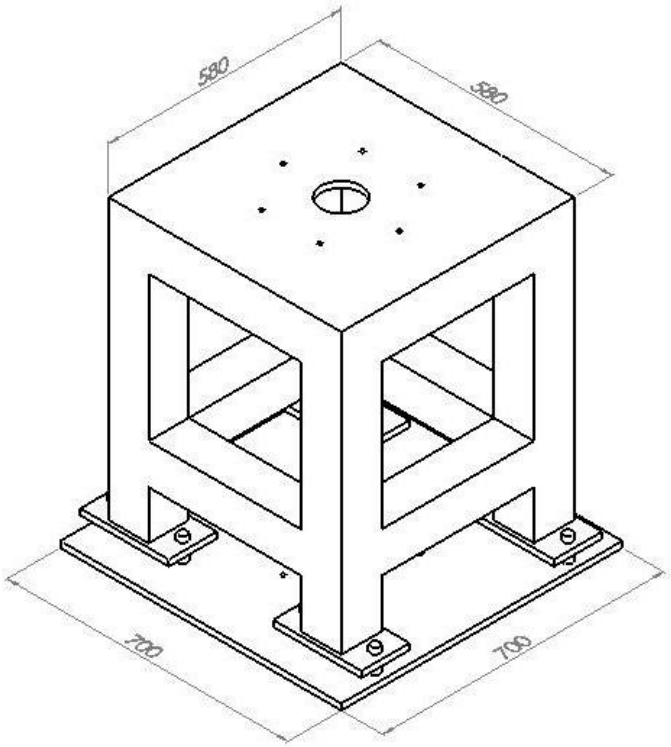


Figure 3.32 Console and the base platform with main dimensions (mm).

3.2.8 Analysing the Robot with the Console

After mounting the robot to the console, all the FEM analyses of the robot are repeated with the console due to the changing rigidity. The console may increase the maximum displacements, stresses, strains and may reduce the minimum natural frequency of the robot. Displacement results of the whole system for two different positions are seen in Figure 3.33. Results of these analyses are listed in Table 3.9.

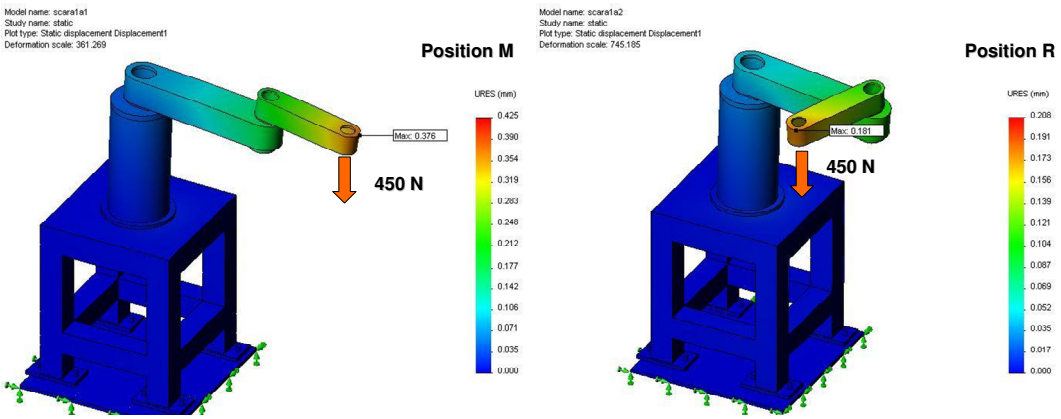


Figure 3.33 Displacements of the robot with the console for position M and R (scale: 361 and 745).

Table 3.9 Results of FEM analyses of the robot with the console.

	Position M	**Position R**
Static Analyses		
Displacement (max.)	0.376 mm	0.208 mm
Stress (vonMises, max.)	12.611 MPa	13.847 MPa
Strain (Equivalent, max.)	0.000046 ESTRN	0.000046 ESTRN
Natural Frequency Analyses		
1st mode (min.)	28.321 Hz	37.876 Hz
2nd mode	32.058 Hz	41.325 Hz
3rd mode	54.085 Hz	50.558 Hz

3.2.9 Manufacturing of the Robot

When all the simulations and analyses of the robot are finished, 2D drawings are generated. All parts are made from alloy steel material. Alloy steel is preferred because of its high resistance and easiness for manufacturing a part.

Macro-positioning robot consists of a body, a bonnet, a long arm, a short arm, a connection part for hexapod and AC servo motors. To carry the robot, a console is used. In addition that, to join this console to the ground a base platform is used. All these parts are manufactured, but AC servo motors (Harmonic Drive CHA-32A-100 and CHA-20A-100) are purchased.

The body part is manufactured by means of cutting, milling, lathing, welding and drilling operations. It consists of two sub parts. First one is upper part of the body and second one is lower part of the body. To provide required dimensions which are given in 2D technical drawings, these sub parts are cut, milled and lathed. After that, these two sub parts are joined by welding operation. To prevent from dimensional deviations, screw holes of the body are drilled after this welding operation because welding operation creates residual stresses and these stresses could change the dimensions of parts.

Connection part for hexapod and bonnet part are manufactured by milling, lathing and drilling operations. They consist of no sub parts and they both made from just one piece of steel part. To prevent from dimensional deviations, lathing reference surface and drilling screw holes operations should be made at the same CNC machine. They are milled also to provide required surface qualities.

The long and the short arm parts are manufactured by same operations and they have same sub part types. These operations are cutting, milling, lathing, drilling and welding. The both arms consist of 7 sub parts. Those are a front, a back, a left, a right, an upper, a lower sided parts and a flange part. The front and the back side parts are made by cutting a cylindrical part from the middle of it. The left and right side parts are made by cutting a steel plate. After that, they are milled for welding operation. The upper and lower side parts are also made by cutting a steel plate. The flange part is made from a single steel part by milling and lathing. All these sub parts are joined by welding operations. Lathing and milling reference surfaces and drilling screw holes operations are made after the welding operations.

The console part is manufactured by cutting, welding, drilling and milling operations. It consists of 12 pieces of square shaped structural steel profiles, an upper plate and 4 lower plates. The upper plate and lower plates are made by cutting a steel plate. All these sub parts are joined by welding operation to form the console. Joint holes of the plates are drilled and surface of the upper plate is milled after the welding.

The base platform part is manufactured by cutting and drilling operations. It is made by firstly cutting a steel plate. Than screw holes are drilled. After all parts are manufactured, they are polished and painted.

Assembly of the robot is made by as the following order: Firstly, the base platform is fixed on the ground and the console is fixed on it. The upper surface of the console must be parallel to the ground. If it is not, the height of it should be adjusted by screwing the nuts of the base platform. Secondly, the body part is joined

to the console by screwing. After that, body of the first AC servo motor (CHA-32A-100) is joined to the bonnet and this assembly is joined to the body by screwing. Than the body of the second AC servo motor (CHA-20A-100) is joined to the long arm and the flange of the long arm is joined to the shaft of the first motor. Flange of the short arm is joined to the shaft of the second motor by screwing. Hexapod is joined to the short arm via a connection part. After finishing the assembly of parts, cable installations are done. Assembled SCARA robot and console system is seen in Figure 3.34.

Figure 3.34 Assembled SCARA robot and console system

3.3 Six Axis Serial Manipulator (DEU-6X5-1500)

In this application an industrial serial robot having 6 degree of freedom is designed. In this new robot manipulator, maximum pay load and reach distance are chosen as close as possible to the ABB IRB 1400 robot. Maximum payload of the IRB 1400 is 5 kg and maximum reach distance is 1440 mm. In the light of these design parameters, the new design is formed. Maximum payload and maximum reach distance was aimed to be 5 kg and 1500 mm. In our design The SolidWorks and CosmosWorks programs are used throughout the design process. The first design of the six-axis robot is seen in Figure 3.35. In this design, there are not details and approximate masses of motors are taken into consideration. Maximum payload is attached to the robot end point in this design stage. This model is used in kinematic, kinetic, static and frequency analyses.

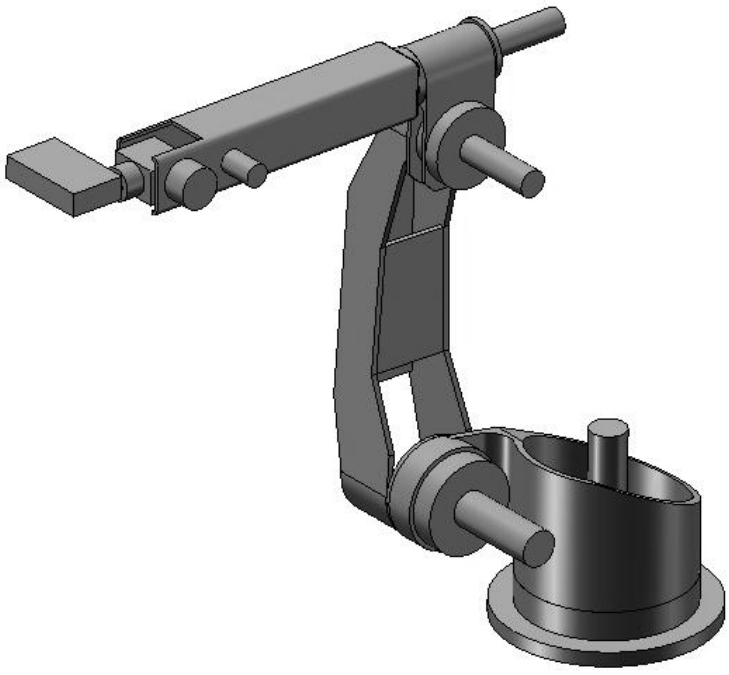


Figure 3.35 First model of the six-axis robot (DEU-6X5-1500)

Workspace of the robot is calculated from kinematic analysis according to the dimensions and the geometry of the parts. The workspace of the robot is seen in

Figure 3.36. Some of trajectories are defined to pass through the maximum reach distance for the kinematic analysis. A sample trajectory is seen in Figure 3.37.

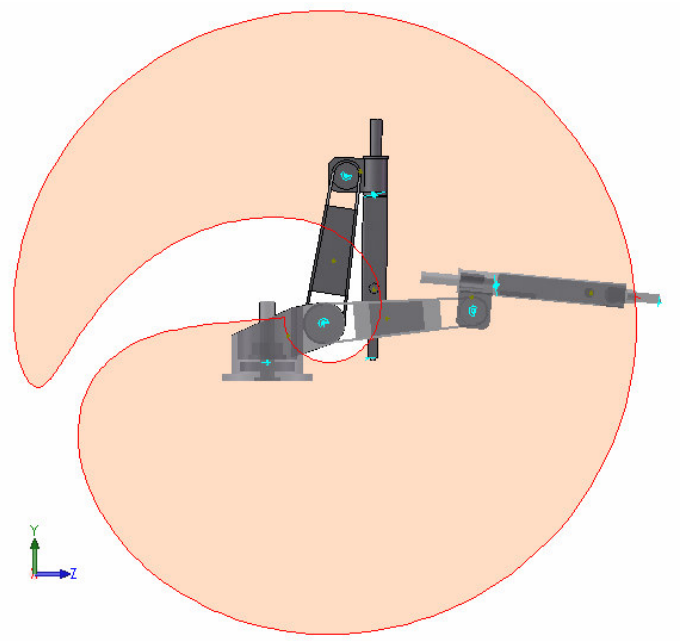


Figure 3.36 Workspace of the DEU-6X5-1500

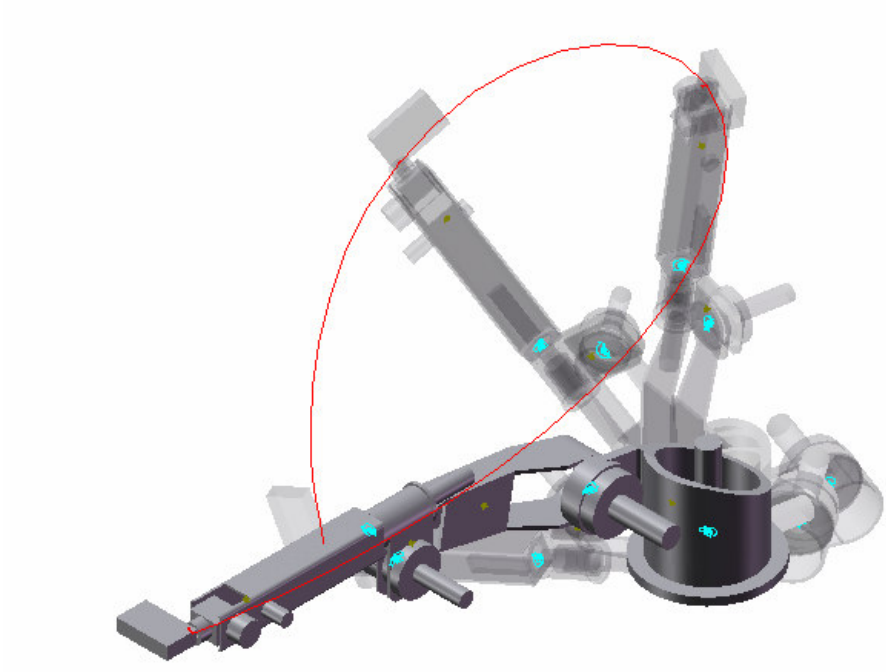


Figure 3.37 A sample trajectory which passes through the maximum reach distance of the robot

Maximum velocity of the end point is desired to be 3000 mm/s. Movement time of the robot is decreased as much as the end point velocity of the robot is 3000 mm/s in the kinematic analysis. The end point velocity profile of the robot for this trajectory is seen in Figure 3.38.

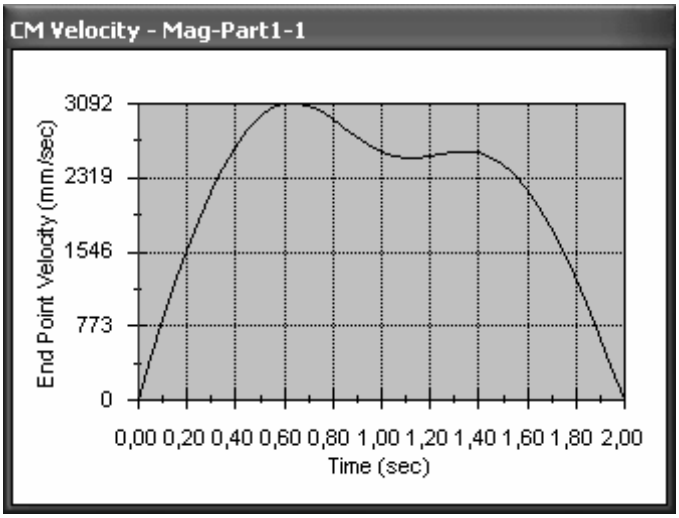


Figure 3.38 End point velocity of the robot for a determined trajectory

Actuators of the robot are selected by using angular velocities which are obtained from kinematic analyses and required axis torques which are obtained from kinetic analysis. Harmonic Drive trademark gears are selected for this robot. Most important features of these gears are having compact structure and zero backlash. Mitsubishi MELSERVO generation servo motors are used for driving purpose (Mitsubishi, 2005). Harmonic drive gear and Mitsubishi servo motor are seen in Figure 3.39. Table 3.10 shows actuators' technical data, axes velocities and torques. Torque characteristics of motors are seen in Figure 3.40.

Figure 3.39 Harmonic Drive gear and Mitsubishi MELSERVO generation servo motors

Table 3.10 Technical properties of motors and gears which are used on the robot

	MOTOR (Mitsubishi, 2005)					**GEAR** (Harmonic Drive, 2006)			**AXIS**		
	Code	Continuous Rated Output (W)	Continuous Rated Torque (Nm)	Maximum Torque (Nm)	Rated Rotation Speed (rpm)	Code	Ratio	Rated Torque at Rated Speed 2000 rpm (Nm)	Continuous Torque (Nm)	Maximum Torque (Nm)	Rated Rotation Speed (rpm) [deg/s]
Axis 1	HC-KFS-73B	750	2.4	7.2	3000	HFUC-40-80-2UH-SP	80	206	192	576	37.5 [225]
Axis 2	HC-SFS-102B	1000	4.78	14.4	2000	HFUC-40-100-2UH-SP	100	265	478	1440	20 [120]
Axis 3	HC-KFS-43B	400	1.3	3.8	3000	HFUC-32-80-2UH-SP	80	137	104	304	37.5 [225]
Axis 4	HC-KFS-23B	200	0.64	1.9	3000	HFUC-20-80-2UH-SP	80	34	51.2	152	37.5 [225]
Axis 5	HC-KFS-13B	100	0.32	0.95	3000	HFUC-20-80-2UH-SP	80	34	25.6	76	37.5 [225]
Axis 6	HC-KFS-053B	50	0.16	0.48	3000	HFUC-14-100-2UH-SP	100	7.8	16	48	30 [180]

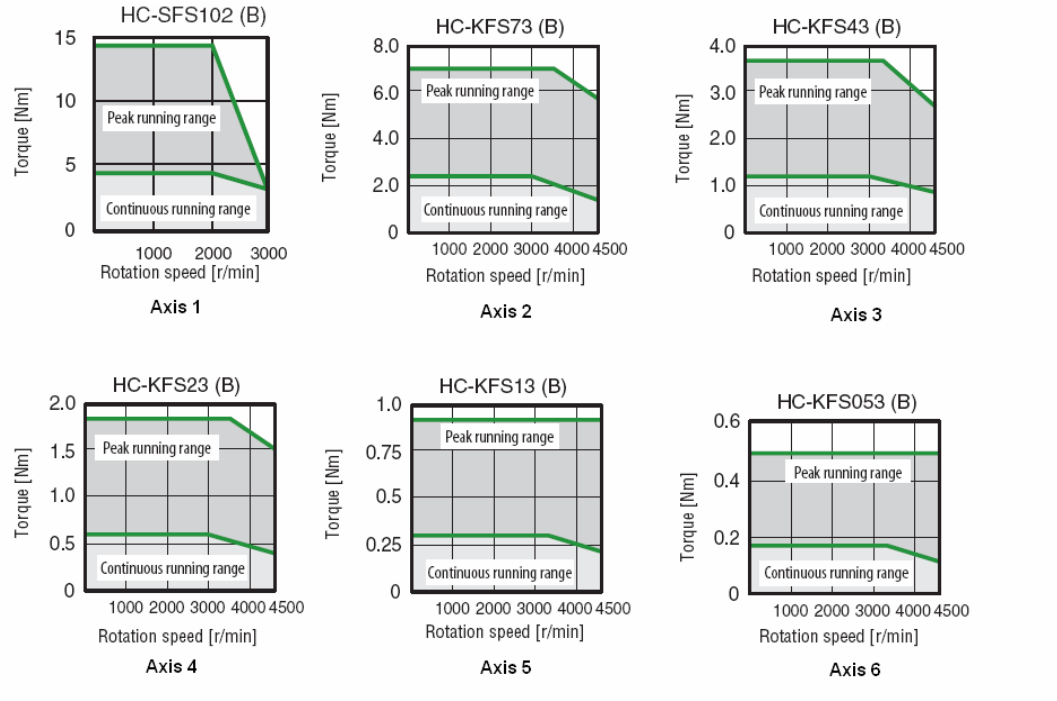


Figure 3.40 Torque characteristics of motors which are used on the robot (Mitsubishi, 2005)

Having selected the actuators, static and frequency analyses are performed for the first design of the robot. Maximum end point deflection which is calculated from static analysis is very important. This value is desired to be as possible as small. In this design application our design goal for end point deflection is 0.2 mm. The displacement results of the static analysis for the first design of the robot is shown in Figure 3.41

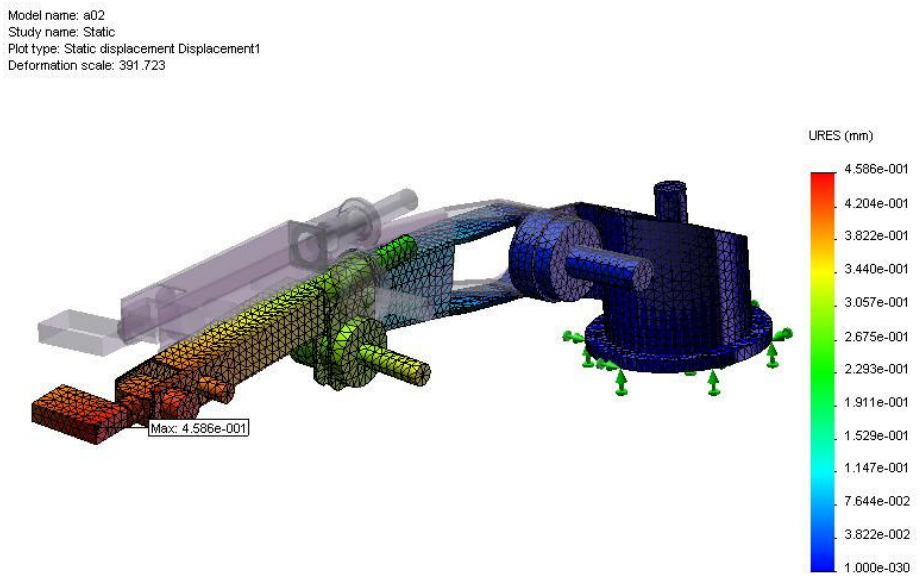


Figure 3.41 Displacement results of the static analysis for the first design of the robot

As seen from the figure that the maximum displacement of the end point is 0.458 mm. It is quite large with respect to the desired value of 0.2 mm. The natural frequencies of a robot manipulator give information about the rigidity. Certainly the stiffness depends on the configuration of the robot. The natural frequency analysis is performed for the different positions of the robot. Figure 3.42 shows the first three mode shapes obtained from the frequency analysis which is performed for the maximum reach distance position of the robot.

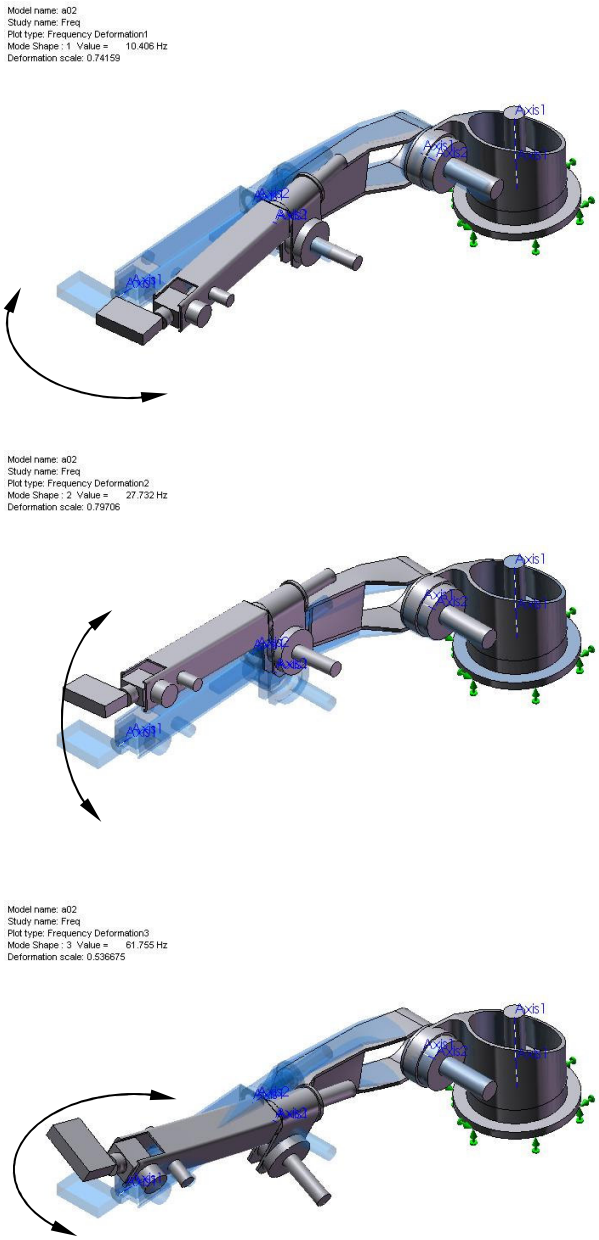


Figure 3.42 First three mode shapes of the robot for the maximum reach distance position

First, second and third natural frequencies are calculated as 10.406 Hz, 27.732 Hz and 61.755 Hz, respectively. First natural frequency is quite small for a robotic system implying small rigidity. First and second natural frequencies of ABB IRB 1400 robot are calculated 27.069 Hz and 29.602 Hz, respectively. Vibrations occur on the robot arms because of their rapid stoppages. To prevent the undesired vibrations, it is desired for robot manipulators to have a rigid structure.

Some modifications are done in order to obtain proper results for displacements and also for fundamental frequencies. Figure 3.43 shows the first and second form of the two main parts of the robot.

Figure 3.43 First and second form of the two main parts of the robot

The static displacement distribution on the robot manipulator is given in Figure 3.44 for the final design. As seen from the figure that the maximum displacement of the end point is 0.186 mm. This value fulfils the desired value.

The static stress results are seen in Figure 3.45 for the final design of the robot. Maximum Von-Mises stress value is decreased from 2.484×10^7 N/m^2 to 1.117×10^7 N/m^2 between the first and final design of the robot.

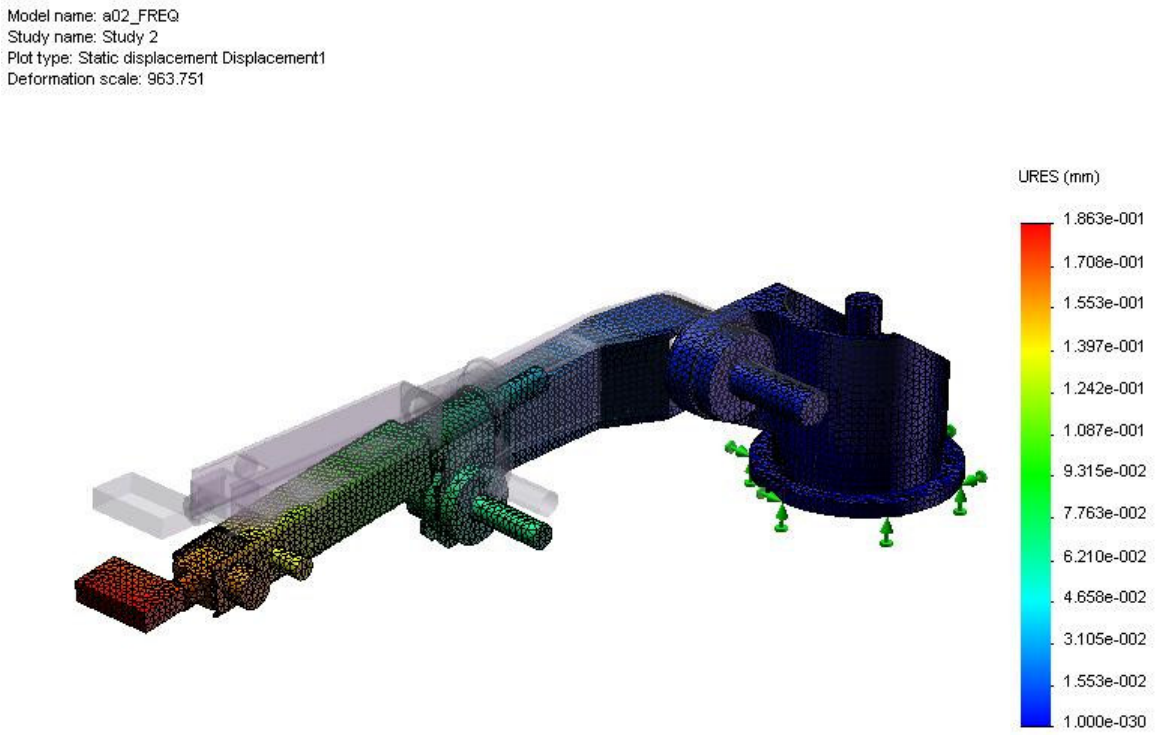


Figure 3.44 Static displacement results of the robot manipulator for the final design

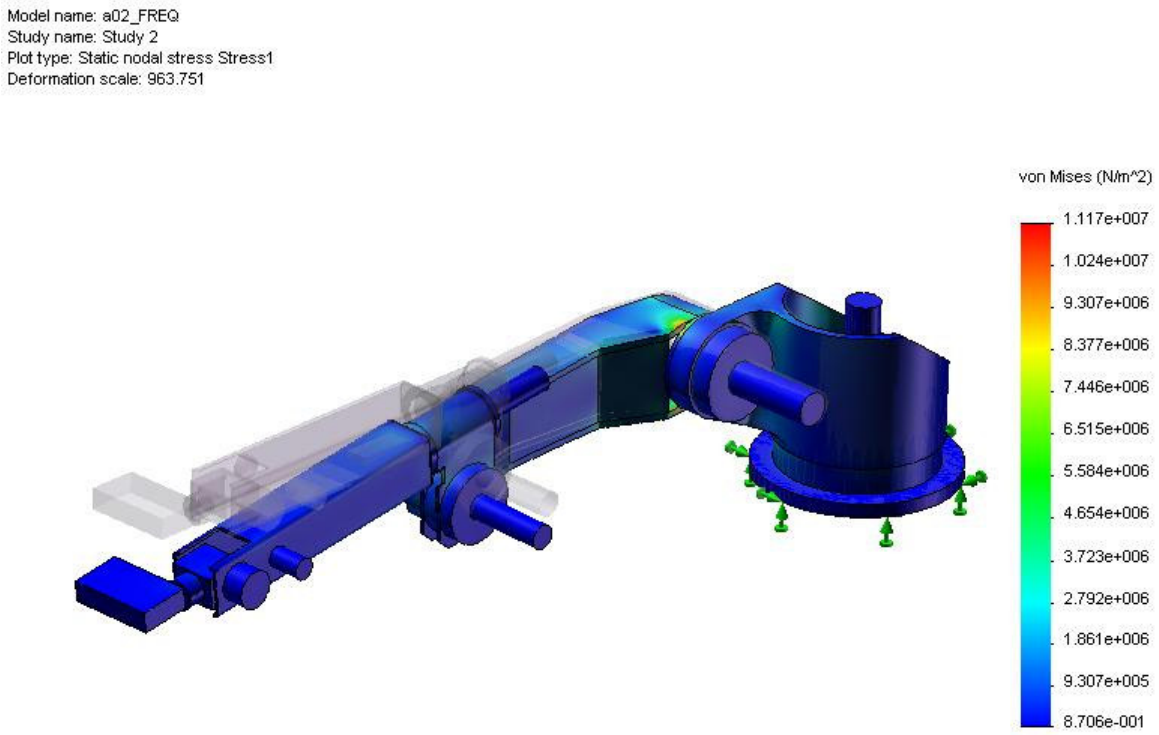


Figure 3.45 Static stress results of the robot manipulator for the final design

The results of the static and frequency analyses are listed in Table 3.11. Improvements in the design can be seen when the values are examined.

Table 3.11 Comparison of numerical results for the first and final design of the robot

	First Design	Final Design
Maximum Displacement of the End Point (mm)	0.458	0.186
Maximum Von-Mises Stress (N/m^2)	$2.484x10^7$	$1.117x10^7$
First Natural Frequency (Hz)	10.406	27.88
Second Natural Frequency (Hz)	27.732	47.216
Third Natural Frequency (Hz)	61.755	105.34

Evaluation and optimization studies are finished after these results. Detailed design of the robot is constructed and technical drawings are obtained. The detailed design of the robot is seen in Figure 3.46.

Figure 3.46 Detailed design of the robot (DEU-6X5-1500)

CHAPTER FOUR
DYNAMIC ANALYSIS OF ROBOT PARTS FOR DIFFERENT TRAJECTORIES

4.1 Theory of Nonlinear Dynamic Solution

Calculation of the stresses and strains on the components of moving systems is important. Inertial forces, center of masses and rigidity of the system are changing during the movement of the system. These dynamic calculations require nonlinear analyses. The ABAQUS finite element software is capable of doing this kind of nonlinear dynamic analysis. The program is used in literature frequently especially for nonlinear analyses because of its powerful solver. In this section, theoretical background used in the nonlinear analysis is given. (ABAQUS[2], 2004)

4.1.1 Nonlinear Solution Methods in ABAQUS/Standard

The finite element models generated in ABAQUS are usually nonlinear and can involve from a few to thousands of variables. In terms of these variables the equilibrium equations obtained by discretizing the virtual work equation can be written symbolically as

$$F^N\left(u^M\right) = 0 \tag{4.1}$$

where F^N is the force component conjugate to the N^{th} variable in the problem and u^M is the value of the M^{th} variable. The basic problem is to solve Equation (4.1) for the u^M throughout the history of interest.

Many of the problems to which ABAQUS will be applied are history-dependent, so the solution must be developed by a series of “small” increments. Two issues arise: how the discrete equilibrium statement Equation (4.1) is to be solved at each increment, and how the increment size is chosen.

ABAQUS/Standard generally uses Newton's method as a numerical technique for solving the nonlinear equilibrium equations. The motivation for this choice is primarily the convergence rate obtained by using Newton's method compared to the convergence rates exhibited by alternate methods (usually modified Newton or quasi-Newton methods) for the types of nonlinear problems most often studied with ABAQUS. The basic formalism of Newton's method is as follows. Assume that, after iteration i, an approximation u_i^M, to the solution has been obtained. Let c_{i+1}^M be the difference between this solution and the exact solution to the discrete equilibrium Equation (4.1). This means that

$$F^N\left(u_i^M + c_{i+1}^M\right) = 0 \tag{4.2}$$

Expanding the left-hand side of this equation in a Taylor series about the approximate solution u_i^M then gives

$$F^N\left(u_i^M\right) + \frac{\partial F^N}{\partial u^P}\left(u_i^M\right)c_{i+1}^P + \frac{\partial^2 F^N}{\partial u^P \partial u^Q}\left(u_i^M\right)c_{i+1}^P + c_{i+1}^Q + \ldots = 0 \tag{4.3}$$

If u_i^M is a close approximation to the solution, the magnitude of each c_{i+1}^M will be small, and so all but the first two terms above can be neglected giving a linear system of equations:

$$K_i^{NP} c_{i+1}^P = -F_i^N, \tag{4.4}$$

where

$$K_i^{NP} = \frac{\partial F^N}{\partial u^P}\left(u_i^M\right) \tag{4.5}$$

is the Jacobian matrix and

$$F_i^N = F^N\left(u_i^M\right) \tag{4.6}$$

The next approximation to the solution is then

$$u_{i+1}^{M} = u_{i}^{M} + c_{i+1}^{M} \tag{4.7}$$

and the iteration continues.

Convergence of Newton's method is best measured by ensuring that all entries in F_i^N and all entries in c_{i+1}^N are sufficiently small. Both these criteria are checked by default in an ABAQUS/Standard solution. ABAQUS/Standard also prints peak values in the force residuals, incremental displacements, and corrections to the incremental displacements at each iteration so that the user can check for these contingencies himself.

Newton's method is usually avoided in the large finite element codes, apparently for two reasons. First, the complete Jacobian matrix is sometimes difficult to formulate; and for some problems it can be impossible to obtain this matrix in closed form, so it must be calculated numerically which an expensive and not always reliable process is. Secondly, the method is expensive per iteration, because the Jacobian must be formed and solved at each iteration. The most commonly used alternative to Newton is the modified Newton method, in which the Jacobian in Equation (4.4) is recalculated only occasionally or not at all, as in the initial strain method of simple contained plasticity problems. This method is attractive for mildly nonlinear problems involving softening behavior such as contained plasticity with monotonic straining but is not suitable for severely nonlinear cases. In some cases ABAQUS/Standard uses an approximate Newton method if it is either not able to compute the exact Jacobian matrix or if an approximation would result in a quicker total solution time. For example, several of the models in ABAQUS/Standard result in a nonsymmetrical Jacobian matrix, but the user is allowed to choose a symmetric approximation to the Jacobian on the grounds that the resulting modified Newton method converges quite well and that the extra cost of solving the full nonsymmetrical system does not justify the savings in iteration achieved by the

quadratic convergence of the full Newton method. In other cases the user is allowed to drop interfiled coupling terms in coupled procedures for similar reasons.

Another alternative is the quasi-Newton method, in which Equation (4.4) is symbolically rewritten

$$c_{i+1}^{P} = -\left[K_i^{NP}\right]^{-1} F_i^{N} \tag{4.8}$$

and the inverse Jacobian is obtained by an iteration process.

There are a wide range of quasi-Newton methods. The more appropriate methods for structural applications appear to be reasonably well behaved in all but the most extremely nonlinear cases, the trade-off is that more iteration are required to converge, compared to Newton. While the savings in forming and solving the Jacobian might seem large, the savings might be offset by the additional arithmetic involved in the residual evaluations that is, in calculating the F_i, and in the cascading vector transformations associated with the quasi-Newton iterations. Thus, for some practical cases quasi-Newton methods are more economic than full Newton, but in other cases they are more expensive.

When any iterative algorithm is applied to a history-dependent problem, the intermediate, nonconverged solutions obtained during the iteration process are usually not on the actual solution path; thus, the integration of history-dependent variables must be performed completely over the increment at each iteration and not obtained as the sum of integrations associated with each Newton iteration, c_i. In ABAQUS/Standard this is done by assuming that the basic nodal variables, u, vary linearly over the increment, so that

$$u(\tau) = \left(1 - \frac{\tau}{\Delta t}\right) u(t) + \frac{\tau}{\Delta t} u(t + \Delta t) \tag{4.9}$$

where $0 \le \tau \le \Delta t$ represents "time" during the increment. Then, for any history-dependent variable, g(t), we compute

$$g(t+\Delta t) = g(t) + \int_{t}^{t+\Delta t} \frac{dg}{d\tau}(\tau) d\tau \qquad (4.10)$$

at each iteration.

The issue of choosing suitable time steps is a difficult problem to resolve. First of all, the considerations are quite different in static, dynamic, or diffusion cases. It is always necessary to model the response as a function of time to some acceptable level of accuracy. In the case of dynamic or diffusion problems time is a physical dimension for the problem and the time stepping scheme must provide suitable steps to allow accurate modeling in this dimension. Even if the problem is linear, this accuracy requirement imposes restrictions on the choice of time step. In contrast, most static problems have no imposed time scale, and the only criterion involved in time step choice is accuracy in modeling nonlinear effects. In dynamic and diffusion problems it is exceptional to encounter discontinuities in the time history, because inertia or viscous effects provide smoothing in the solution. However, in static cases sharp discontinuities such as bifurcations caused by buckling are common. Softening systems, or unconstrained systems, require special consideration in static cases but are handled naturally in dynamic or diffusion cases. Thus, the considerations upon which time step choice is made are quite different for the three different problem classes.

ABAQUS provides both "automatic" time step choice and direct user control for all classes of problems. Direct user control can be useful in cases where the problem behavior is well understood as might occur when the user is carrying out a series of parameter studies or in cases where the automatic algorithms do not handle the problem well. However, the automatic schemes in ABAQUS are based on extensive experience with a wide range of problems and, therefore, generally provide a reliable approach.

For static problems a number of schemes have been suggested for automatic step control (Bergan, Horrigmoe, Krakeland, & Soreide, 1978). ABAQUS/Standard uses a scheme based predominantly on the maximum force residuals following each iteration. By comparing consecutive values of these quantities, ABAQUS/Standard determines whether convergence is likely in a reasonable number of iterations. If convergence is deemed unlikely, ABAQUS/Standard adjusts the load increment; if convergence is deemed likely, ABAQUS/Standard continues with the iteration process. In this way excessive iteration is eliminated in cases where convergence is unlikely, and an increment that appears to be converging is not aborted because it needed a few more iterations. One other ingredient in this algorithm is that a minimum increment size is specified, which prevents excessive computation in cases where buckling, limit load, or some modeling error causes the solution to stall. This control is handled internally, with user override if needed. Several other controls are built into the algorithm; for example, it will cut back the increment size if an element inverts due to excessively large geometry changes. These detailed controls are based on empirical testing.

In dynamic analysis when implicit integration is used, the automatic time stepping is based on the concept of half-step residuals (Hibbitt & Karlsson, 1979). The basic idea is that the time stepping operator defines the velocities and accelerations at the end of the step $(t + \Delta t)$ in terms of displacement at the end of the step and conditions at the beginning of the step. Equilibrium is then established at $(t + \Delta t)$ which ensures an equilibrium solution at the end of each time step and, thus, at the beginning and end of any individual time step. However, these equilibrium solutions do not guarantee equilibrium throughout the step. The time step control is based on measuring the equilibrium error, the force residuals, at some point during the time step, by using the integration operator, together with the solution obtained at $(t + \Delta t)$, to interpolate within the time step. The evaluation is performed at the half step $(t + \Delta t/2)$. If the maximum entry in this residual vector, the maximum half-step residual, is greater than a user-specified tolerance, the time step is considered to be too big and is reduced by an appropriate factor. If the maximum half-step residual is sufficiently below the user-specified tolerance, the time step can be increased by an

appropriate factor for the next increment. Otherwise, the time step is deemed adequate. The algorithm is somewhat more complicated at traumatic events such as impact. Here, the time step can also be adjusted based on the magnitude of residuals in the first or second iteration following such events. Clearly, if these residuals are several orders of magnitude greater than those permitted, convergence is unlikely and the time step is altered immediately to avoid unproductive iteration.

4.2 Modelling of ABB IRB 1400 Robot Parts

The solid model of the ABB IRB 1400 robot is needed for dynamic FEM analyses. The solid model of the robot could not be obtained directly from the ABB website. Only the outer surface model is downloaded and then the solid model is created according to the results of the some geometrical measurements on the real system. SolidWorks program is used to obtain the solid model. Some geometrical assumptions are also made to create the whole solid model of the robot IRB 1400. The outer surface model of the robot which is obtained from ABB website (ABB, 2007) is given in Figure 4.1. This model consists of hundreds of planar surfaces. In addition, robot arms are settled in the opposite direction on the base arm. Most important problem on the robot model is that there is no information about the thicknesses of the robot parts. Determination of the part thicknesses is very important in the solid modeling process because it affects both total robot weight and stiffness. The other similar problem is there no knowledge about the weight of the system components like gears, bearings, transmission shaft and oil for lubrication. This knowledge is also very important because it affects the location of the center of gravity of the robot.

For modeling the ABB IRB 1400 robot which is in use in our laboratory, lots of dimensions are taken on the real model. The outer surface model is used as the reference model. In order not to dismount the IRB 1400, thickness values and the dimensions of some components in the robot are modeled approximately. In fact, these assumptions are very effective on inertia forces, rigidity of the parts and the total weight of the robot. The solid CAD model of the ABB IRB 1400 is given in Figure 4.2.

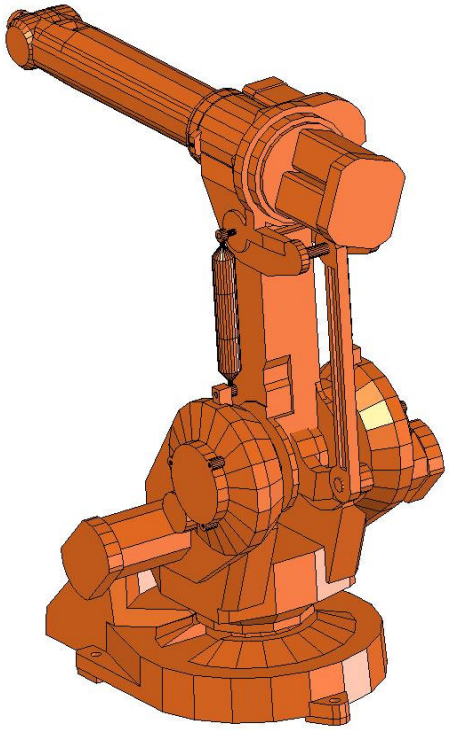


Figure 4.1 Outer surface model of ABB IRB 1400 robot (ABB, 2007)

Figure 4.2 The solid CAD model of the ABB IRB 1400

4.3 Dynamic Analysis of ABB IRB 1400 Robot

The solid CAD model of the IRB 1400 which was modeled in SolidWorks program is imported in the ABAQUS in order to perform the dynamic stress analysis by using the finite element method. Before the numerical analyses, all the solid parts of the robot are meshed with different mesh size. Figure 4.3 shows the finite element model of the robot. Four node linear tetrahedral elements are used for Arm2, Arm3, Arm4, Force Arm1 and Force Arm2. Hexahedral elements are generally more reliable element type but the complex structured robot arms could not be meshed with this element. Four node, quadrilateral, stress/displacement shell element is selected for Arm 1 because this part is the largest part and there are negligible stresses expected during the dynamic analysis. This shell element is preferred in order to decrease the element number in the finite element model. Table 4.1 shows the details of mesh and mass properties of the parts.

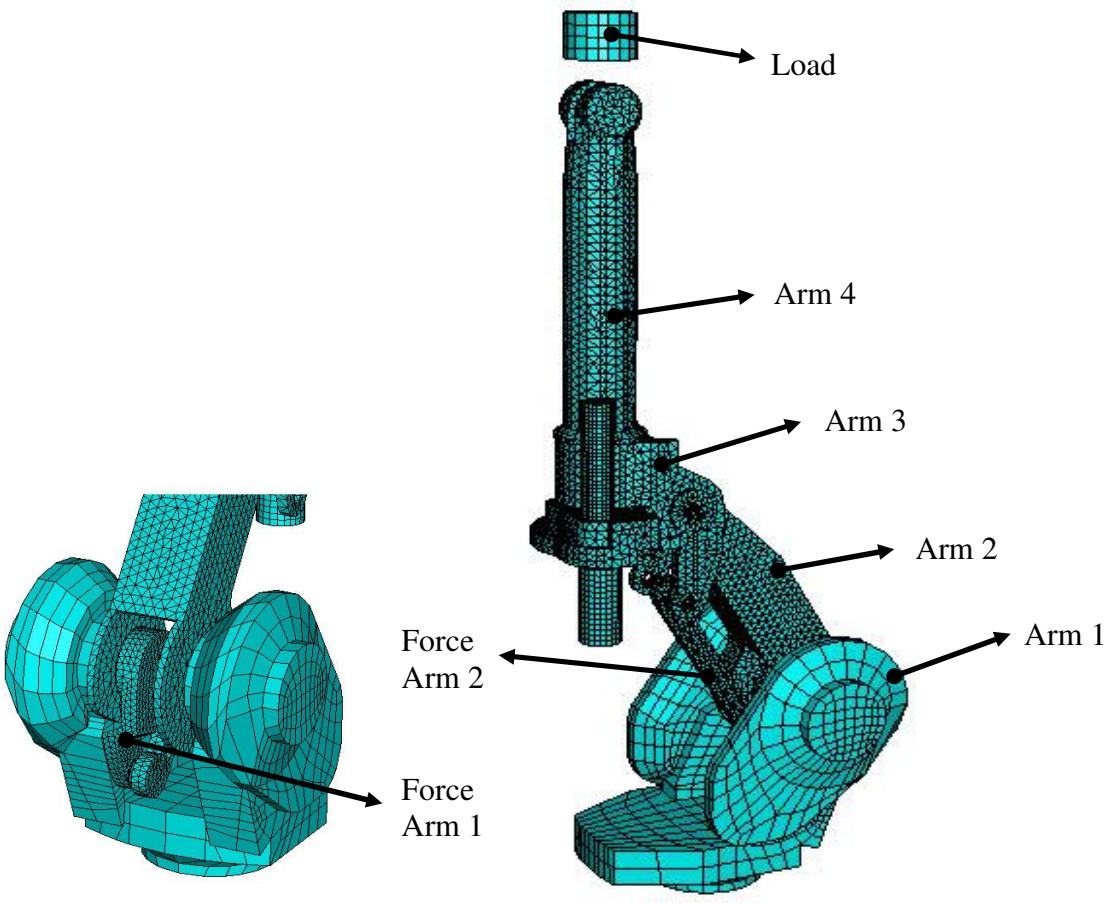
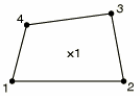
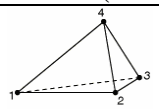
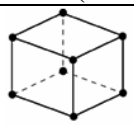


Figure 4.3 The finite element model of the ABB IRB 1400

Table 4.1 Mesh details and mass properties of the robot parts

Part Name	Approximate global mesh size (mm)	Element type	Number of elements	Mass (kg)
Arm1	30	 Quadrilateral Element (S4R)	1380	_
Arm2	15	 Tetrahedral element (C3D4)	6777	13.92
Arm3	14	Tetrahedral element (C3D4)	9319	24.29
Arm4	15	Tetrahedral element (C3D4)	6734	13.62
Force Arm1	10	Tetrahedral element (C3D4)	6531	5.92
Force Arm2	10	Tetrahedral element (C3D4)	3397	2.96
Load	20	 Hexahedral element (C3D8R)	144	5.6

For the dynamic analyses, three different end point trajectories shown in Figure 4.4 are created by ABB Robot Studio program.

The velocity of the end point is chosen as 500 mm/s and the then the dynamic stress analyses are performed. Total moving time of the robot is 6.08 seconds for trajectory 1, 6.34 seconds for trajectory 2 and 6.08 seconds for trajectory 3. As seen from these values, first and third trajectories have the same total moving time. In these trajectories the same job is performed in the same time period. The second trajectory with dimension is seen in Figure 4.5. Maximum reach distance for this trajectory is 955 mm while the maximum reach distance of the robot is 1440 mm. So, the defined motions are not arduous job to perform by the robot.

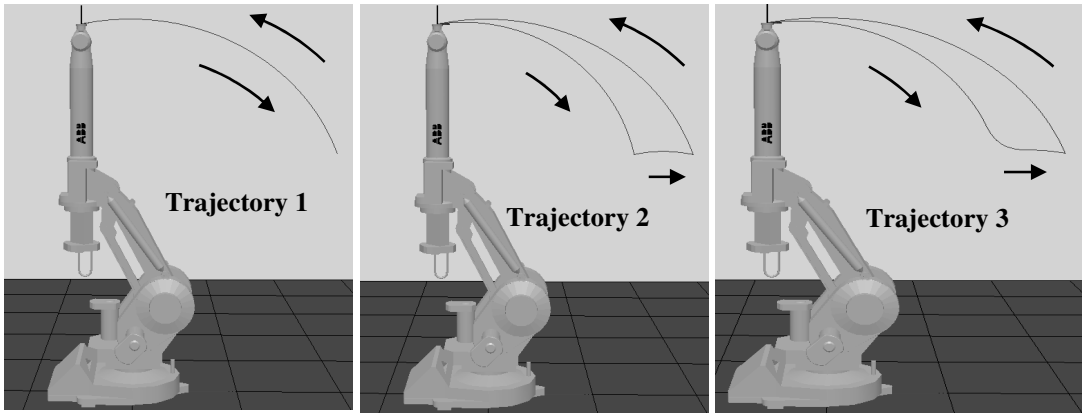


Figure 4.4 Three different end point trajectories

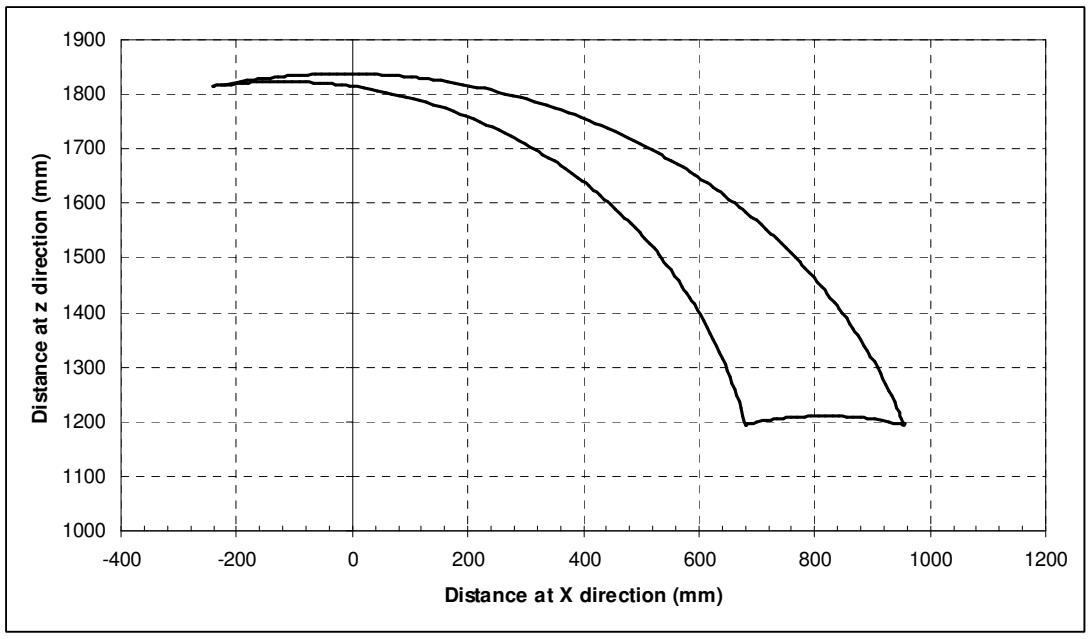


Figure 4.5 The end point trajectory with dimensions (Trajectory 2)

Firstly, the end point velocity profiles obtained from ABB Robot Studio are used as the end point input in ABAQUS and the joint velocities are calculated. Then these joint velocities are used as the input for the finite element analysis. Figure 4.6 shows the input-output relationship between the programs. ABB Robot Studio program uses the rigid body dynamics and elastic deformations are not considered. In the real robot system, elastic deformations occur during the motion. The elastic deformations and respective strain and stress values are calculated using the ABAQUS finite element package considering the joint velocities, inertias of robot components and the payload.

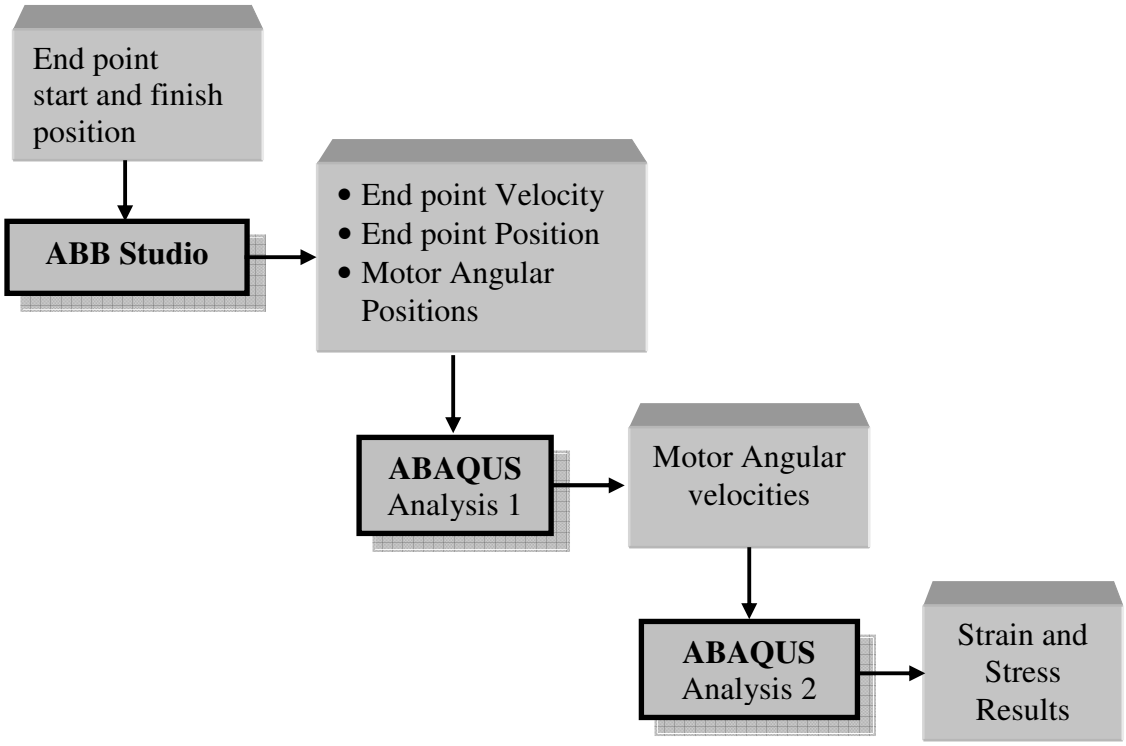


Figure 4.6 Input-output relationships between the programs

The end point velocity profiles obtained from ABB Robot Studio and the results of the second analysis made by using ABAQUS are shown in Figure 4.7. The end point velocity results calculated using ABAQUS are slightly different from the values obtained by ABB Studio due to the dynamic considerations in the finite element analysis.

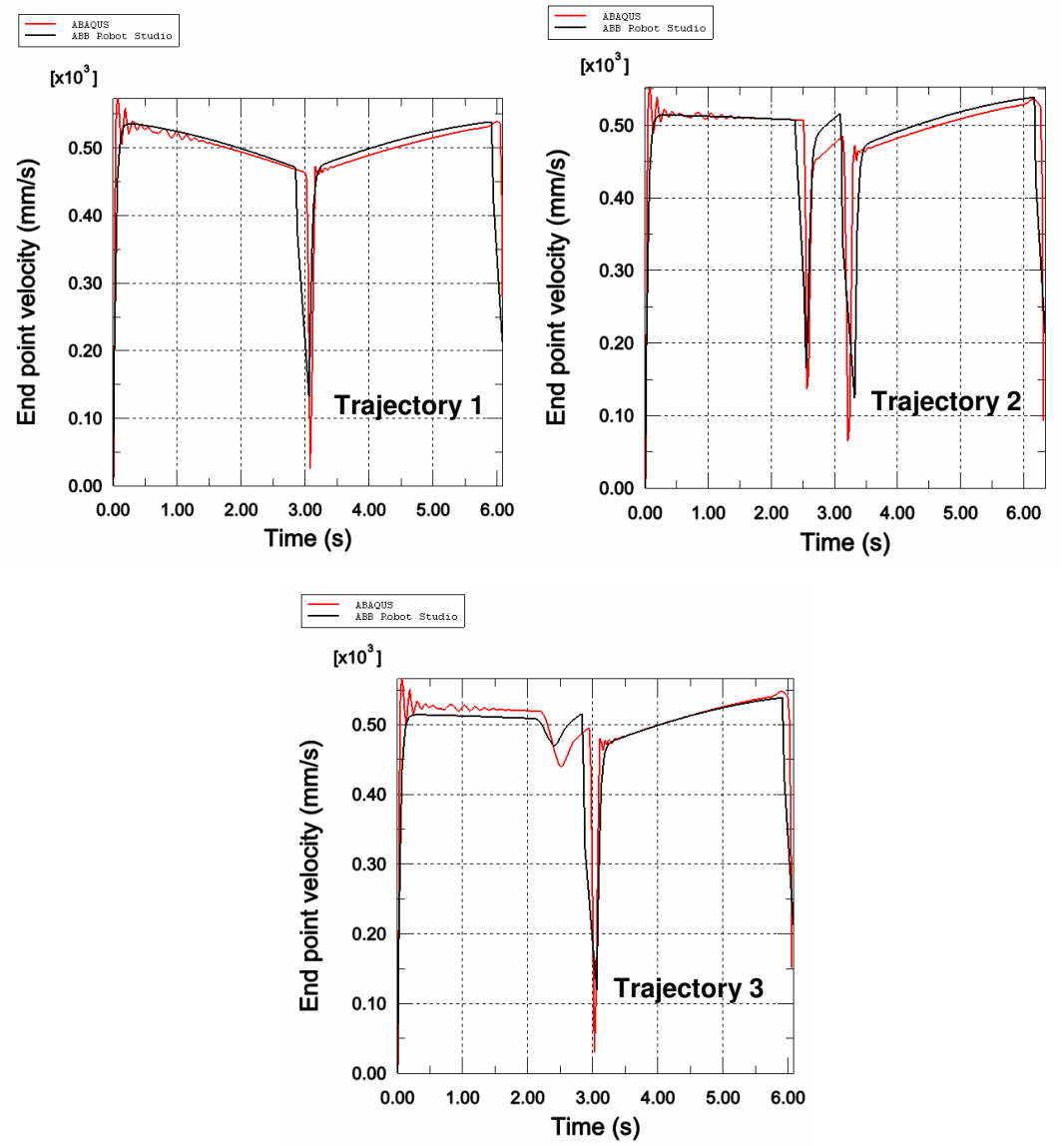


Figure 4.7 End point velocity profiles obtained from ABB studio and ABAQUS programs for three different trajectories

The finite element analyses are performed for three end point trajectories in order to determine the effect of trajectory on the dynamic results. Dynamic strain and stress values are calculated in x direction for point A shown in Figure. 4.8. Point A is located near the root and x direction is kept parallel to the Arm 4 during the motion. The results are given in Figures 4.9, 4.10 and 4.11.

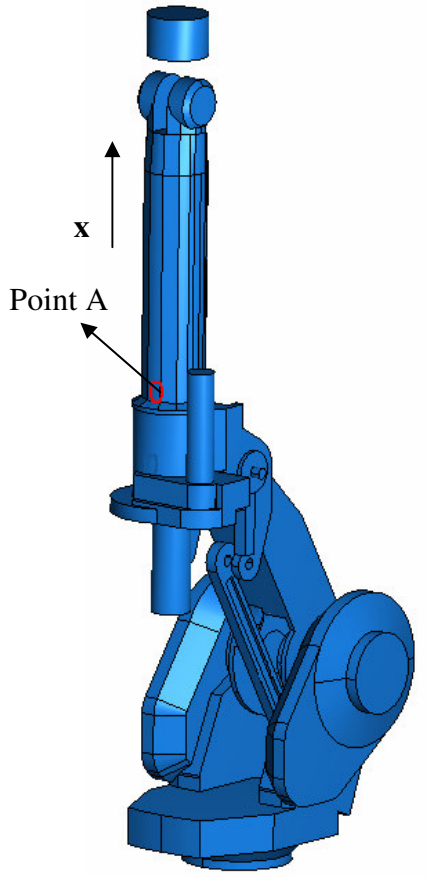


Figure 4.8 The point for which the stress and strain behavior is calculated

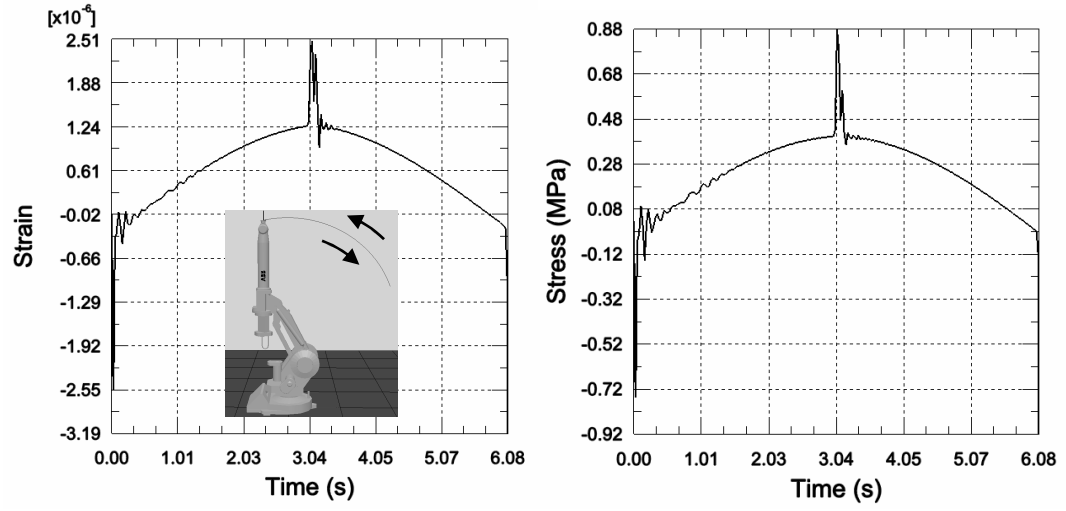


Figure 4.9 Strain and stress results on the point A for trajectory 1, 500 mm/s end point velocity with 5.6 kg mass.

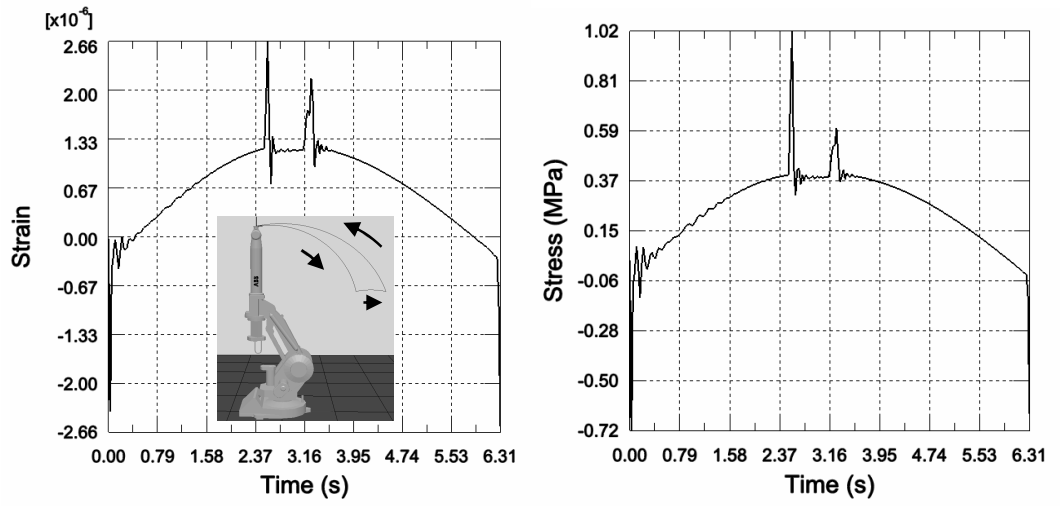


Figure 4.10 Strain and stress results on the point A for trajectory 2, 500 mm/s end point velocity with 5.6 kg mass.

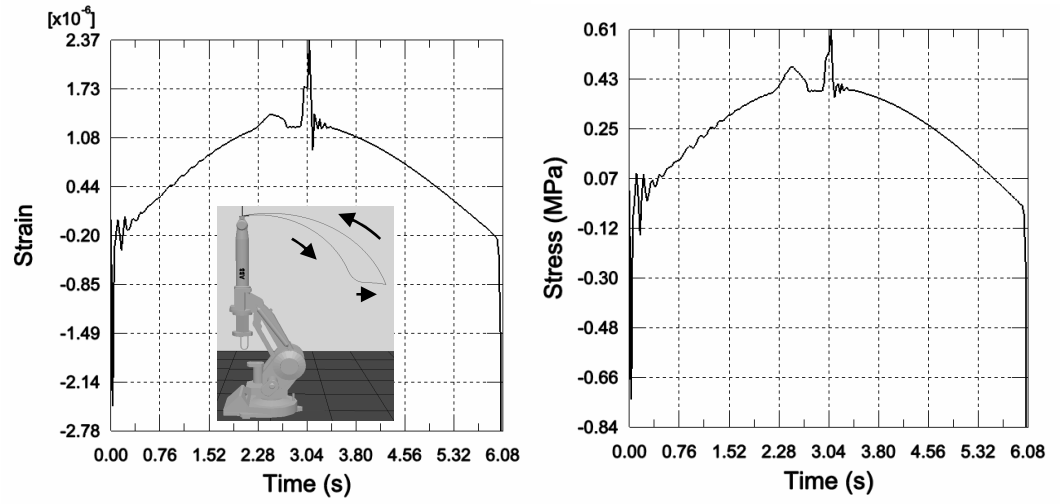


Figure 4.11 Strain and stress results on the point A for trajectory 3, 500 mm/s end point velocity with 5.6 kg mass.

As seen from the figures maximum strain value is obtained for trajectory 2 when the manipulator reaches to end of the trajectory. There are two spikes in the trajectory 2 corresponding to the shape of the trajectory. The magnitude of the first spike in the strain and stress responses reduces when the first corner in the trajectory is replaced by a radius. Minimum strain and stress values were calculated for trajectory 3. Comparison of strain and stress results for three trajectories is given in Figure 4.12.

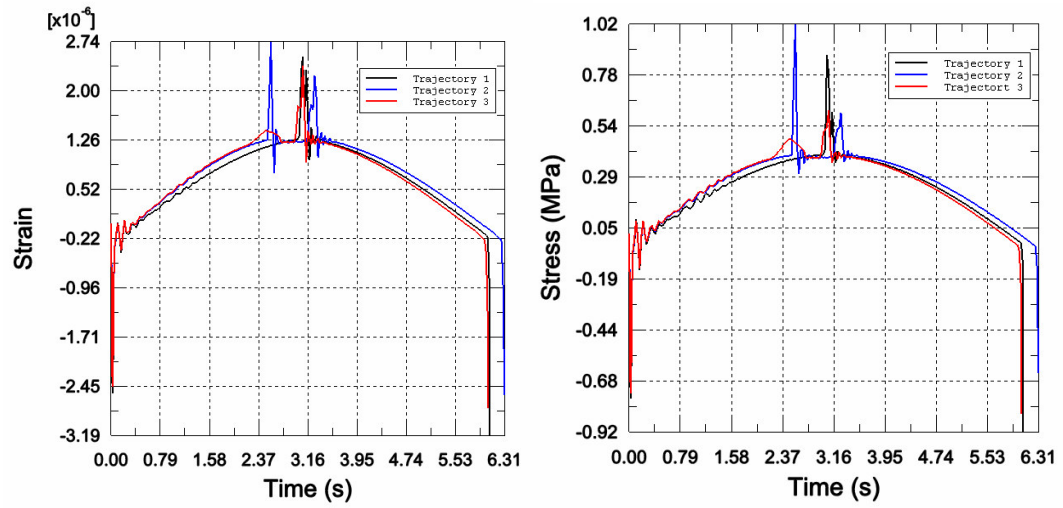


Figure 4.12 Comparison of strain and stress results for three trajectories

As seen from the figure that stress results for trajectory 3 are lower than those calculated for other trajectories. Trajectory 1 and trajectory 3 have the same total motion time, but stress results of the trajectory 3 are lower than about 30% from the results of trajectory 1. When the end point velocity of the robot increases, difference between the results will also increase due to the inertial forces. Dynamic analyses are repeated for 1000 mm/s end point velocity. Results of these analyses are seen in Figures 4.13, 4.14 and 4.15.

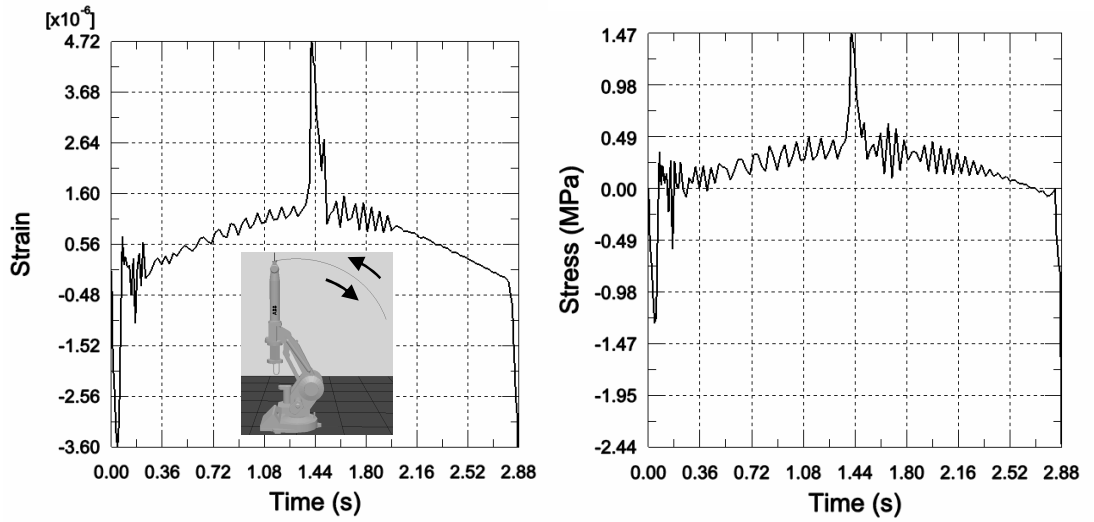


Figure 4.13 Strain and stress results on the point A for trajectory 1, 1000 mm/s end point velocity with 5.6 kg mass.

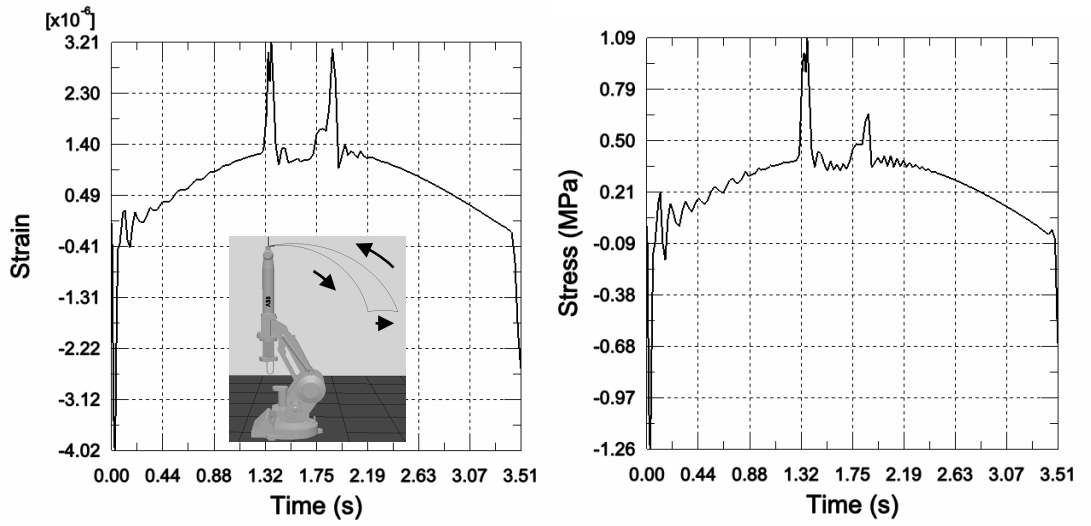


Figure 4.14 Strain and stress results on the point A for trajectory 2, 1000 mm/s end point velocity with 5.6 kg mass.

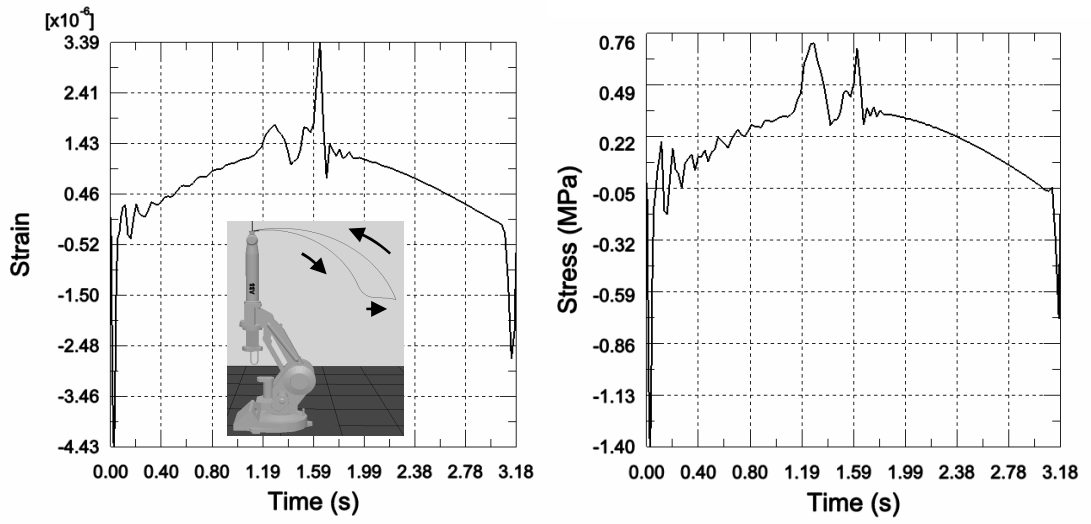


Figure 4.15 Strain and stress results on the point A for trajectory 3, 1000 mm/s end point velocity with 5.6 kg mass.

As seen from the figures, maximum strain and stress values are calculated for trajectory 1. The total motion times of trajectory 1 and trajectory 3 are not equal for 1000 mm/s end point velocity. The total motion time of trajectory 3 increase 10% with respect to the total motion time of trajectory 1. Nevertheless stress values on the point A for trajectory 3 decrease 50% percent with respect to the stress values obtained for trajectory 1. Comparison of strain and stress results for three trajectories at 1000 mm/s end point velocity is seen in Figure 4.16.

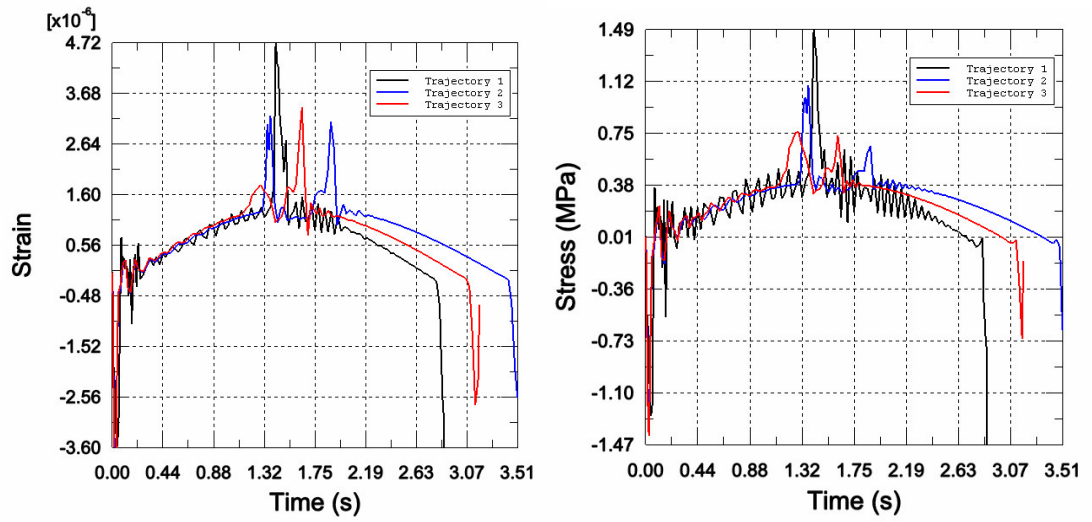


Figure 4.16 Comparison of strain and stress results for three trajectories, 1000 mm/s end point velocity

The stress values obtained in this study may be considered very small with respect to the strength of the robot material for the considered trajectories, but changes in the dynamic results show the importance of the trajectory design. The proper trajectory design may become very important when the manipulator durability is in question and the increase in the total movement time may be ignored. A sample view of the maximum principle strain distribution on the robot is shown in Fig. 4.17 for different instants of the trajectory 1.

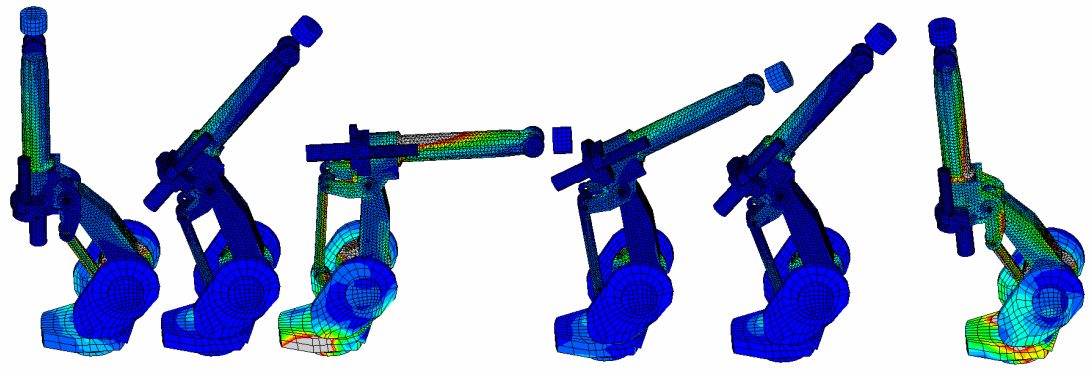


Figure 4.17 Strain distributions on the robot during motion for trajectory 1

4.4 Experimental Results for ABB IRB 1400

In this section, the results of an experimental study for trajectory 1 are presented. The experiments are conducted with and without mass attached to the end point of the robot. In these experiments, four different end point (TCP-Tool Center Point) velocities are used. Table 4.2 shows the cases considered in the experimental study. For the case that the mass attached to the end point, the robot could not be operated at high speeds due to the design limitations. High acceleration occurs at the beginning of the motion for high end point velocity definition. It causes high value of electrical current on the motors. Motor current is controlled by ABB robot controller and if it exceeds the limit, controller terminates the motion.

Table 4.2 Experimental study cases

TCP Velocity	**Without Mass**	**With Mass (5.6 kg)**
250 mm/s	✓	✓
500 mm/s	✓	✓
1000 mm/s	✓	X
2000 mm/s	✓	X

Three strain-gauges are used in order to measure the dynamic strain values on three different parts of the robot. The positions of the strain-gauges are chosen for the locations for which the largest strain values are expected. Figure 4.18 shows the positions of the strain-gauges on the robot parts and the measurement system schematically. A data acquisition unit, National Instruments, having sixteen input channel is used. The strain-gauges are connected to the NI SC-SG01 strain-gauge input module. The strain signals are sent to the computer through NI SC-2345 signal conditioning unit and NI PCI-6220 data acquisition card. The measurement signals are processed by a program written in LabVIEW.

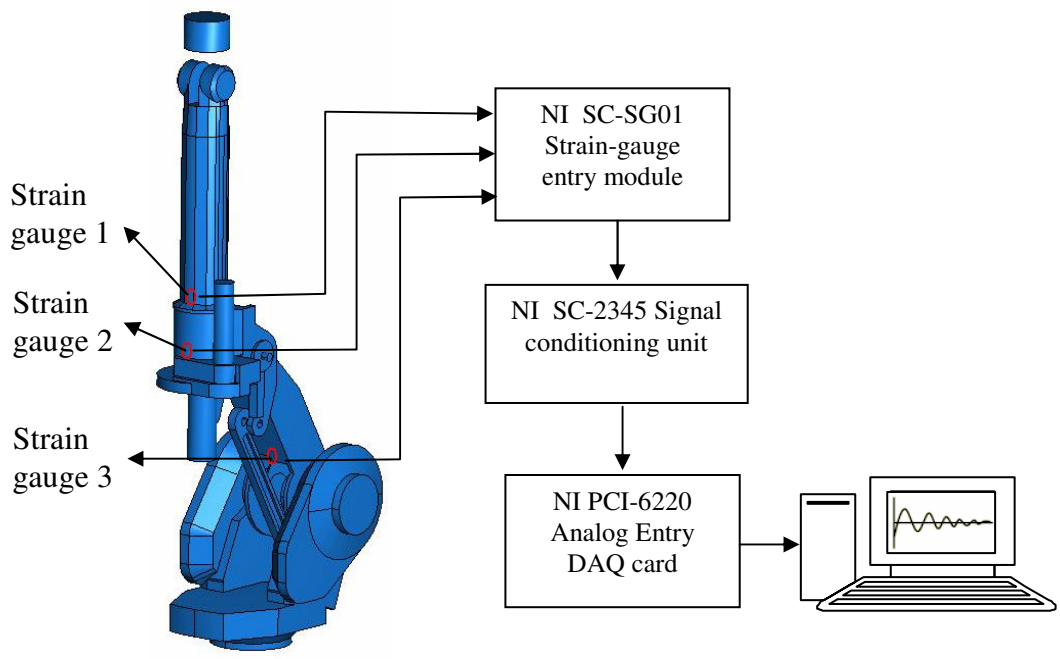


Figure 4.18 Location of strain gauges and scheme of measurement system

The results of the measurements show that the strain values for strain gauge 2 and strain gauge 3 are very small and it is very difficult to isolate them from the noise coming from electrical units.

Strain-gauges are located on the robot at the start position. The strain values recorded for this position are used as the reference zero at the beginning of the experiments. Experimental dynamic strain signals were filtered by using a program written in MATLAB. Experimental strain results for strain gauge 1 when the robot moving on trajectory 1 without mass at different end point velocities are seen in Figures between 4.19 and 4.22. Figure 4.23 and 4.24 show the strain results for strain gauge 1 when the robot moving on trajectory 1 with 5.6 kg mass at 250 mm/s and 500 mm/s end point velocities.

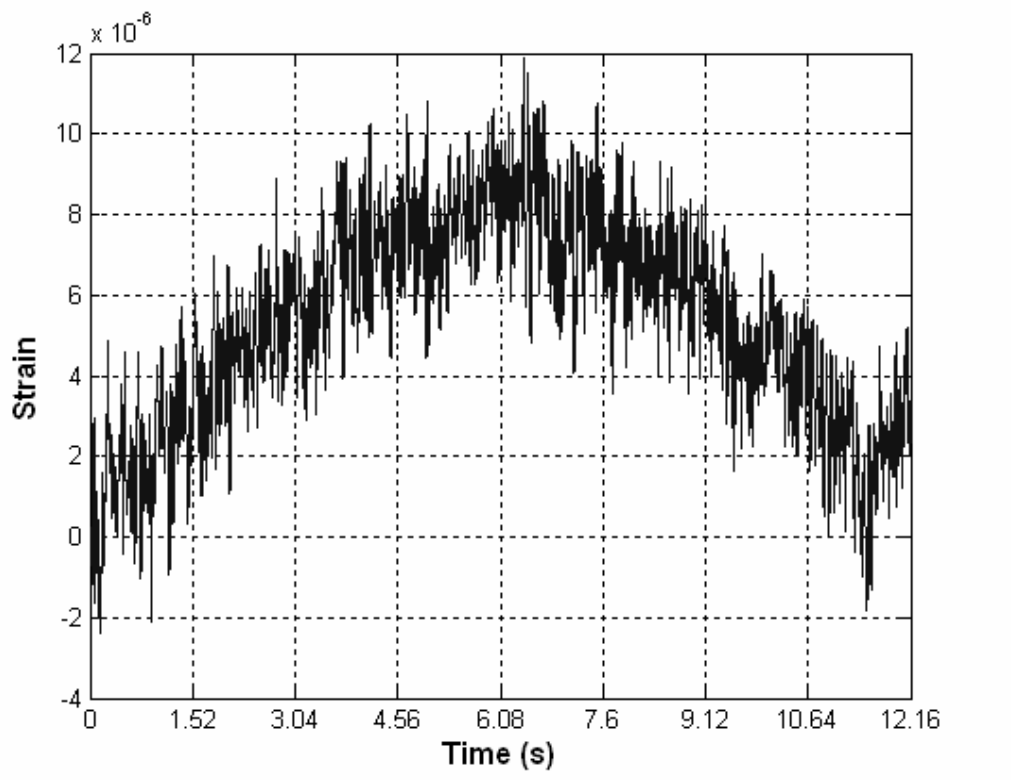


Figure 4.19 Strain results for strain gauge 1 when the robot moving on trajectory 1 with 250 mm/s end point velocity (without mass).

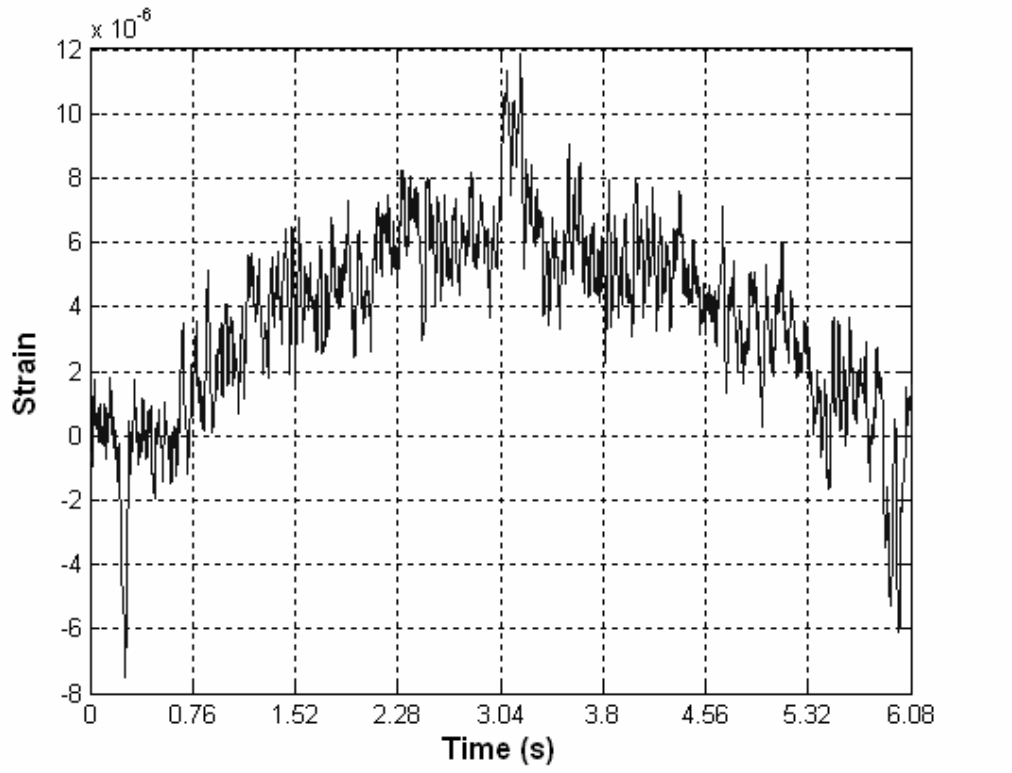


Figure 4.20 Strain results for strain gauge 1 when the robot moving on trajectory 1 with 500 mm/s end point velocity (without mass).

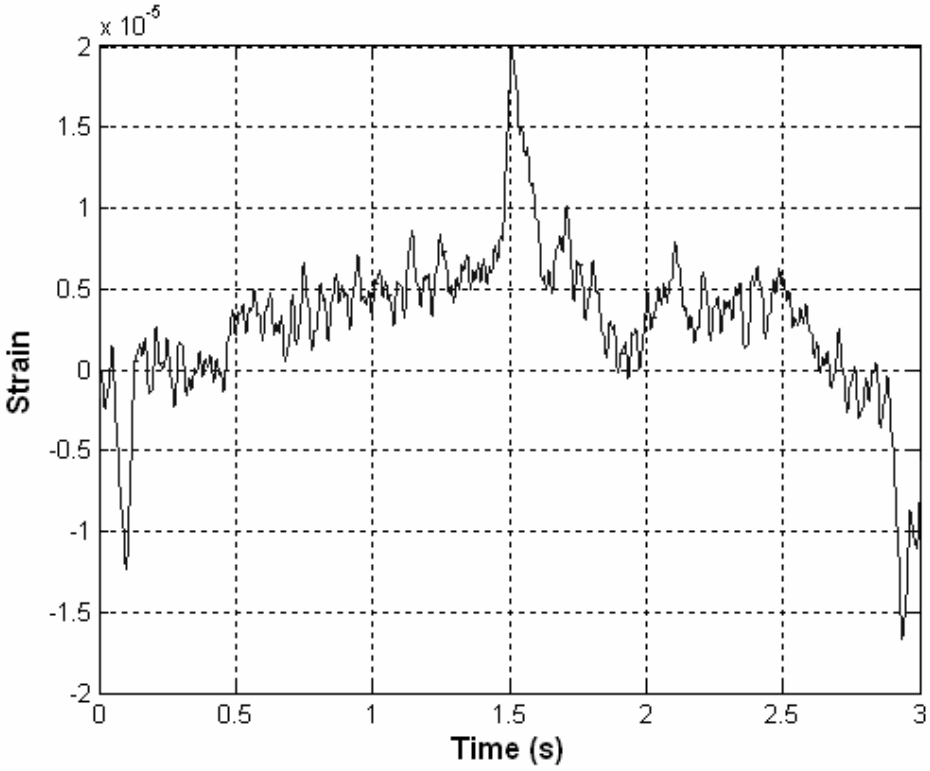


Figure 4.21 Strain results for strain gauge 1 when the robot moving on trajectory 1 with 1000 mm/s end point velocity (without mass).

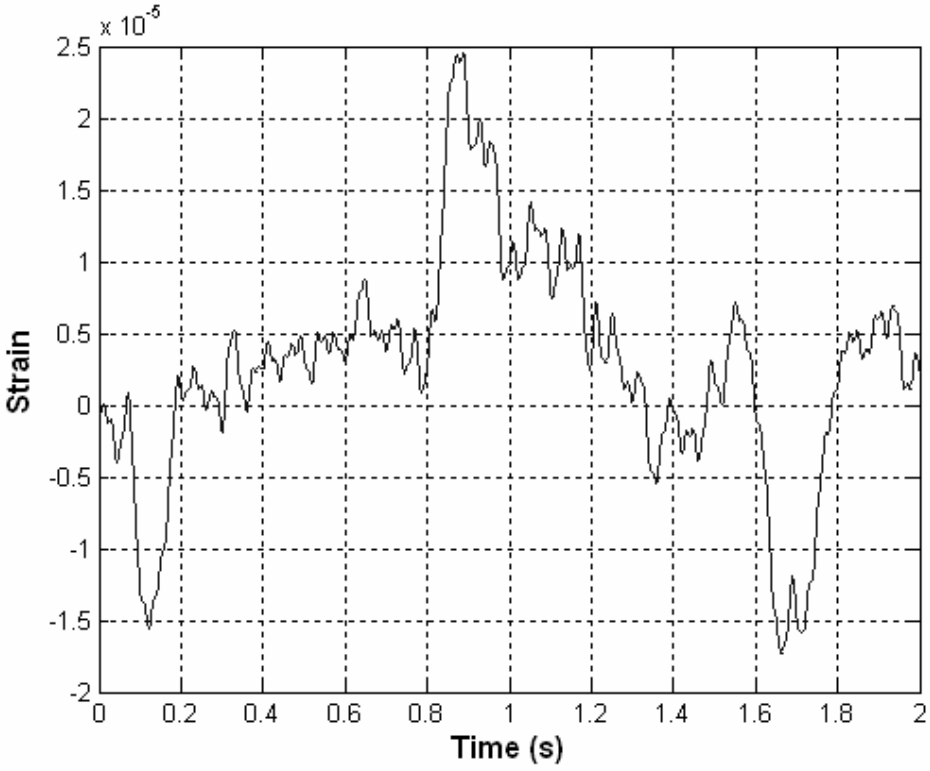


Figure 4.22 Strain results for strain gauge 1 when the robot moving on trajectory 1 with 2000 mm/s end point velocity (without mass).

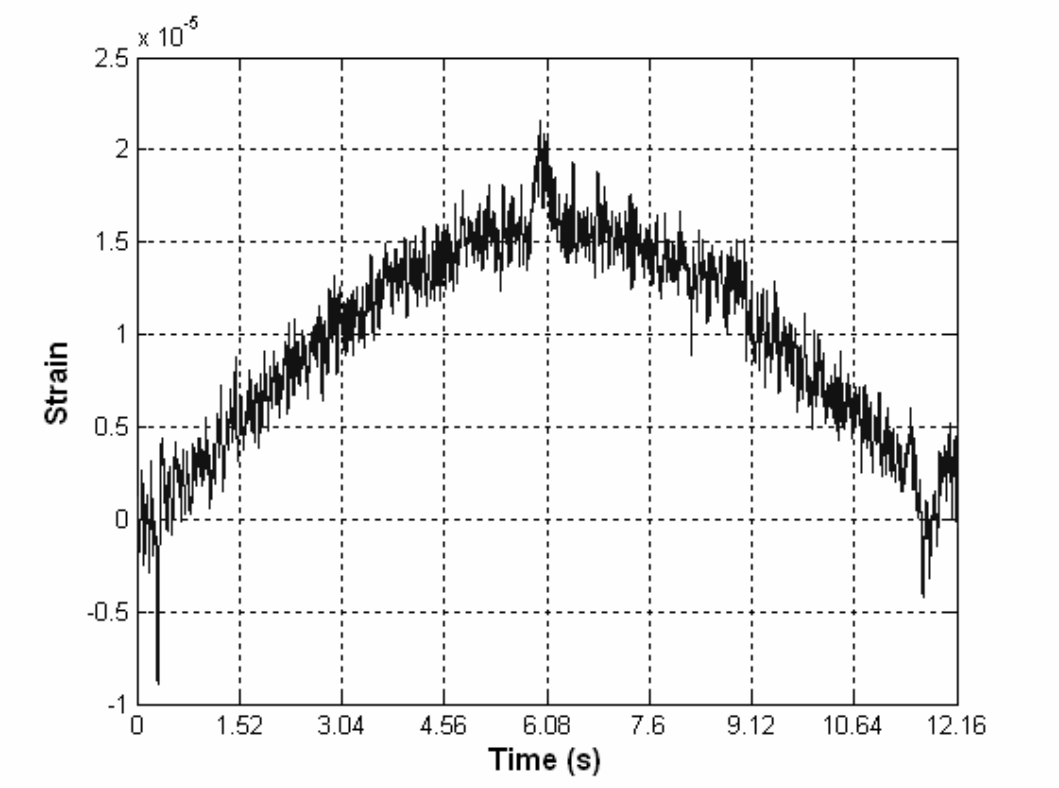


Figure 4.23 Strain results for strain gauge 1 when the robot moving on trajectory 1 with 250 mm/s end point velocity (with 5.6 kg mass).

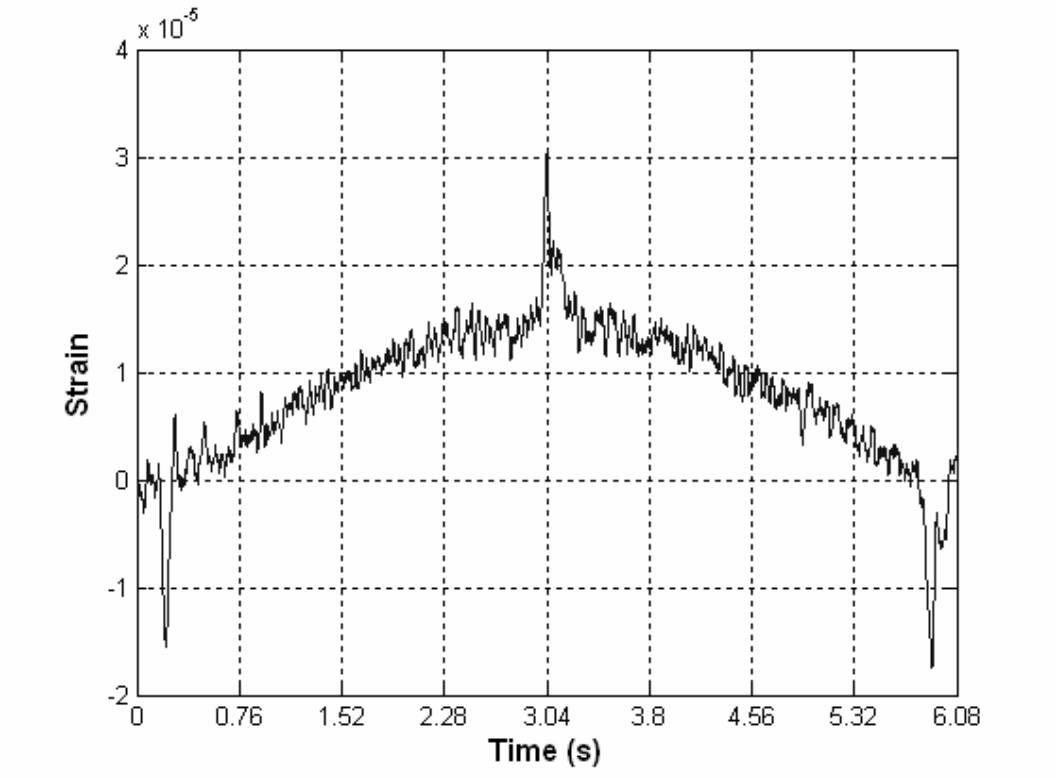
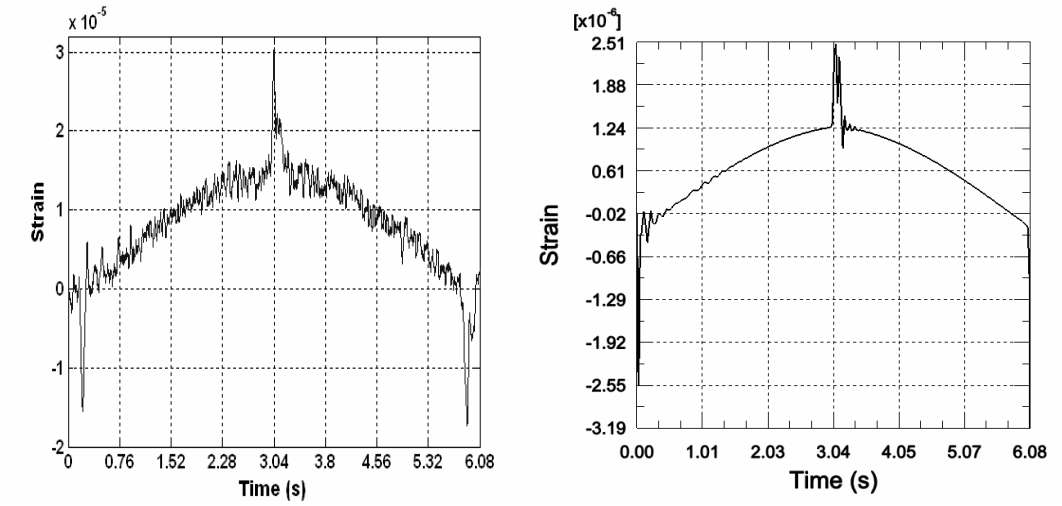


Figure 4.24 Strain results for strain gauge 1 when the robot moving on trajectory 1 with 500 mm/s end point velocity (with 5.6 kg mass).

4.5 Conclusions

The results obtained from the experimental study and FEM analyses seem to be well matched. There are some differences in the strain magnitudes due to the fact that the CAD model does not reflect the real robot model entirely. So, these discrepancies are expected. Experimental results are approximately ten times greater than the results obtained from numerical analyses. Figure 4.25 shows the difference between the numerical and experimental results. It is possible to reduce the difference between the results when the real robot system is modeled exactly.



(a) (b)

Figure 4.25 Comparison of a) experimental and b) numerical strain results when the robot moving on trajectory 1 with 500 mm/s end point velocity (with 5.6 kg mass).

The dynamic strain results at 500 mm/s end point velocity for the cases with and without end point mass are shown in Figure 4.26. The results show that the maximum strain value obtained for the 5 kg end point mass (payload) is 3 times greater than the maximum strain value for no point mass case. This emphasis the influence of dynamic effects even for small payloads with respect to the total weight of the robot manipulator.

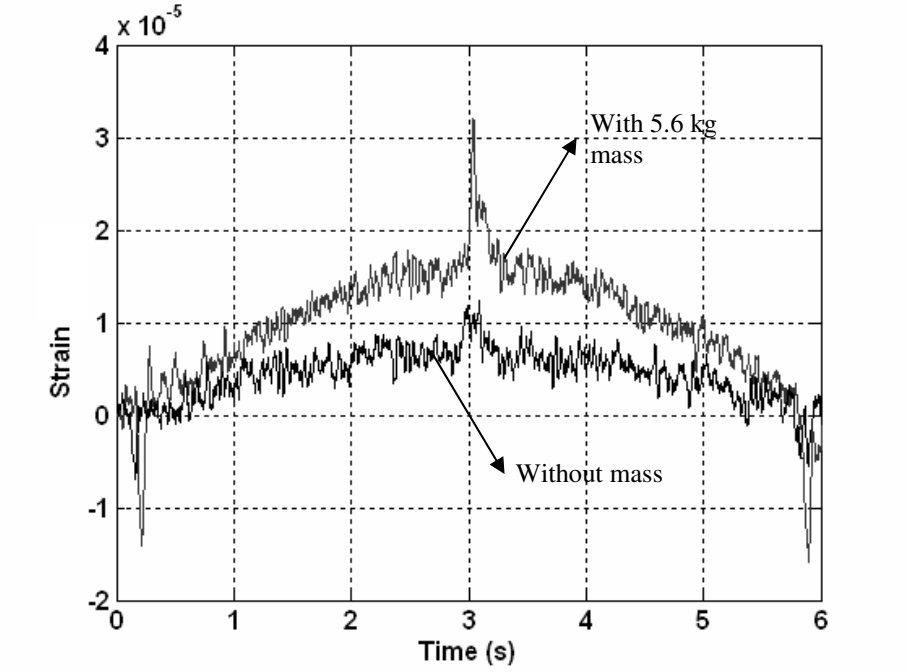
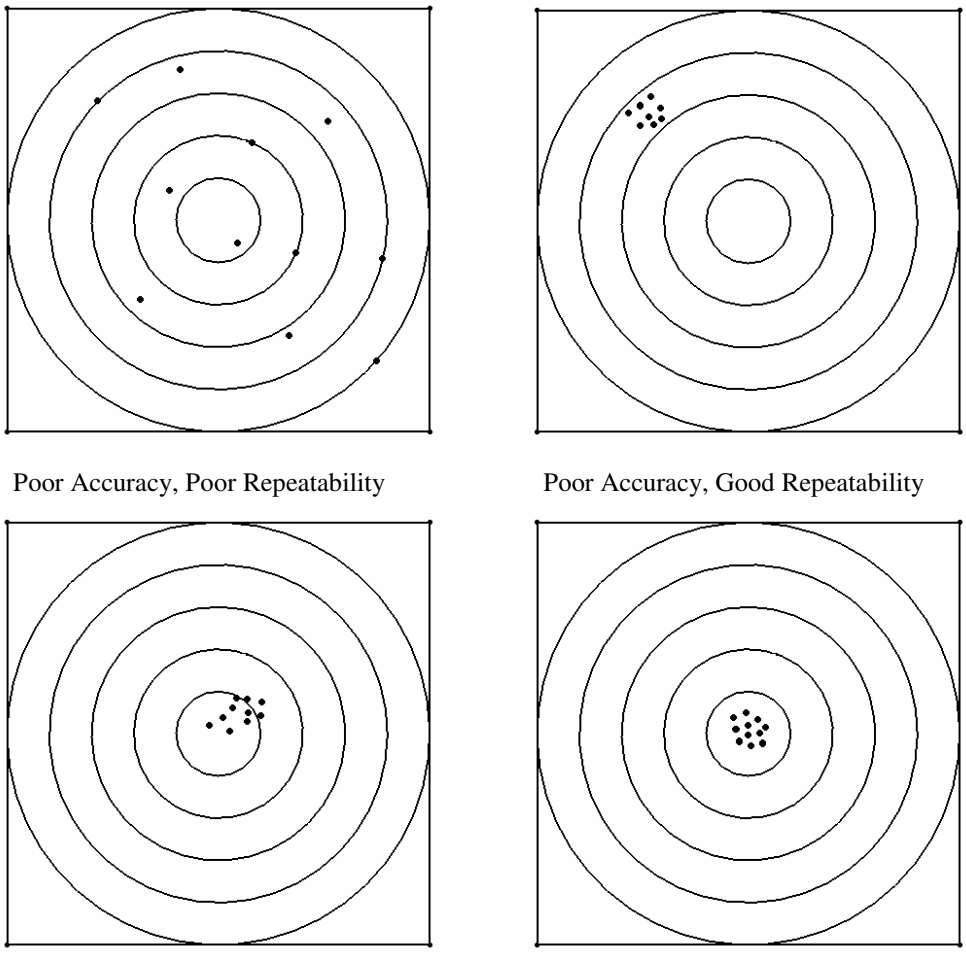


Figure 4.26 Comparison of experimental strain results with and without mass when the robot moving on trajectory 1 at 500 mm/s end point velocity

CHAPTER FIVE
NEW APPROACH: RIGIDITY WORKSPACE

5.1 Compensation of Static Deflection

Generaly there are two main concepts about the robot sensitivity. These are accuracy and repeatability. The position is usually measured at the end of the arm for robots. Accuracy is the difference between the measured value and the command value of a specified position in the robot's workspace. Repeatability is a measure of the spread of positions in a series of attempts to position the manipulator at a fixed location. A graphical illustration of accuracy and repeatability is shown in Figure 5.1.



Poor Accuracy, Poor Repeatability

Poor Accuracy, Good Repeatability

Good Accuracy, Poor Repeatability

Good Accuracy, Good Repeatability

Figure 5.1 Graphical definition of accuracy and repeatability (N-Nagy & Siegler, 1987).

The deflections for a robot system strongly depends on the stiffness and the weight of the robot. The stiffness of bearings in joints, reduction gears and transmission mechanisms are not negligible because their stiffness values are generally lower than the arms and increase the end point deflections. Deflection affects the accuracy of the robot remarkably. For this reason, compensation of static deflection is performed for industrial robots for which the accuracy is crucial. For the compansation procedure, the most important subject is the change in the static deflection values due to the positions of the robot and the payload. The change due the position has a nonlinear characteristic.

An accuracy compensation method which is used by ABB robot company is explained in section 5.1.1. This method is applied one by one for every manufactured robot and a special compensation data file is created for each robot. Every manufactured robot is certificated at the end of this compensation procedure.

5.1.1 Absolute Accuracy Compensation Method

Absolute Accuracy (ABB 2004) is a calibration concept, which ensures that the Tool Center Point (TCP) accuracy in most cases is better than ±1 mm in the entire working range. The difference between an ideal robot and a real robot can be several millimeters, resulting from mechanical tolerances and deflection in the robot structure. Absolute Accuracy compensate for these differences, ensuring that the given coordinates coincide with the actual robot position. Here are some examples of when this accuracy is important:

- Off-line programming with minimum touch-up.
- Exchangeability of robots
- On-line programming with accurate reorientation of tool
- Re-use of programs between applications

The errors compensated for in the controller derive from the mechanical tolerances of the constituent robot parts. A subset of these is detailed in Figure 5.2. Compliance errors are due to the effect of the robot's own weight together with the weight of the current payload. These errors depend on gravity and the characteristics of the load.

Kinematic errors are caused by position or orientational deviations in the robot axes. These are independent of the load.

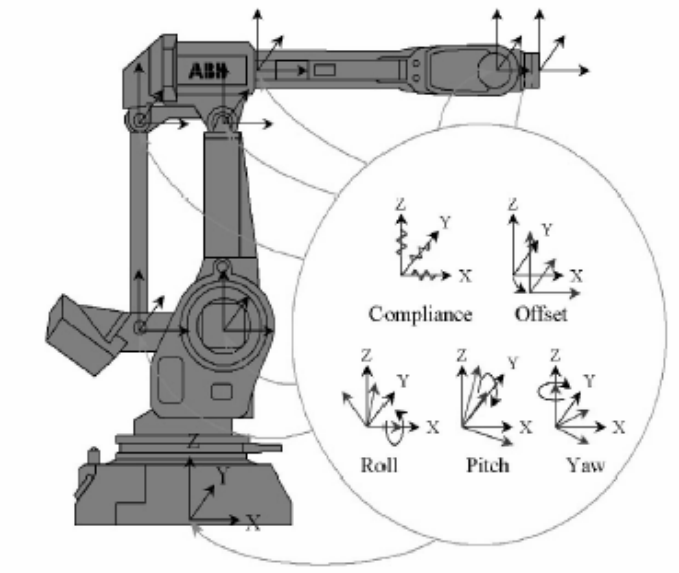
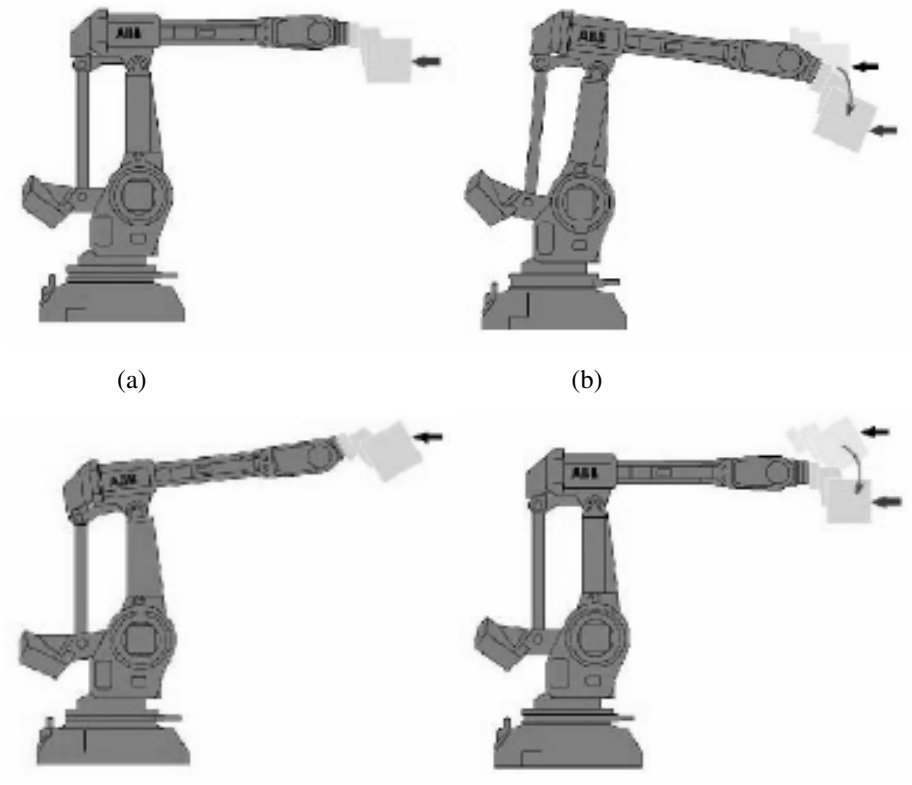


Figure 5.2 Several types of errors that can occur in each joint (ABB, 2004)

Both compliance and kinematical errors are compensated for with "fake targets". Knowing the deflection of the robot (i.e. deviation from ordered position), Absolute Accuracy can compensate by ordering the robot to a fake target.

Process of Absolute Accuracy Compensation method is seen in Figure 5.3. The aim of this process is reach the desired position with high accuracy (Figure 5.3 a). However the deflection which occurs by self weight of the robot and payload is the cause of leaves the desired end point (Figure 5.3.b). In that graphics the deflections are exaggerated. In order to get the desired position, Absulute Accuracy calculates a fake target. When you enter a desired position, the system recalculates it to a fake target that after the deflection will result in the desired position (Figure 5.3.c). The actual position will be the close as your desired position. The user will not notice the fake target or the deflection. The robot will behave as if it had no deflection (Figure 5.3.d).



(a) (b)

(c) (d)

Figure 5.3 Absolute accuracy compensation (ABB, 2004)

The most important point of the compensation method is how to determine the fake target. The position of the fake target affects the accuracy of the robot directly. The calibration procedure is explained in the following section.

5.1.2 Calibration Process

Each robot is calibrated with maximum load to ensure that the correct compensation parameters are detected (calibration at lower load might not result in a correct determination of the robot flexibility parameters.) The process runs the robot to 100 joint target poses and measures each corresponding measurement point coordinate. The list of poses and measurements are fed into the calibration core program and a robot compensation parameter configuration file is created. In Figure 5.4 graphical illustration and real application of absolute accuracy compensation are seen. Absolute accuracy compensation flowchart is shown in Figure 5.5.

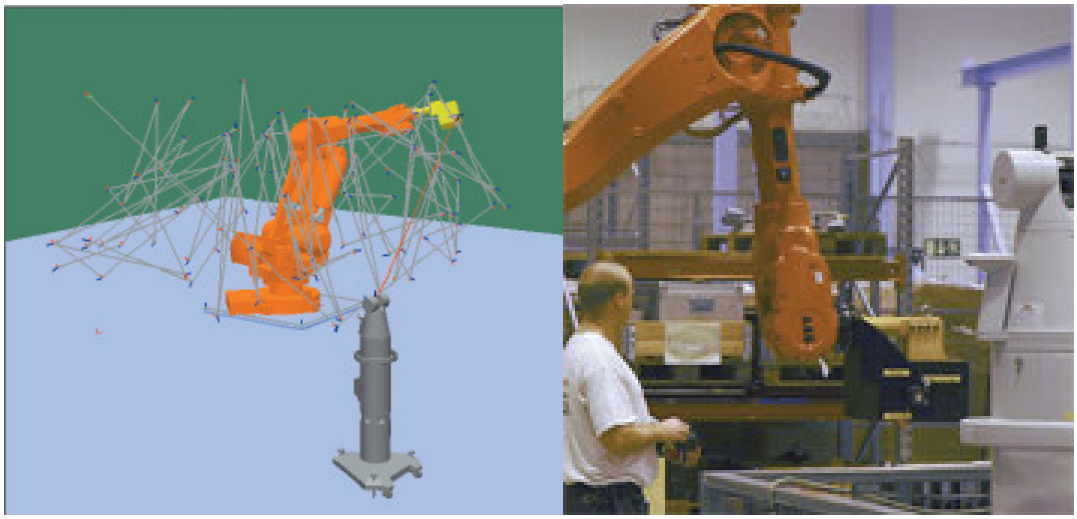


Figure 5.4 Graphical illustration and real application of absolute accuracy compensation (ABB, 2006)

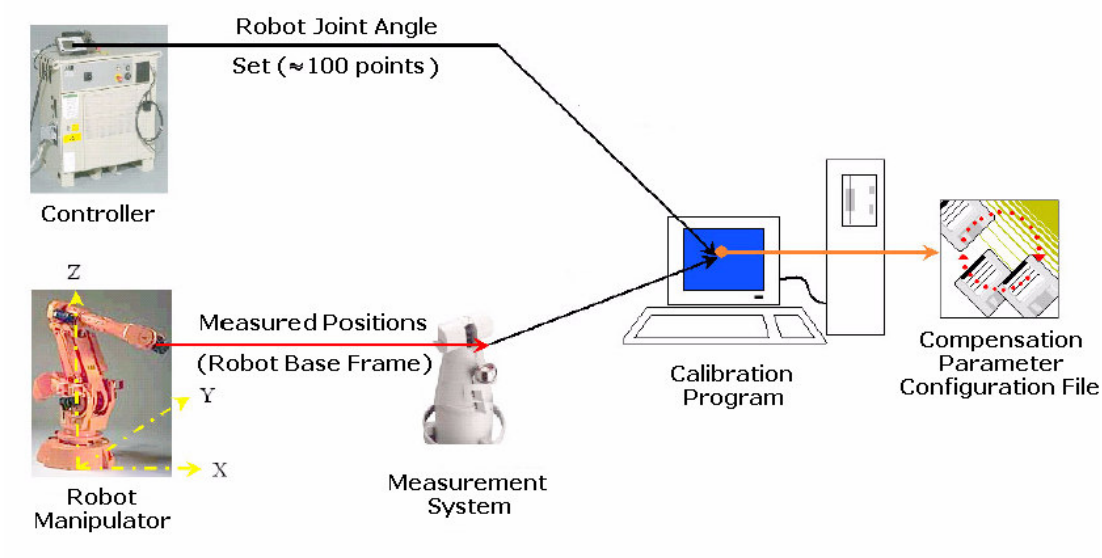


Figure 5.5 Absolute accuracy compensation flowcharts (ABB, 2004)

Laser interferometry measurement system is used in Accuracy Compensation procedures. These systems are too expensive because its' measurement range is too large. It can measure until 70-80 meter long distance. Measurement accuracy is decreased until 10 µm + 0.4 µm/m.

Having obtained the compensation parameters, these are loaded onto the controller and activated. The robot is then run to a set of 50 point target poses. Each

pose is measured and the deviation from nominal determined. Absolute deviations of the robots that are various types are less than 1 mm for the typically % 90 poses.

5.1.3 Disadvantages of Present Compensation Method

Similar compensation methods are used by different robot companies but they have some disadvantages. These are listed below:

1. Compensation parameters must be created for at least 100 points in the workspace of the each manufactured robot and according to these parameters verification must be done for 50 different points. This procedure requires long time and it is very onerous.
2. Laser interferometry measurement system requires investment at least 300.000 $. For that reason this measurement system is not profitable especially for small and medium size of companies.
3. Robot can be positioned at infinite point in workspace. Compensation on these points is calculated by using the results obtained from 100 measured point. This calculation is caused to some error.
4. Payload can change according to job. However in this compensation method payload is taken as maximum payload and compensation parameter configuration file is defined according to this condition. When the robot is working with different payload, accuracy compensation does not absolutely true.
5. Despite the high resolution measurement systems, the accuracy for the end point of the robot is approximately 1 mm. The accuracy may greater than 1 mm especially for the robots that have high payloads. Table 5.1 gives the success rates of the position accuracy compensation for ABB robots.
6. Deflections have remarkable values for even rigid industrial robots which have payload-self weight ratio between 1/40 and 1/10. Deflections are most important for Light Weight Robots (LWR) which is thought as the future of the robots. So this method of accuracy compensation will not fulfil the desired high level accuracy expectations.

Table 5.1 Position accuracy for ABB robot family (ABB, 2006)

	Position accuracy (Typical production data)		
Robot type	Average (mm)	% within 1 mm	Maximum (mm)
IRB 140	0.35	100%	0.80
IRB1400	0.45	100%	0.95
IRB 2400L IRB 2400/10kg and 16kg	0.45 0.30	95% 100%	1.15 0.70
IRB 4400	0.30	100%	0.70
IRB 6600-175kg/2.55m; 225kg/2.55m; 200kg/2.75m	0.55	90%	1.20
IRB 6600-175kg/2.80m; 125kg/3.20m	0.75	80%	1.45
IRB 6650S-200kg/3.00m; 125kg/3.50m			
IRB 7600-400kg/2.55m; 150kg/3.50m	0.65	85%	1.40
IRB 7600-500kg/2.30m	0.55	90%	1.20
IRB 7600-340/2.80			

5.2 Rigidity Workspace

The concept of Rigidity Workspace is introduced in this thesis. The Rigidity Workspaces are obtained for DEU-3X2-550 and DEU-S45-900 robot in terms of the end point deflections and natural frequencies.

5.2.1 A New Approach for Robot Workspace

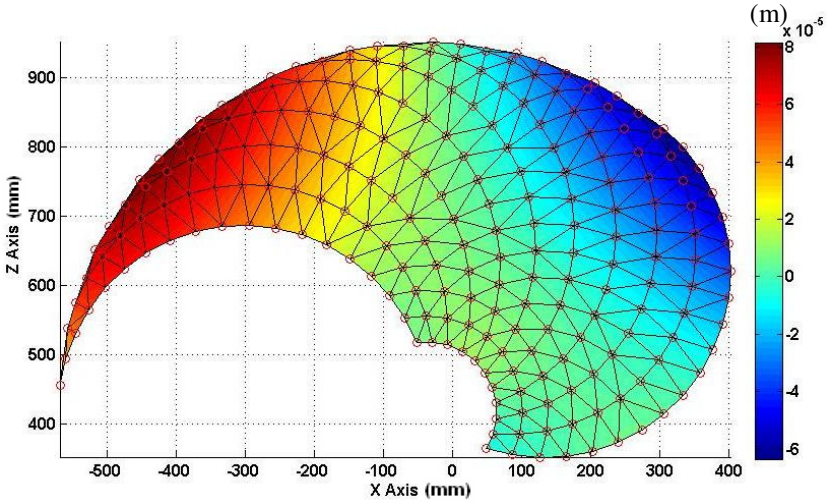
Workspace can be defined as the group of points to which the robot end point can reach. The workspace of a robot is limited by the dimensions of the robot arms and limitations on the freedom of joint axes. By another definition “existence or non existence of a kinematic solution defines the workspace of a given manipulator” (Craig, 1986). Generally, the concept of workspace is considered in terms of the kinematical properties of the robot. Unlike the other machines, the whole body of the robot can move the positions of its components may vary in the work space. Consequently, as the position of the robot changes, kinetic, static and modal properties also change completely. In this thesis, a new workspace definition “Rigidity Workspace” including position dependent end point deflections and position dependent modal properties is introduced. By the new workspace definition,

it is aimed to plan most proper trajectories that provide maximum accuracy for the performed job. Static and modal behaviors of the robot for different configurations can be studied by the aid of the new workspace concept.

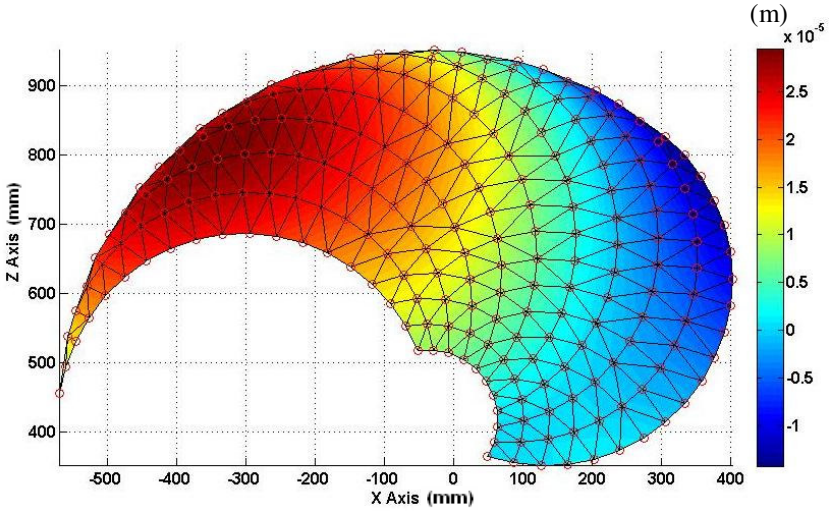
5.2.1.1 Rigidity Workspace of DEU-3X2-550 Robot

The Rigidity Workspace of the robot DEU-3X2-550 designed and manufactured in the scope of this thesis is explained in this section. Rigidity Workspace of this robot is obtained in terms of the end point deflections and the natural frequencies of the robot. In the derivation of the rigidity workspace, the ABAQUS program is used as the finite element solver.

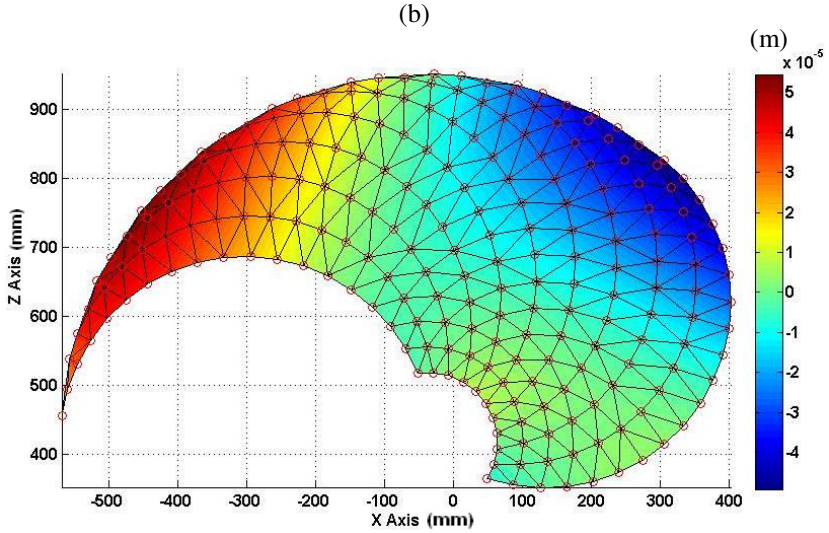
Firstly, 220 different points are defined in the workspace of the robot by using a code written in MATLAB. This code performs the kinematic calculations for the robot. The coordinates and the corresponding joint angles for these points are prepared as an input file. These points lie on x-z plane of the robot workspace. Whole workspace of the robot is obtained by revolving the x-z plane about the first axis of the robot. For that reason, numerical simulations are performed only the points on x-z plane and so the whole workspace of the robot is determined. All parts of the robot are modeled parametrically by using script code that is written in ABAQUS program and the robot assembly is created according to the joint angles which are read from the input file. All definitions required for the FEM analysis are done also in this script code. All details of the robot model and definitions about the solution set are changeable in the script code. Static and frequency analyses are performed successively for all the positions in the workspace by using the script code. The script code is given in the Appendix A2. Static analyses are performed for maximum payload and no payload cases. The static end point deflections can be calculated for other load values by linear interpolation. Figure 5.6, 5.7, 5.8 and 5.9 show the Rigidity Workspace in terms of the end point deflection at x, y, z direction and magnitudes for maximum and no payload condition and their difference, respectively.



(a)

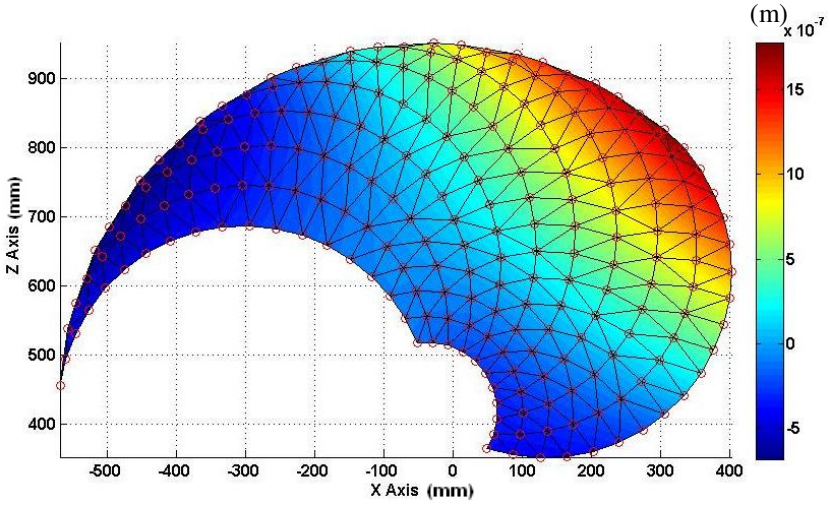


(b)

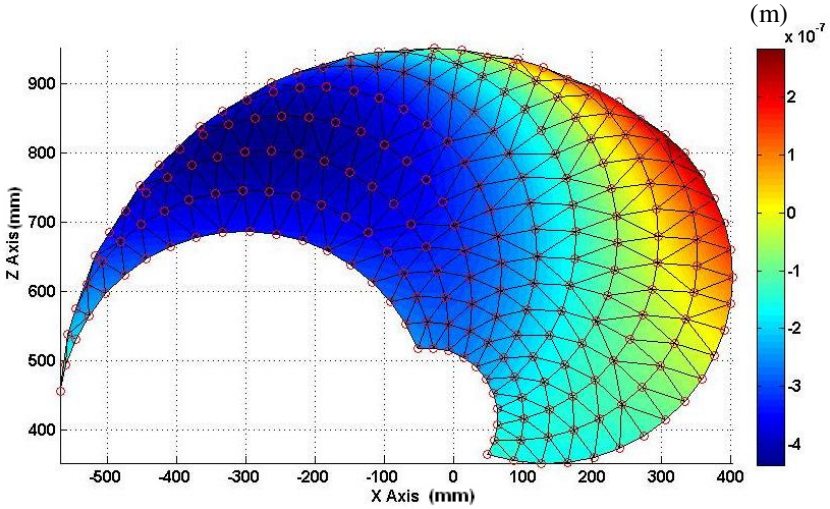


(c)

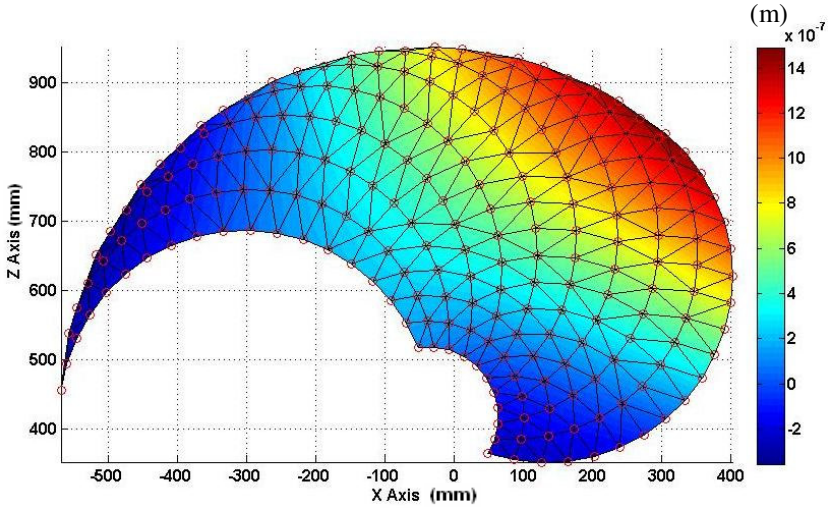
Figure 5.6 Rigidity Workspace in terms of the end point deflection on x direction for a) maximum payload, b) no payload, c) difference.



(a)

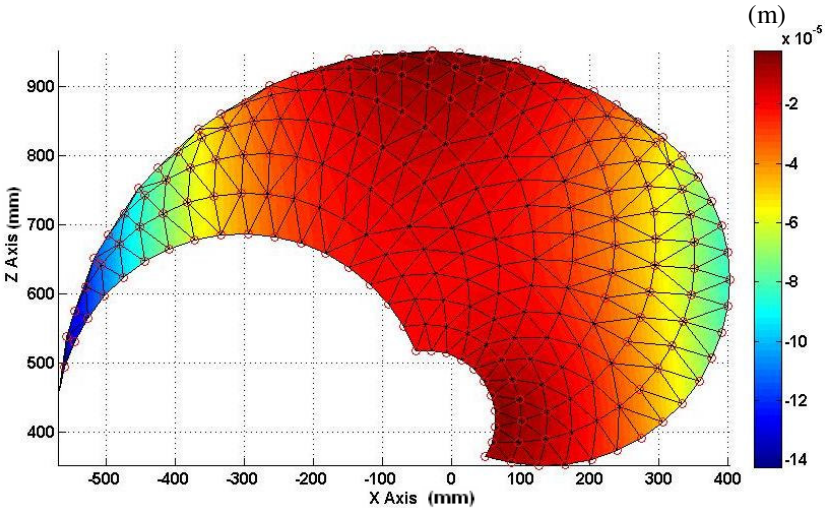


(b)

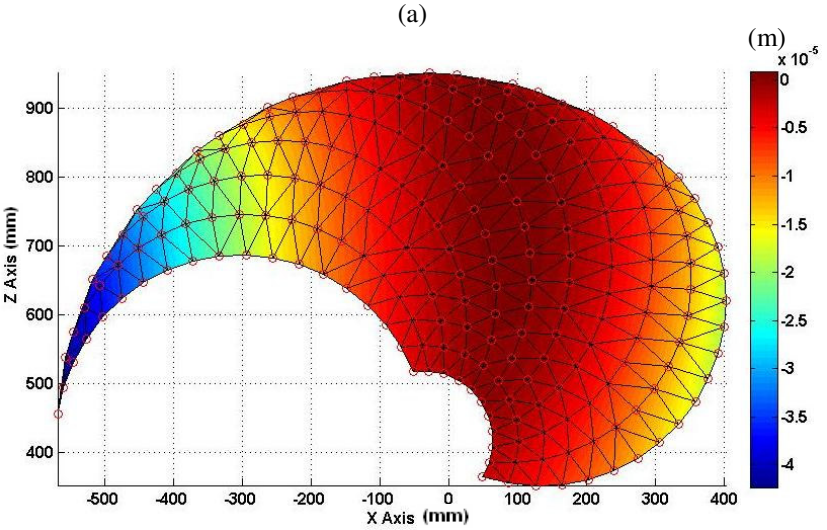


(c)

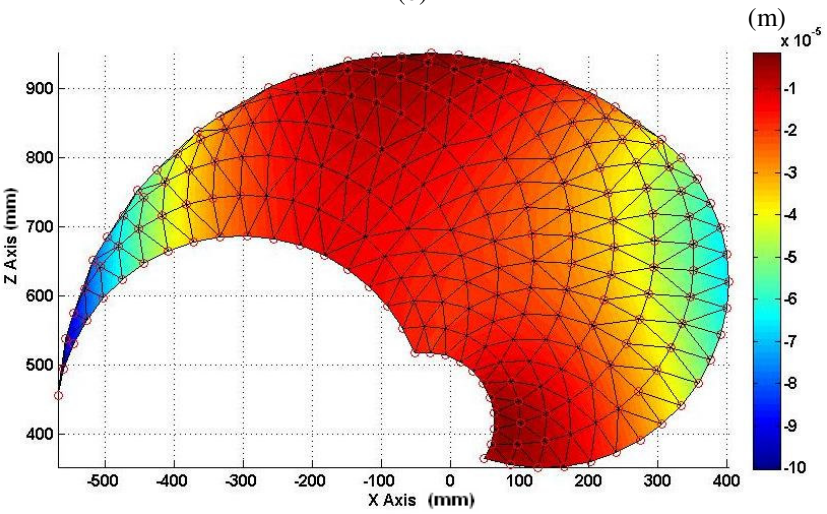
Figure 5.7 Rigidity Workspace in terms of the end point deflection on y direction for a) maximum payload, b) no payload, c) difference.



(a)

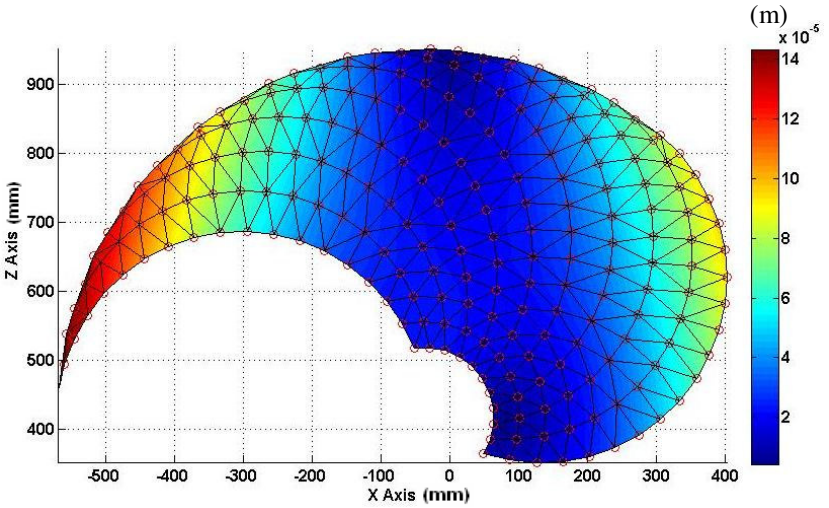


(b)

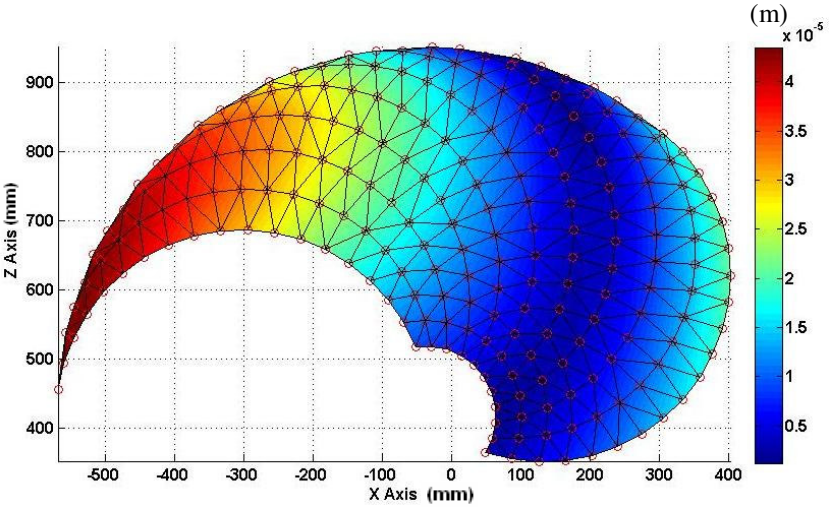


(c)

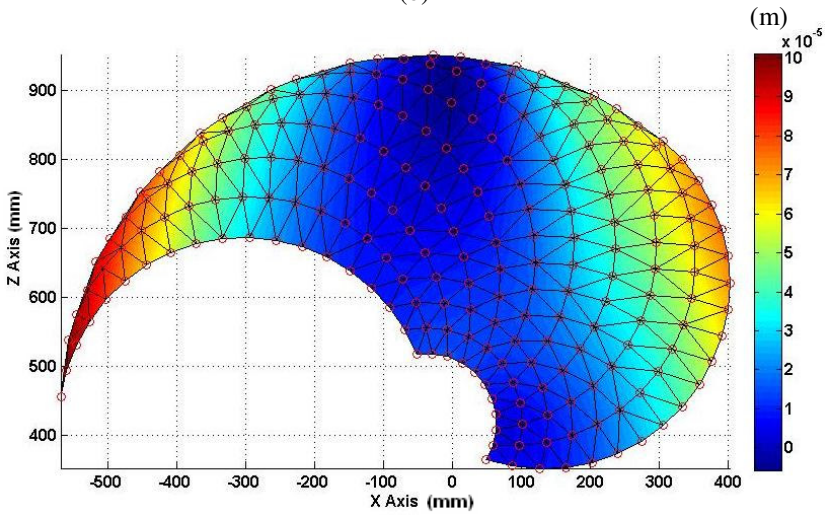
Figure 5.8 Rigidity Workspace in terms of the end point deflection in z direction for a) maximum payload, b) no pay load, c) difference.



(a)



(b)



(c)

Figure 5.9 Rigidity Workspace in terms of the end point deflection (magnitude) for a) maximum payload, b) no payload, c) difference

The graphical representation of the Rigidity Workspace is obtained by MATLAB. The workspace can be illustrated in 3-D. Figure 5.10 shows the 3-D view of the rigidity workspace. The vertical axis in this illustration is the magnitude of the end point deflection of the robot.

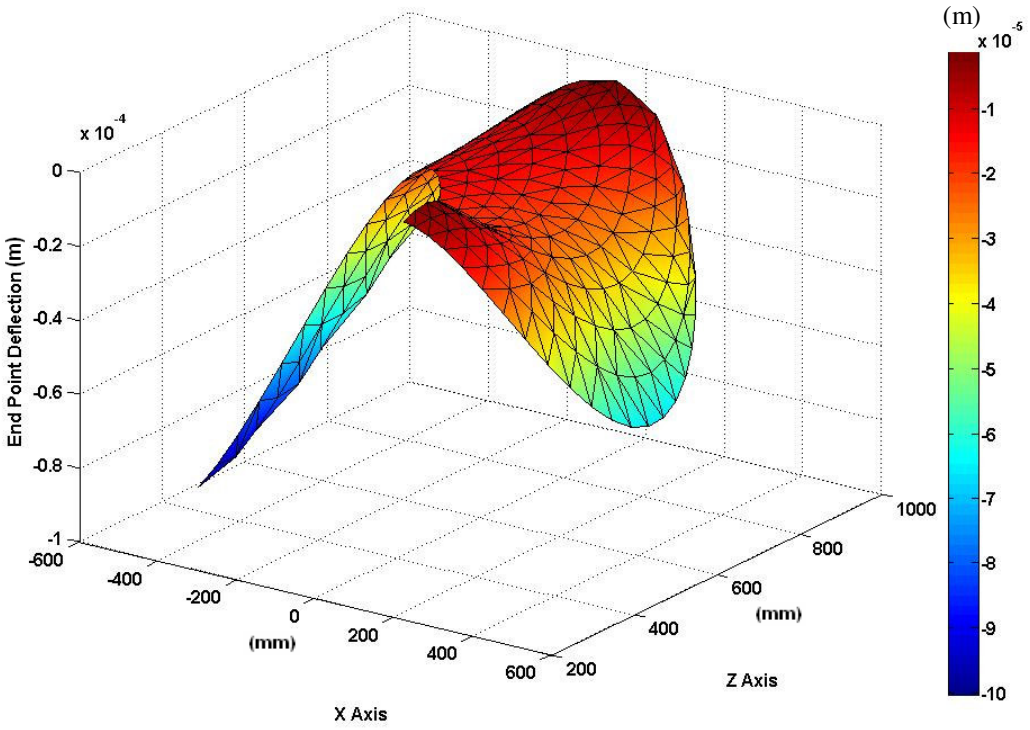
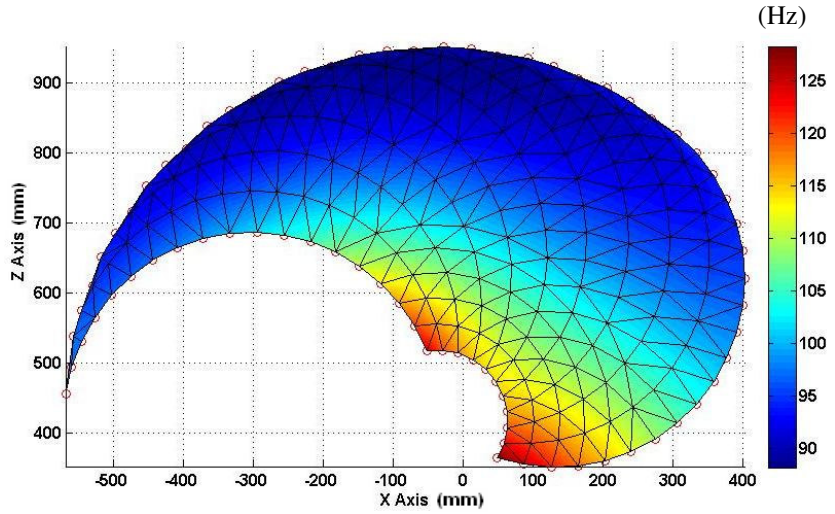
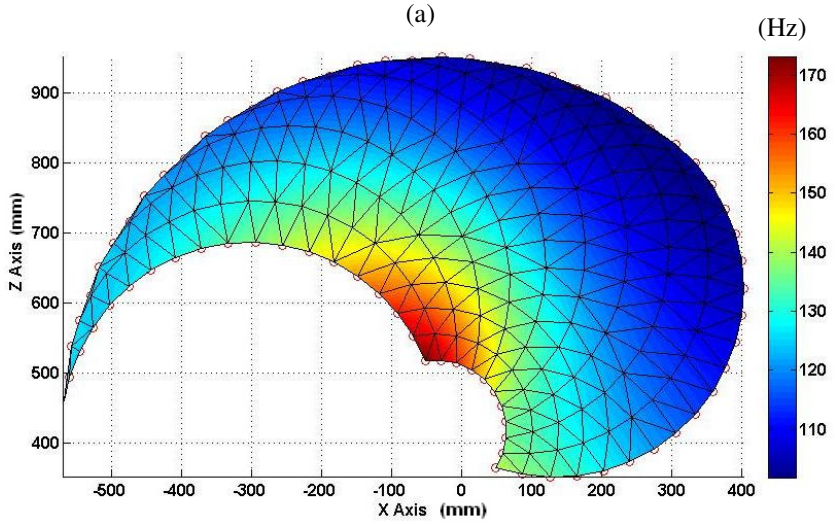


Figure 5.10 3D view of the Rigidity Workspace versus end point deflection

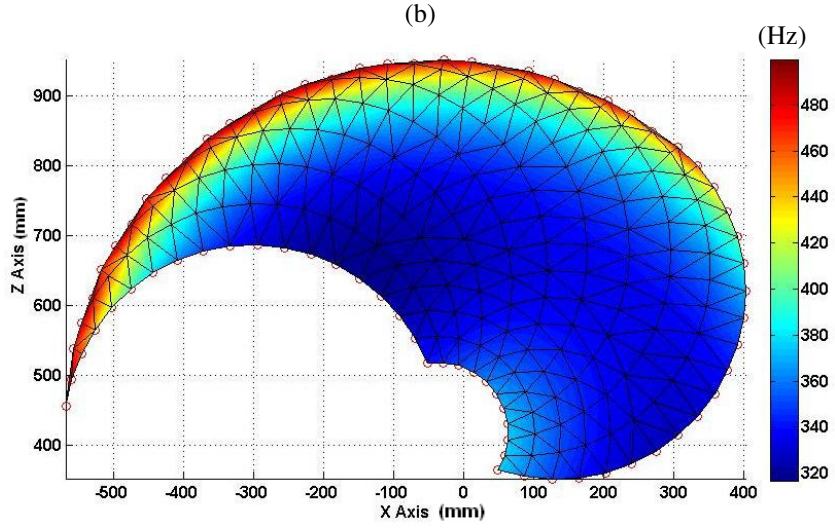
Natural frequencies of a structure are also the decisive measure of its rigidity. Natural frequencies of a robot manipulator change when the position of the robot changes. Vibrations which occur after starting and finishing the motion especially with high end point velocities affect the work of the robot. The acceleration and deceleration profiles can be controlled by the robot controller taking the natural frequencies into account according to position of the robot in the workspace to minimize the vibration levels. This subject becomes more important for the Light Weight Robots (LWR). The rigidity workspaces in terms of first, second and third natural frequencies are shown in Figure 5.11.



(a)



(b)



(c)

Figure 5.11 Rigidity Workspace versus (a) first, (b) second and (c) third natural frequencies

5.2.1.2 Rigidity Workspace of DEU-S45-900 Robot

In this section, the Rigidity Workspace is obtained for DEU-S45-900 SCARA robot. Eleven different points are defined in the workspace of the robot. The first arm is kept stationary and the second arm is rotated 150° with the angular increment of 15° while defining these points. Workspace of the robot is obtained by revolving these eleven points about the first axis of the robot. Static and frequency analyses are done for these eleven points to construct the rigidity workspace. Static analyses are performed for both maximum load and no load conditions.

The natural frequencies of the SCARA robot had been calculated in chapter 3 by Cosmos Works software. In this section, the rotor and the stator of the driving motor is modeled separately to see how the rigidity of the motors affects the modal behavior of the robot. Rolling element bearings used in the Harmonic Drive motors which are the main driving units of robot are taken into consideration for defining the boundary conditions. Interaction is defined between the rotor and the stator of the motor for the location that the axial load is carried by the contact elements. By the aid of the contact elements, the load can be transferred from one part to another like real machines. Workspace of the robot is considered only on the x-y plane because there is no axis that moves along the z direction. Kinematic workspace of DEU-S45-900 and positions that are used in analyses are seen in Figure 5.12. M position is maximum reach position and R position is minimum reach position. The results of static and modal analyses for eleven points are listed in Table 5.2.

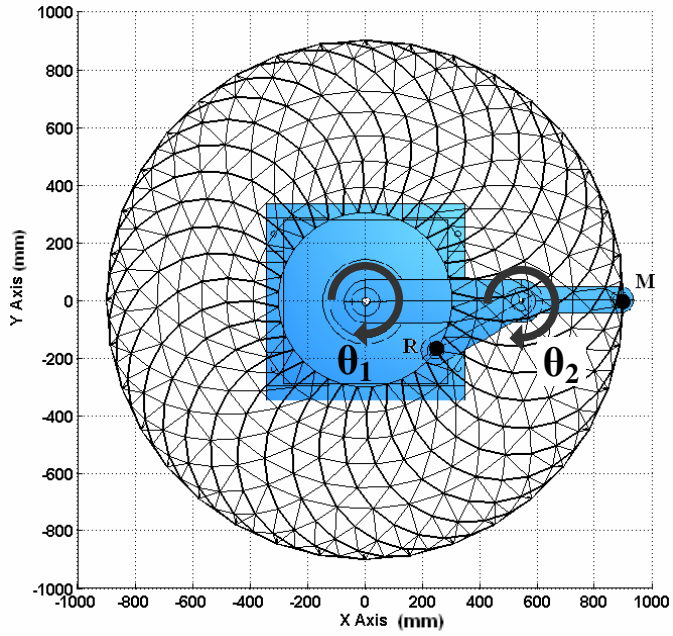


Figure 5.12 Kinematic workspace of DEU-S45-900 and positions that are used in the analyses

Table 5.2 Results of static and frequency analyses for eleven points in workspace of SCARA robot

Position (Teta2) Degree	Deflection mm (45 kg)	Deflection mm (No Load)	Deflection mm (Difference)	Frequency 1 (Hz)	Frequency 2 (Hz)	Frequency 3 (Hz)
0 (M)	-0.546888	-0.166665	-0.380223	14.828	44.209	71.876
15	-0.539932	-0.164425	-0.375507	14.876	44.285	71.03
30	-0.519958	-0.157922	-0.362036	15.029	44.539	69.19
45	-0.488192	-0.147566	-0.340626	15.272	44.896	66.763
60	-0.446834	-0.134117	-0.312717	15.62	45.345	64.763
75	-0.398805	-0.118539	-0.280266	16.019	45.695	63.105
90	-0.347312	-0.101851	-0.245461	16.5	46.21	62.477
105	-0.295902	-0.0852071	-0.2106949	16.994	46.717	62.046
120	-0.248303	-0.0697384	-0.1785646	17.476	47.62	61.769
135	-0.207645	-0.0564908	-0.1511542	17.901	48.832	61.049
150 (R)	-0.176726	-0.046391	-0.130335	18.234	50.52	59.889

Figure 5.13 shows the Rigidity Workspace of DEU-S45-900 robot in terms of the end point deflection along z direction for maximum load, no load and their differences.

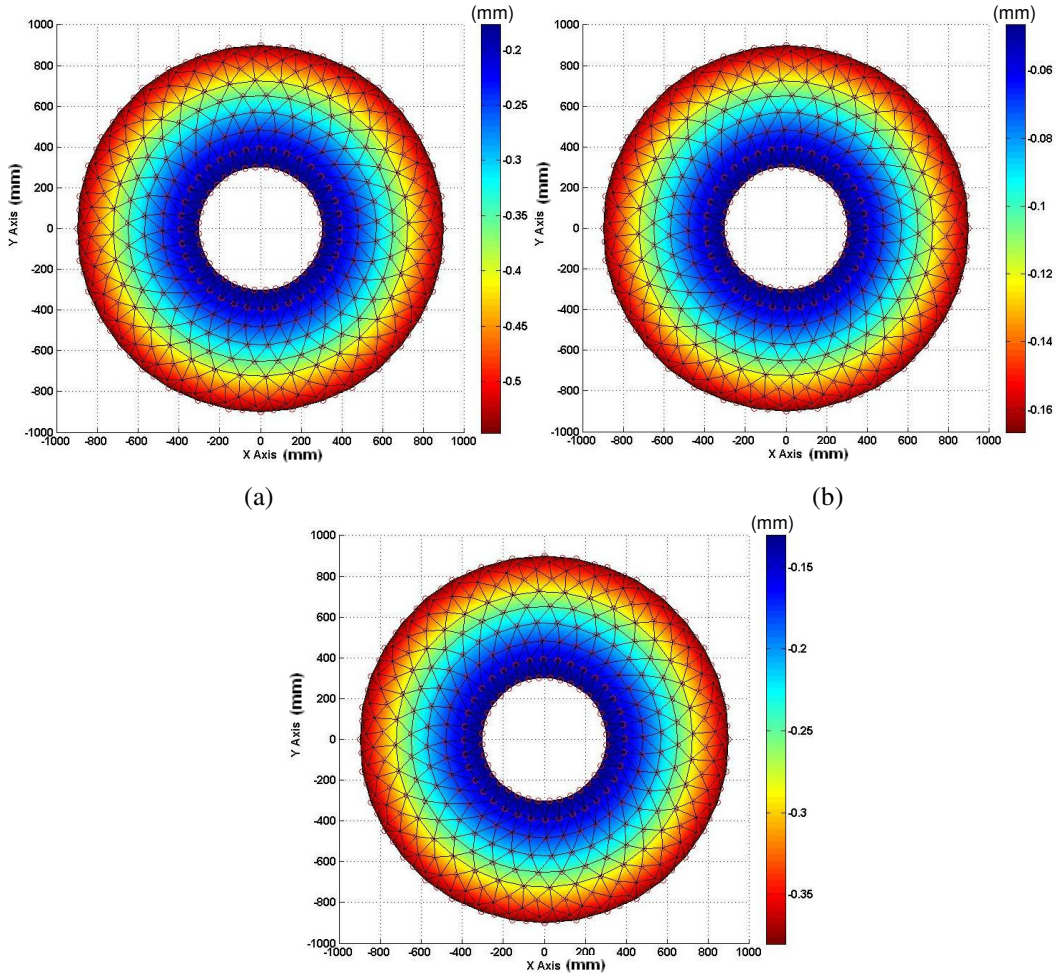
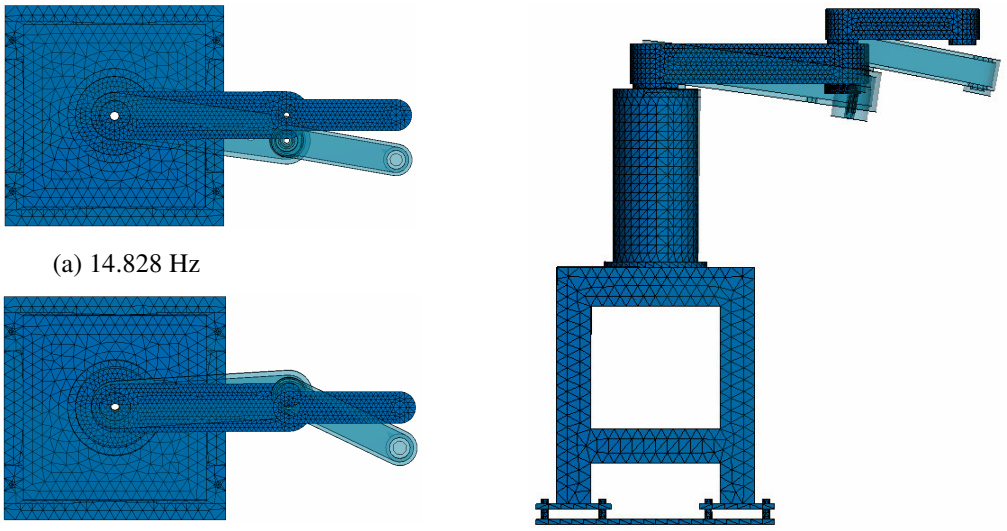


Figure 5.13 Rigidity Workspace of DEU-S45-900 versus end point deflection on z direction for a) with maximum payload, b) without payload, c) difference.

Figure 5.15 shows the first three fundamental frequencies of the SCARA robot. The effect of the interaction in the motor components shows its effect on the natural frequencies and the associated mode shapes. The natural frequencies and mode shapes given in chapter 3 are calculated for the case in which the surface of the rotor and stator of the motor are rigidly connected. But, this assumption does not reflect the real situation and the natural frequencies and the mode shapes are far from the real values. Figure 5.15 shows the Rigidity Workspace in terms of the first, second and third natural frequencies.

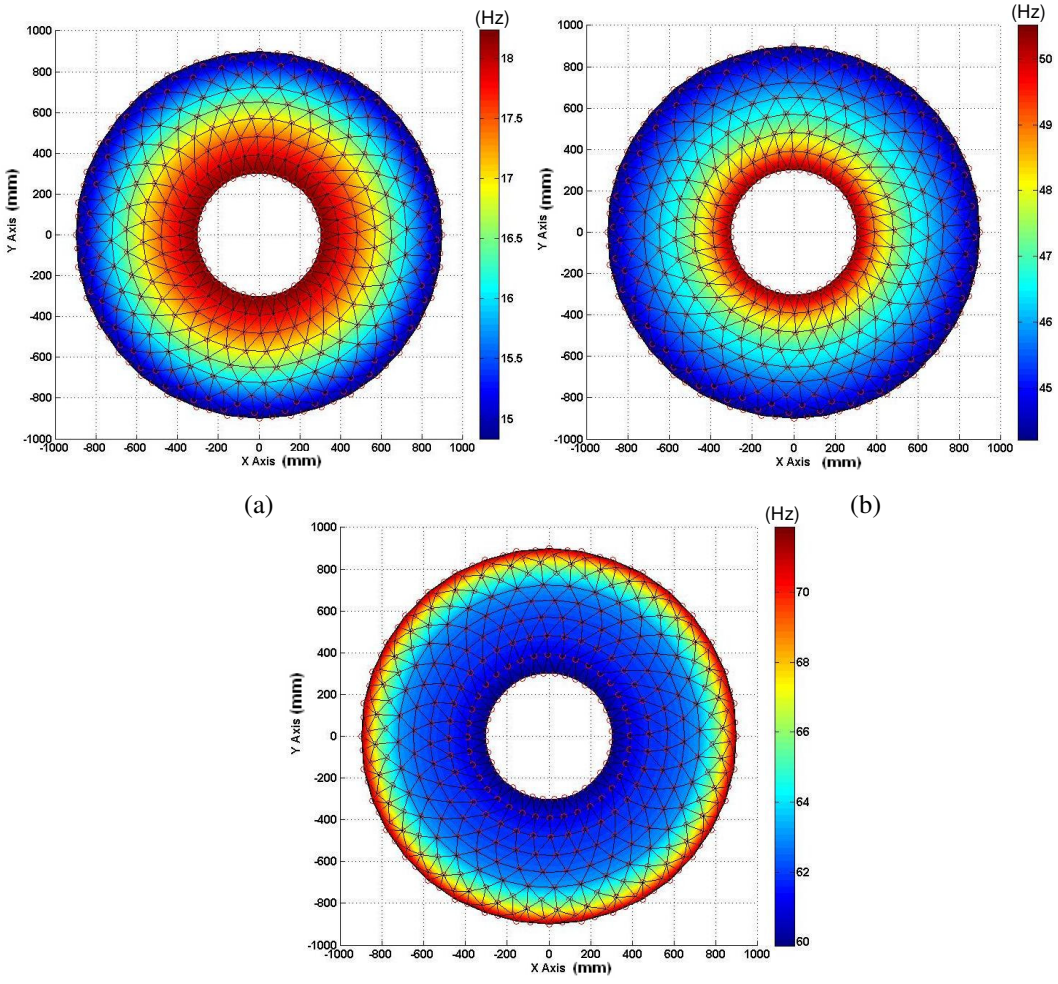


(a) 14.828 Hz

(c) 71.876 Hz

(b) 44.209 Hz

Figure 5.14 Mode shapes of DEU-S45-900 on M position for a) first, b) second, c) third natural frequencies



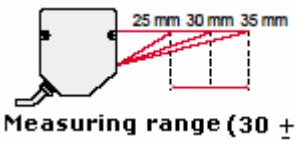
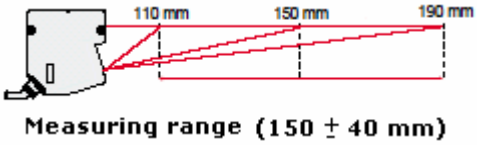
(a) (b)

(c)

Figure 5.15 Rigidity Workspace of DEU-S45-900 in terms of a) first, b) second, c) third natural frequencies.

The simulated results are verified by the experimental measurements. Static deflections and natural frequencies of the real robot system are measured via laser displacement sensors. First of all, the static displacements at the end point of the robot are measured for the positions M and R. Then the natural frequencies for two fundamental mode shapes are measured for the same positions. The fundamental modes of the robot manipulator are excited by using a standard hammer. Two Keyence LK-G series laser displacement sensor are used for measurements. The technical specifications of the sensors are given in Table 5.3 (Keyence, 2007).

Table 5.3 Technical specifications of laser displacement sensors (Keyence, 2007)

	Keyence Laser Displacement Sensor	
Model	LK-G37	LK-G157
Measurement Range		
Resolution	0.05 µm	0.5µm
Spot Size	30x850 µm	120x1700 µm
Sampling Frequency	20/50/100/200/500/1000 µs (Selectable from 6 levels)	

The LK-G37 laser displacement sensor is located as shown in Figure 5.16 for static displacement measurements. Up to 5 weights, each one is 9 kg, can be attached to the end point of the robot via an apparatus mounted onto the robot. By this way, the static displacement of the end point of the robot is measured for the case of maximum pay load. The real view of the experimental system for the positions M and R is shown in Figure 5.16. The static deflection measurement is performed step by step for increasing pay load in order to verify the linearity of the static deflections at the en point of the robot. The static deflection results given in Figures 5.17 and 5.18 show that the static deflection values versus the applied weight has linear characteristic.

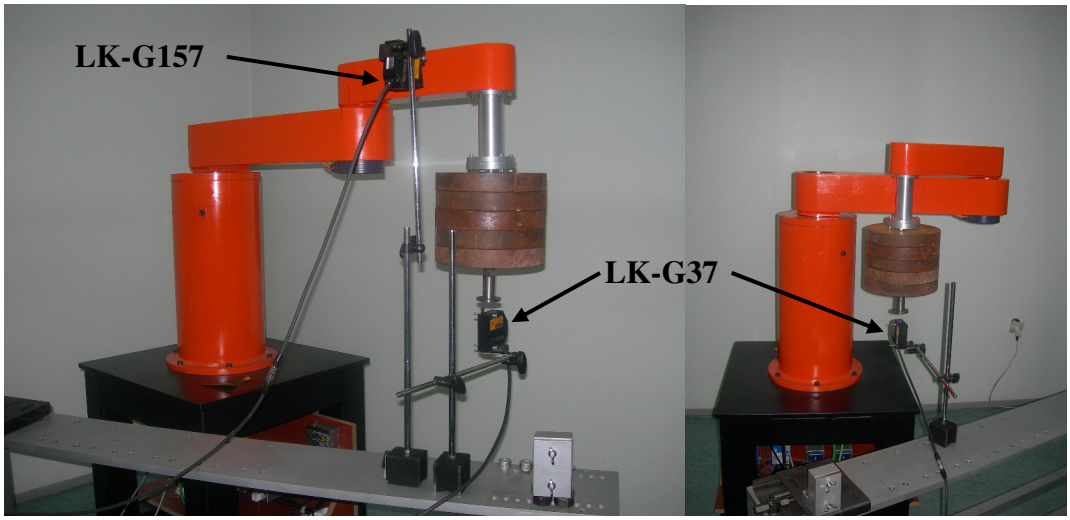


Figure 5.16 End point deflection measurement system for maximum load for positions M and R

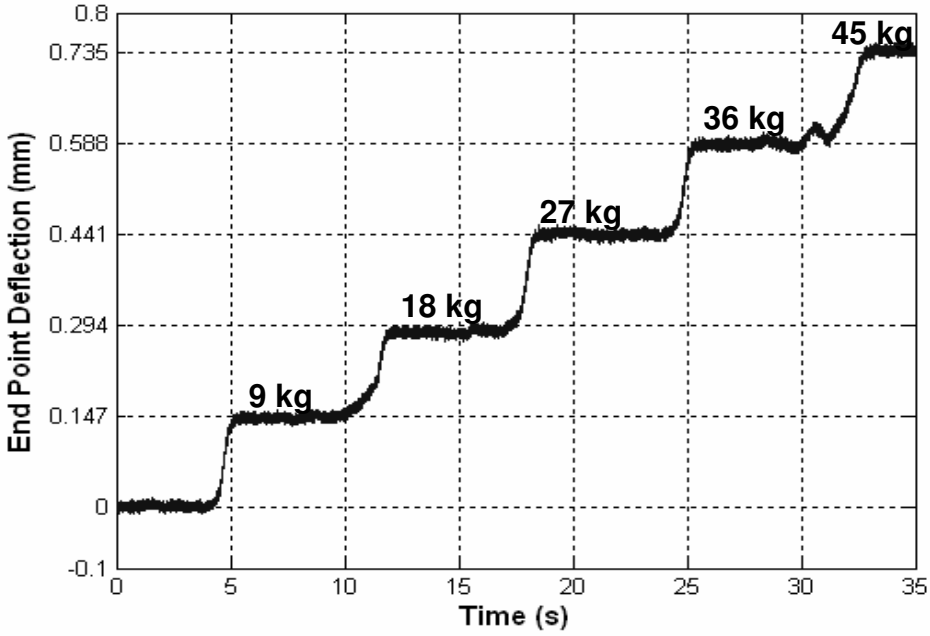


Figure 5.17 End point deflections of DEU-S45-900 for M position for different payloads.

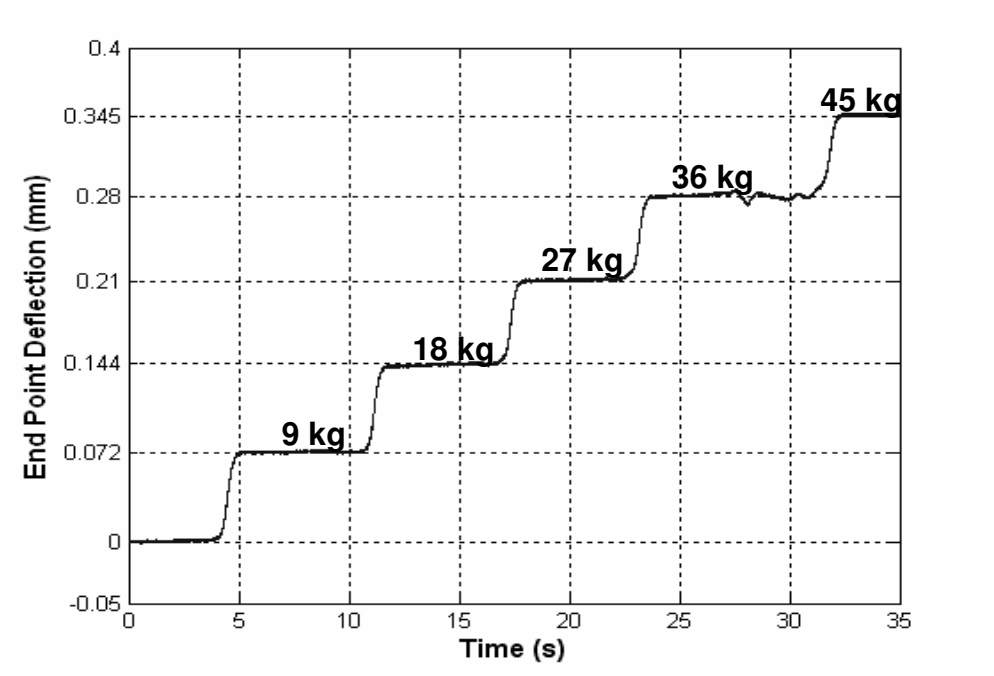


Figure 5.18 End point deflection of DEU-S45-900 for R position for different payloads.

Both laser displacement sensors are used for natural frequency measurement. The LK-G157 sensor is used to capture the mode shape parallel in x-z plane (parallel to the ground) and the LK-G37 sensor is used to capture the mode shape in x-y plane (vertical plane). A standard hammer having plastic head is used to excite the two vibration modes. The experimental system is shown in Figure 5.19.

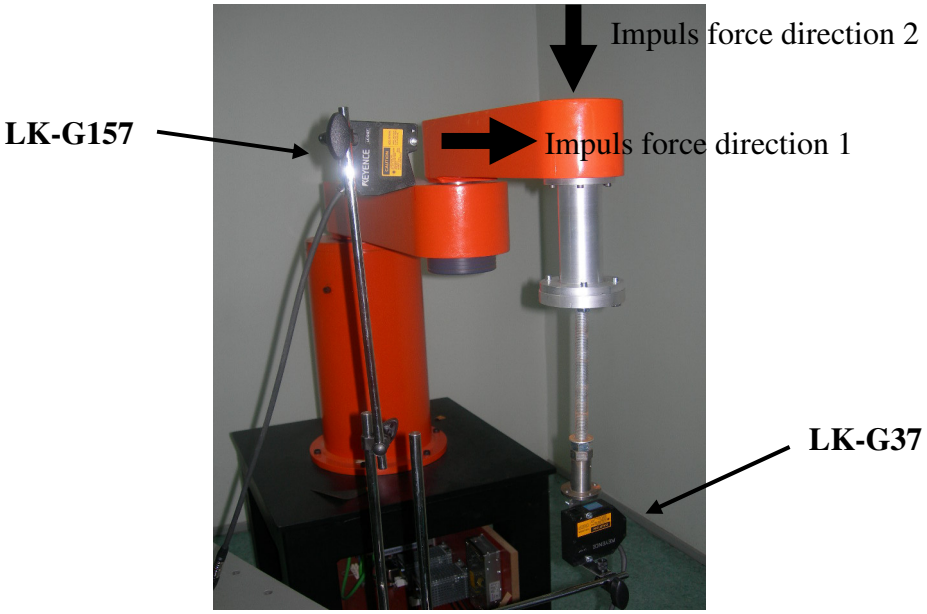


Figure 5.19 Natural frequency measurement system for the position M

The vibration responses and their frequency content obtained by Fast Fourier Transform are shown in Figures 5.20 and 5.21.

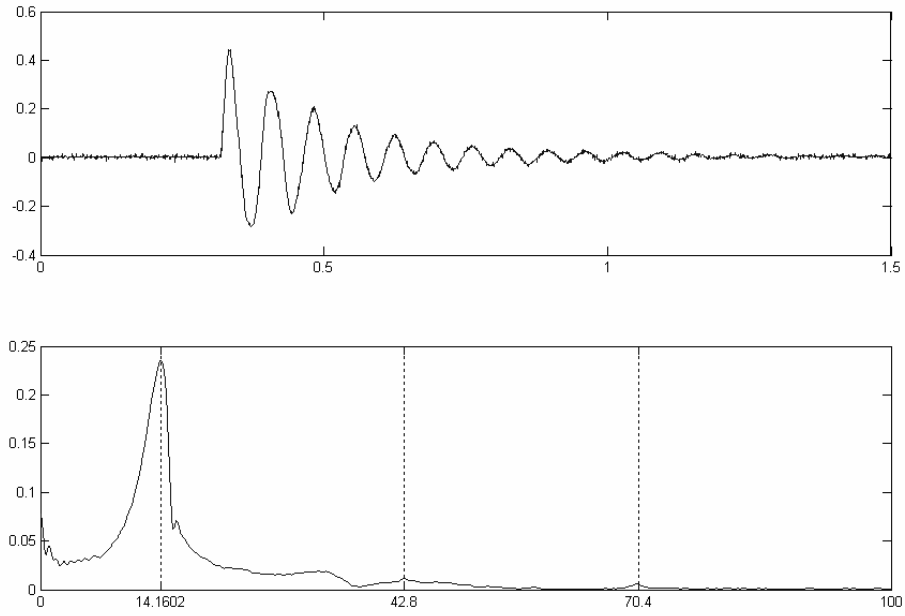


Figure 5.20 Free vibration response obtained from LK-G157 (direction 1) and its frequency spectrum for position M

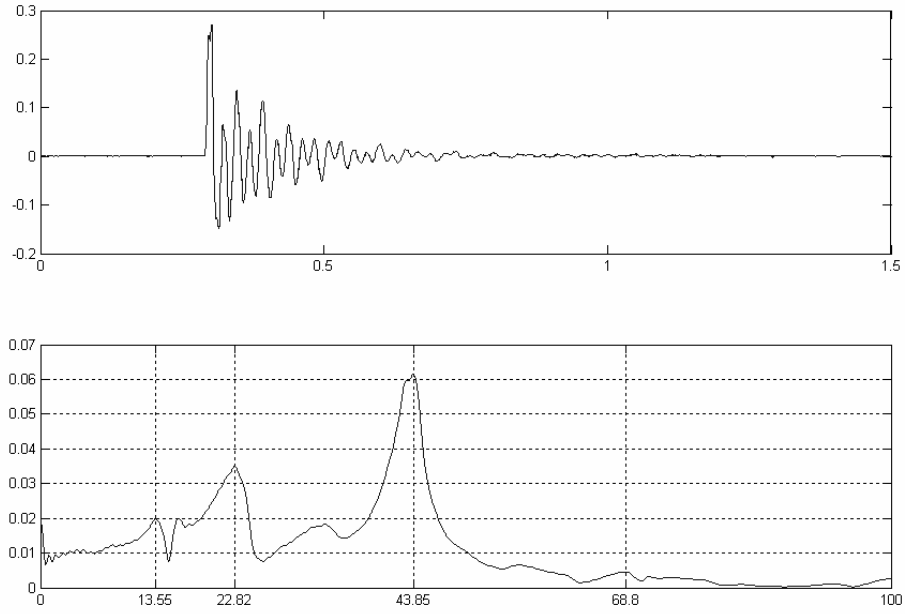


Figure 5.21 Free vibration response obtained from LK-G37 (direction 2) and its frequency spectrum for position M

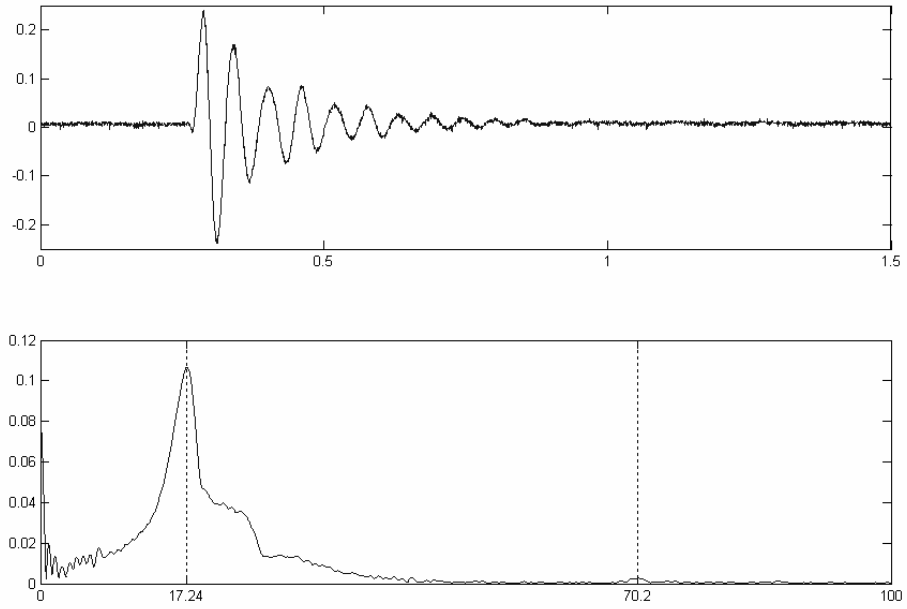


Figure 5.22 Free vibration response obtained from LK-G157 (direction 1) and its frequency spectrum for position R

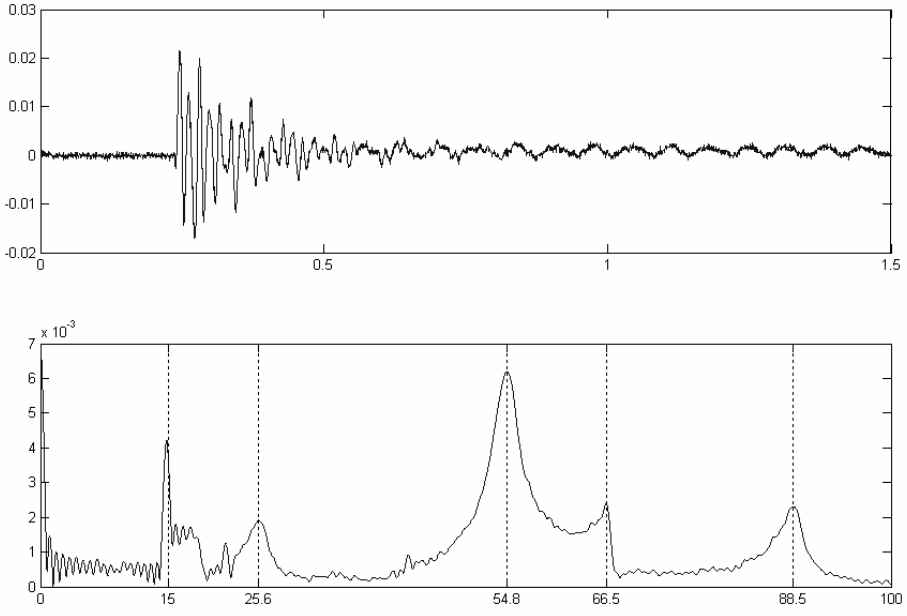


Figure 5.23 Free vibration response obtained from LK-G37 (direction 2) and its frequency spectrum for position R.

The simulated and experimental results are given in tabular form in Table 5.4. As shown from the table the simulated and experimental natural frequencies are well-matched. There are some considerable discrepancies between the numerical and experimental deflection values. These discrepancies are attributed to the assumptions made in the finite element modeling.

Table 5.4 Comparison of numerical and experimental results of DEU-S45-900

Position	**Comparison Value**	**FEM Results**	**Experimental Results**
M	Static Deflection (mm)	0.380223	0.735
	First Natural Frequency (Hz)	14.828	14.1602
	Second Natural Frequency (Hz)	44.209	43.85
	Third Natural Frequency (Hz)	71.876	70.04
R	Static Deflection (mm)	0.130335	0.345
	First Natural Frequency (Hz)	18.234	17.24
	Second Natural Frequency (Hz)	50.52	54.8
	Third Natural Frequency (Hz)	59.889	66.5

5.2.2 New Proposal for Static Compensation

A new compensation method can be developed by using the Rigidity Workspace proposed in this thesis. The fake positions can be determined by using the deflection values given in the Rigidity Workspace and then the compensation operation will be performed according to these positions for every payload.

The Rigidity Workspace should be obtained during the design stage. So, the static and modal behavior of the robot manipulator can be controlled before the production of the robot. The most critical point in the calculation of the Rigidity Workspace is how to model the actuators, rolling element bearings in the joints, reduction gears and the transmission mechanisms. These components have relatively complex structures and it is very difficult to create their detailed finite element models. The

numerical results obtained with the simplified models of these components produce some discrepancies. The simplified models of these components having the same rigidities with the exact model may be developed by their manufacturers for the practical usage of the engineers. But the robot part manufacturers have not supplied such data yet.

It is obvious that there are always some differences between the exact and simulated models of robot manipulators. The variations in the production tolerances and the improper simplifications in the equipments of the manipulators are the main reason for these differences. On the other hand, even for the same part, there are no two parts which have exactly the same properties. So the static and dynamic properties of the manufactured robots may vary. The calibration of a robot can be done by comparing the results obtained from the Rigidity Workspace and the measurements performed on the real system. It is thought that it will be sufficient to perform the calibration operation for a few specific positions of the robot manipulator.

CHAPTER SIX
CONCLUSIONS

The optimal design of a robot manipulator is a hard task and requires an advanced design skill as well as comprehensive knowledge of mechanics and electronics. It is an inevitable requirement for an industrial robot to work with high precision even under with high working speeds. In addition to being high precision and having a high working speed, the workspace of the robot is desired to be as large as possible. Therefore, a lot of parameters must be taken into account simultaneously in robot design. The modern CAE procedures, in which the finite element method is employed are the powerfull tools for robot designers.

In this thesis, an integrated design sceme for robot design is proposed. The solid modeling programs employed in the proposed design scheme are capable of performing parametric solid modeling. During the design process, different structural modifications are made according to the results of various engineering analyses. The parametric modeling provides easiness, rapidity and effectiveness in this time consuming design process. In the scope of this thesis, the proposed design scheme is applied to three different robot manipulators. Kinematic, kinetic, static, dynamic and modal analyses are performed for the robots. The design loop is run and improvements are made until the predefined design goals have been satisfied for the robots considered. Three robot manipulators are manufactured at the end of the design process, in which two of them are assembled and currently are ready for operation. The efficiency of the integrated design scheme is shown by three succesful aplications.

Rapid movements of the robot manipulators are necessary to reduce the production times. On the other hand, rapid movement means acceleration and deceleration that cause considerable inertial forces. The importance of the trajectory selection and its effect on the dynamic response is shown by the experimental strain measurements and numerical strain and stress analyses performed for an industrial

robot having six-degrees of freedom. The dynamic strain and stress values for an industrial robot can be controlled via proper selection of its trajectory.

The most important problem encountered in the offline programming of industrial robots is their insufficient accuracies. The main parameters which affect the acccuracy are the flexibility of the robot arms, production tolerances and the joint flexibilities. Different compensation techniques are developed in order to overcome this accuracy problem encountered especially for the jobs which require high precision. Today, the most preferable method is the laser interferomety in which the end point deflections for every manufactured robot are measured for different configurations in the workspace of the robot. A new concept "Rigidity Workspace" which will contribute to the accuracy compensation is introduced in this thesis. The Rigidity Workspace reflects the static end point deflections and modal behaviors of the robot manipulator for every end point position within its kinematic workspace. The static end point deflections and modal behaviours of the robot manipulator for numerous points in the workspace are calculated by ABAQUS via a script code developed in this study and the Rigidity Workspaces are illustrated by contour graphs in terms of static deflection and the fundamental frequencies. The numerical results presented in this study via rigidity workspace concept exhibit the structural behavior of the robot manipulators being considered. The structural behavior of the robot manipulators can be used to compensate the end point deflections from the real target. The numerical results are compared with the results obtained from experimental measurements carried out on DEU-S45-900 robot manipulator. As a result of this comparison, it is seen that the proper definition of the joint flexibilities has great influence on the numerical results. The manufacturers of robot equipments such as motors and bearings should supply the CAD models of these parts with the same rigidities in order to create the accurate finite element models for precise numerical analyses. Therefore, static and dynamic behavior of the manipulator can be predicted through the numerical analyses. The experimental results presented in this thesis imply that the accurate models for the robot manipulators considered can be established by using today's CAE tools.

The future works in order to extend this study can be summarized as follows,

- The Rigidity Workspace concept can be adopted to be employed in joint torques as “Kinetic Workspace”. The Kinetic Workspace can be used for the proper selection of the actuators of a robot manipulator.
- A new method can be developed in order to obtain the equivalent CAD models of the elements used in the robot manipulators such as motors, reduction gears, bearings and transmission systems. The validity of the developed method should be tested by experiments.

REFERENCES

ABAQUS[1] 6.5, (2004) Analysis user's manual. *ABAQUS version 6.5 documentation.*

ABAQUS[2] 6.5, (2004) Theory manual. *ABAQUS version 6.5 documentation.*

ABB 2004 Automation Technologies AB Robotics, (2004). Application manual motion performance revision A.

ABB Absolute Accuracy paper. (2006).http://www.jandmtech.com/PoonaAutomation /IRB%206400RF%20R4%20tryck.pdf

ABB IRB 1400 Data sheet, December 2007, http://library.abb.com/GLOBAL/SCOT/SCOT241.nsf/VerityDisplay/48DBA2E3FB893ED7482573D7002F4D0A/$File/IRB1410_EN_hi.pdf

Abderrahim, M. & Whittaker, A.R. (2000). Kinematic model identification of industrial manipulators. *Robotics and Computer Integrated Manufacturing, 16,* 1-8.

Albu-Schaffer, A., Haddadin, S., Ott, C., Stemmer, A., Wimböck, T. & Hirzinger, G. (2007). The DLR lightweight robot: design and control concepts for robots in human environments. *Industrial Robot, 34* (5), 376-385.

Alici, G. & Shirinzadeh, B. (2005). A systematic technique to estimate positioning errors for robot accuracy improvement using laser interferometry based sensing. *Mechanisn and Machine Theory, 40,* 879-906.

Bergan, P.B., Horrigmoe, G., Krakeland, B. & Soreide, T.H. (1978). Solution techniques for non-linear finite element problems. *International Journal for Numerical Methods in Engineering*, *12*, 1677–1696.

Bhatia, P., Thirunarayanan, J. & Dave, N. (1998). An expert system-based design of SCARA robot. *Expert Systems with Applications 15,* 99-109

Chin, J.-H. & Lin, S.-T. (1997). The path precompensation method for flexible arm robot. *Robotics & Computer Integrated Manufacturing, 13* (3), 203-215.

Clark, S. & Lin, Y.J. (2007). CAD tools integration for robot kinematics design assurance with case studies on PUMA robots. *Industrial Robot, 34* (3), 240-248.

Craig, J.J. (1986). *Introduction to robotics mechanics & control.* (1th ed.). United States of America: Addison-Wesley.

Drouet, P., Dubowsky, S. & Mavroidis, C. (1998). Compensation of geometric and elastic deflection errors in large manipulators based on experimental measurements: application to a high accuracy medical manipulator. *6th International Symposium on Advances in Robot Kinematics, Austria.*

Dwivedy, S.K. & Eberhard, P. (2006). Dynamic analysis of flexible manipulators, a literature review. *Mechanism and Machine Theory, 41,* 749-777.

Feliu, V. & Ramos, F. (2005). Strain gauge based control of single-link flexible very lightweight robots robust to payload changes. *Mechatronics, 15,* 547-571.

Harmonic Drive AG (2006). Precision in Motion Catalogue 2005/2006. In *CHA Hollow Shaft Actuators* (256-283).

Hashimoto, M., & Kiyosawa, Y. (1998). Experimental study on torque control using harmonic drive built-in torque sensors. *Journal of Robotic Systems, 15* (8), 435-445.

Hibbitt, H.D. & Karlsson, B.I. (1979). Analysis of pipe whip. *EPRI, Report NP,* 1208.

Jang, J.H., Kim, S.H. & Kwak, Y.K. (2001). Calibration of geometric and non-geometric errors of an industrial robot. *Robotica, 19,* 311-321.

Karagülle, H. & Malgaca, L. (2004). Analysis of end point vibrations of a two-link manipulator by integrated CAD/CAE procedures. *Finite Elements in Analysis and Design, 40,* 2049-2061

Keyence LK-G series technical data sheet, 2007, http://www.keyence.co.uk/products/vision/laser/lkg/lkg.php

Lee, E. & Mavroidis, C. (2004). Geometric design of spatial PRR manipulators. *Mechanism and Machine Theory, 39,* 395-408.

Lucchetta, G., Bariani, P.F. & Knight, W.A. (2005). Integrated design analysis for product simplification. *CIRP Annals-Manufacturing Technology, 54* (1), 147-150.

Mir-Nasiri, N. (2004). Design, modelling and control of four-axis parallel robotic arm for assembly operations. *Assembly Automation, 24* (4), 365-369.

Mitsubishi Electric, (2005). *Sevo amplifiers and servo motors melservo technical catalogue*

Morozov, A. & Angeles, J. (2007). The mechanical design of a novel Schönflies-motion generator. *Robotics and Computer-Integrated Manufacturing, 23,* 82-93.

Mrozek, Z. (2003). Computer aided design of mechatronic systems. *International Journal of Applied Mathematics and Computer Science, 13* (2), 255-267.

Myung, S. & Han, S. (2001). Knowledge-based parametric design of mechanical products based on configuration design method. *Expert Systems with Applications, 21,* 99-107

N-Nagy, F. & Siegler, A. (1987). *Engineering foundations of robotics.* (1th ed.). UK: Prentice-Hall

O'Halloran, O., Wolf, A. & Choset, H. (2005). Design of a high-impact survivable robot. *Mechanism and Machine Theory, 40,* 1345-1366.

Omron IAB (2006).W-series Ac Servomotors/Servodrivers Catalogue 2006/2007.

Ouyang, P.R., Li, Q. & Zhang, W.J. (2003). Integrated design of robotic mechanisms for force balancing and trajectory tracking. *Mechatronics, 13,* 887-905.

Park, K., Kim, Y.S., Kim, C.S. & Park, H.J. (2007). Integrated application of CAD/CAM/CAE and RP for rapid development of a humanoid biped robot. J*ournal of Materials Processing Technology,187-188,* 609-6.13

Shirinzadeh, B., Teoh, P. L., Foong, C. W. & Liu, Y.D. (1999). A strategy for accurate guidance of a manipulator using laser interferometry-based sensing technique. *Sensor Review, 19* (4), 292-299.

Sun, D. & Mills, J.K. (1999). High-accuracy trajectory tracking of ındustrial robot manipulator using adaptive-learning scheme. *Proceedings of the American Control Conference, California,* 1935-1939.

Thomson, C. C. (1984). Robot modelling-the tools needed for optimal design and utilization. *Computer-Aided Design*, *16*(6), 335-337.

Vukobratovic, M., Potkonjak, V., Inoue, K. & Takano, M. (2002). Actuators and computer-aided design of robots. Nwokah, O.D.I.(Ed.). Mechanical systems design handbook (523-556) CRC Press.

Wang, X.S., Xu, W.L., Tso, S.K & Zhang J.Z. (1997). Path error compensation of a two-link flexible robot arm based on integrated laser transducers. *IEEE International Conference, 4,* 3786-3790.

Xu, W.L., Tso, S.K. & Wang, X.S. (1998). Sensor based deflection modelling and compensation control of flexible robotic manipulator. *Mechanism and Machine Theory, 33* (7), 909-924.

Yang, T.W., Xu, W.L. & Tso, S.K. (2001). Dynamic modeling based on real-time deflection measurement and compensation control for flexible multi-link manipulators. *Dynamics and Control, 11,* 5-24.

Young, K. & Pickin, C.G. (2000). Accuracy assessment of the modern industrial robot. *Industrial Robot, 27* (6), 427-436.

Zhang, M.T. & Goldberg, K. (2005) Fixture-based industrial robot calibration for silicon wafer handling. *Industrial Robot, 32* (1), 43-48.

APPENDICES

APPENDIX A1
MATLAB CODE FOR INVERSE KINEMATIC

```
%=========================================================================
    %This program is performs the inverse kinematic calculation for
                              DEU-3X2-550
%=========================================================================

clc;clear
h=418;                      %Height of the second axis frame from ground
h1=276;                     %Height of the base
px=[63.0001 89 115 141 167 193 219 245 271 297 323];%End point x coord.
py=[0 0 0 0 0 0 0 0 0 0 0];                         %End point y coord.
pza=[433 433 433 433 433 433 433 433 433 433 433];  %End point z coord.
pz=pza-h;
n=11;                                                    %Point number
TETA1max=100; TETA1min=-100;               % Moving Restraint of Axis 1
TETA2max=180;TETA2min=50;                  % Moving Restraint of Axis 2
TETA3max=180;TETA3min=-90;                 % Moving Restraint of Axis 3
alfa1=pi/2;   alfa2=0;   alfa3=pi/2;             %
a1=-37;        a2=265;    a3=100;               %   Link Parameters
d3=0 ;        d4=250;                           %

fid = fopen('r3_imal1.dat','w');
fprintf(fid,'teta1       teta2        teta3\n');
%----------------------Calculation of teta 1--------------------------
for t=1:n;
ro=((px(t))^2+(py(t))^2)^(1/2);
fi=atan2((py(t)),(px(t)));
q11=atan2((d3/ro),(1-(d3^2/ro^2))^(1/2));
q12=atan2((d3/ro),-(1-(d3^2/ro^2))^(1/2));
teta11=fi-q11;
teta12=fi-q12;
teta11d=teta11*180/pi;
teta12d=teta12*180/pi;
%----------------------Calculation of teta 3--------------------------
K1=2*a2*d4;
K2=2*a2*a3;
K31=px(t)^2+py(t)^2+pz(t)^2-2*px(t)*a1*cos(teta11)-
2*py(t)*a1*sin(teta11)+a1^2-a2^2-a3^2-d4^2;
K32=px(t)^2+py(t)^2+pz(t)^2-2*px(t)*a1*cos(teta12)-
2*py(t)*a1*sin(teta12)+a1^2-a2^2-a3^2-d4^2;

teta31=2*atan2((K1+sqrt(K1^2+K2^2-K31^2)),(K31+K2));
teta32=2*atan2((K1-sqrt(K1^2+K2^2-K31^2)),(K31+K2));
teta33=2*atan2((K1+sqrt(K1^2+K2^2-K32^2)),(K32+K2));
teta34=2*atan2((K1-sqrt(K1^2+K2^2-K32^2)),(K32+K2));
```

```
teta31d=teta31*180/pi;
teta32d=teta32*180/pi;
teta33d=teta33*180/pi;
teta34d=teta34*180/pi;
TET1=[teta11 teta12];
TET3=[teta31 teta32 teta33 teta34];
%----------------------Calculation of teta 2---------------------------
s=0;d=0;
for i=1:2
    for j=1:4
    s=s+1 ;

mu1=a2+a3*cos(TET3(j))+d4*sin(TET3(j));
mu2=a3*sin(TET3(j))-d4*cos(TET3(j));
nu1=-a3*sin(TET3(j))+d4*cos(TET3(j));
nu2=a2+a3*cos(TET3(j))+d4*sin(TET3(j));
gama1=px(t)*cos(TET1(i))+py(t)*sin(TET1(i))-a1;
gama2=pz(t);

   A=[mu1 nu1
       mu2 nu2];
   b=[gama1
      gama2];

  xx=inv(A)*b;
  costeta2(s)=xx(1);
  sinteta2(s)=xx(2);
  TET2(s)=atan2(sinteta2(s),costeta2(s));

  d=d+1;
            TETA1(d)=TET1(i)*180/pi;
            TET2(s);
            if TET2(s)> pi;
               TET2(s)=TET2(s)-2*pi;
            elseif TET2(s)< -pi;
                TET2(s)=2*pi-TET2(s);
            end
            TET2(s);
            TETA2(d)=TET2(s)*180/pi;

            if TET3(j)> pi;
                TET3(j)=TET3(j)-2*pi;
            elseif TET3(j)< -pi;
                TET3(j)=2*pi+TET3(j);
            end
            TETA3(d)=TET3(j)*180/pi;
    end
end

coz1=[TETA1(1),TETA2(1),TETA3(1)];
coz2=[TETA1(1),TETA2(2),TETA3(2)];
coz3=[TETA1(5),TETA2(7),TETA3(7)];
coz4=[TETA1(6),TETA2(8),TETA3(8)];
```

```
if TETA1(1)> TETA1min && TETA1(1)< TETA1max && TETA2(1)> TETA2min &&
TETA2(1)< TETA2max && TETA3(1)> TETA3min && TETA3(1)< TETA3max
    cozum=coz1;

elseif TETA1(1)> TETA1min && TETA1(1)< TETA1max && TETA2(2)>TETA2min
&& TETA2(2)< TETA2max && TETA3(2)> TETA3min && TETA3(2)< TETA3max
    cozum=coz2;

elseif TETA1(5)> TETA1min && TETA1(5)< TETA1max && TETA2(7)> TETA2min
&& TETA2(7)< TETA2max && TETA3(7)> TETA3min && TETA3(7)< TETA3max
    cozum=coz3;

elseif TETA1(5)> TETA1min && TETA1(5)< TETA1max && TETA2(8)> TETA2min
&& TETA2(8)< TETA2max && TETA3(8)> TETA3min && TETA3(8)< TETA3max
    cozum=coz4;

else t,'Out of range'

end
fprintf(fid,'%6.4f     %6.4f      %6.4f    \n',cozum(1), cozum(2),
cozum(3));
    T1(t)=cozum(1)*pi/180;
    T2(t)=cozum(2)*pi/180;
    T3(t)=cozum(3)*pi/180;

    Tgraph1=T1*180/pi;
    Tgraph2=T2*180/pi;
    Tgraph3=T3*180/pi;

    T1k(t)=cozum(1)
    T2k(t)=cozum(2)
    T3k(t)=cozum(3)
end
fclose(fid);

%-----------------------------Drawing Graph-----------------------------

for i=1:n
  x(1)=0;
  y(1)=0;
  z(1)=0;
  x(2)=0;
  y(2)=0;
  z(2)=h1;
  line1=plot3(x,y,z,'LineWidth',6,...
                    'Color','red');
  %axis([-1000 1000 -1000 1000 -400 1000]);
  axis on;
  grid on;

  x(3)=x(2);
  y(3)=y(2);
  z(3)=z(2);
  x(4)=a1*cos(T1(i));
```

```
 y(4)=a1*sin(T1(i));
 z(4)=z(3);
 line2=plot3(x,y,z,'LineWidth',6,...
                   'Color','blue');
 %axis([-1000 1000 -1000 1000 -400 1000]);
 axis on;
 grid on;

 x(5)=x(4);
 y(5)=y(4);
 z(5)=z(4);
 x(6)=x(5);
 y(6)=y(5);
 z(6)=z(5)+h-h1;
 line3=plot3(x,y,z,'LineWidth',4,...
                   'Color','blue');
 %axis([-1000 1000 -1000 1000 -400 1000]);

 x(7)=x(6);
 y(7)=y(6);
 z(7)=z(6);
 x(8)=x(7)+a2*cos(T2(i))*cos(T1(i));
 y(8)=y(7)+a2*cos(T2(i))*sin(T1(i));
 z(8)=z(7)+a2*sin(T2(i));
 line4=plot3(x,y,z,'LineWidth',4,...
                   'Color','green');
 %axis([-1000 1000 -1000 1000 -400 1000]);

 x(9)=x(8);
 y(9)=y(8);
 z(9)=z(8);
 x(10)=x(9)+a3*cos(T2(i)+T3(i))*cos(T1(i));
 y(10)=y(9)+a3*cos(T2(i)+T3(i))*sin(T1(i));
 z(10)=z(9)+a3*sin(T2(i)+T3(i));
 line5=plot3(x,y,z,'LineWidth',4,...
                   'Color','green');
 %axis([-1000 1000 -1000 1000 -400 1000]);

 x(11)=x(10);
 y(11)=y(10);
 z(11)=z(10);
 x(12)=x(11)+d4*cos(T2(i)+T3(i)-pi/2)*cos(T1(i));
 y(12)=y(11)+d4*cos(T2(i)+T3(i)-pi/2)*sin(T1(i));
 z(12)=z(11)+d4*sin(T2(i)+T3(i)-pi/2);
 line6=plot3(x,y,z,'LineWidth',4,...
                   'Color','red');
% axis([-1000 1000 -1000 1000 -400 1000]);

 hold on;
 line7=plot3(px,py,pza,'LineWidth',2,...
                   'Color','green');
 nokta=plot3(px,py,pza,'o','MarkerSize',5);
% text(px(i),py(i),pza(i),...
 %     ['       ',num2str(i)],...
```

```
   %    'HorizontalAlignment','center');
  axis([-500 500 -500 1000 0 1000]);
  hold off;

  az = 0;%30
  el = 0;%30
  view(az, el);


  axis on;
  grid on;


  pause(0.2);

end
```

APENDIX A2
ABAQUS SCRIPT CODE FOR PARAMETRIC MODELLING AND THE FEM ANALYSES

```
This ABAQUS script code is performs parametric modeling and
static-frequency analyses for DEU-3X2-550
```

```
fx=0;fy=-20;fz=0;             #End point force

#--------------parametric dimensions of robot parts-----------------

b1=0.013;d1=0.009;d2=0.009
s0=0.10;de=0.002
p0a1=0.065;  p0t1=0.012;
p0r1=0.039;  p0t2=0.008; p0b1=0.047;
p0r2=0.1125;   p0b2=0.275; p0t3=0.010

p1a1=0.037;  p1a2=0.055;  p1b1=0.082;  p1b2=0.057;  p1d1=0.105;
p1r1=0.0125;       p1r2=0.04;   p1r3=0.0175;
p1t1=0.012;      p1t2=0.008;  p1t3=0.008;
p1b3=0.044;
p1t4=0.010;
p1r4=0.014; p1r5=0.021
p1d2=0.064

l2=0.265;    p2b1=0.08;
p2t1=0.006;   p2r1=0.021;
p2r2=0.021;  p2a1=0.065;  p2r3=0.0225;
p2d1=0.046;
p2t2=0.005;  p2t3=0.006

p3b1=0.1;  p3d1=0.09;
p3r1=0.01;  p3r2=0.04;
p3t1=0.005;  p3t2=0.005;  p3t3=0.005;  p3t4=0.003; p3t6=0.005
p3a1=2*p3r2+p3t6

l3=0.250;    p4r1=0.030;
p4t1=0.0025;  p4t2=0.010;  p4t3=p3t6;  p4t4=p4t2;  p4t5=p4t3
p4x2=l3-(p3a1-p3r2)

p3r3=p4r1-p4t1
p3t5=p4r1+p4t2-p3r3-p3t3;

from part import *
from material import *
from section import *
from assembly import *
from step import *
from interaction import *
from load import *
from mesh import *
from job import *
from sketch import *
```

```
from visualization import *
import xyPlot
import displayGroupOdbToolset as dgo
import visualization


r06=mdb.Model(name='r06')
del mdb.models['Model-1']

steel=r06.Material(name='Steel')
steel.Density(table=((7800,),))
steel.Elastic(table=((210e9,0.32),))
section1=r06.HomogeneousSolidSection(material='Steel',
name='section1',thickness=1.0)
aluminum=r06.Material(name='Aluminum')
aluminum.Density(table=((2700.0,),))
aluminum.Elastic(table=((71000000000.0,0.32),))
section2=r06.HomogeneousSolidSection(material='Aluminum',
name='section2',thickness=1.0)

#   Part0-------------------------------------------------------------

sketch1=r06.Sketch(name='Sketch1',sheetSize=s0)
sketch1.CircleByCenterPerimeter(center=(p0a1,0),point1=(p0a1+p0r2,0)
)
part0=r06.Part(name='Part0',
dimensionality=THREE_D,type=DEFORMABLE_BODY)
part0.BaseSolidExtrude(sketch=sketch1,depth=p0t1)
del sketch1

f1=part0.faces.findAt((p0a1,0,p0t1));
e1=part0.edges.findAt((p0a1+p0r2,0,p0t1));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil1',sheetSize=s0,
    transform=part0.MakeSketchTransform(sketchPlane=f1,
sketchPlaneSide=SIDE1,
    sketchUpEdge=e1,sketchOrientation=RIGHT,origin=(0,0,0)))
part0.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p0r1,0))
part0.SolidExtrude(depth=p0b1,flipExtrudeDirection=OFF,
sketch=sketch1,
sketchOrientation=RIGHT,sketchPlane=f1,sketchPlaneSide=SIDE1,
sketchUpEdge=e1)
del sketch1


f1=part0.faces.findAt((p0a1,0,0));
e1=part0.edges.findAt((p0a1+p0r2,0,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil1',sheetSize=s0,
transform=part0.MakeSketchTransform(sketchPlane=f1,
sketchPlaneSide=SIDE1,
    sketchUpEdge=e1,sketchOrientation=RIGHT,origin=(0,0,0)))
part0.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(p0a1,0),
point1=(p0a1+p0r2,0))
sketch1.CircleByCenterPerimeter(center=(p0a1,0),
point1=(p0a1+p0r2-p0t3,0))
part0.SolidExtrude(depth=p0b2,flipExtrudeDirection=OFF,
```

```
sketch=sketch1,
sketchOrientation=RIGHT,sketchPlane=f1,sketchPlaneSide=SIDE1,
sketchUpEdge=e1)
del sketch1

sketch1=r06.Sketch(gridSpacing=0.1*s0, name='profil1',sheetSize=s0,
transform=part0.MakeSketchTransform(part0.faces[6],
sketchPlaneSide=SIDE1,sketchUpEdge=part0.edges[1],
sketchOrientation=RIGHT,origin=(0,0,0)))
sketch1.sketchOptions.setValues(decimalPlaces=3)
part0.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p0r1-p0t2,0))
part0.CutExtrude(flipExtrudeDirection=OFF, sketch=sketch1,
sketchOrientation=RIGHT,sketchPlane=part0.faces[6],
sketchPlaneSide=SIDE1,sketchUpEdge=part0.edges[1])
del sketch1

#   Part1-----------------------------------------------------------

sketch1=r06.Sketch(name='Sketch1',sheetSize=s0)
sketch1.Line(point1=(p1r2,-p1b1+p1t1),point2=(p1r2,0))
sketch1.Line(point1=(-p1r2,-p1b1+p1t1),point2=(-p1r2,0))
sketch1.Line(point1=(-p1r2,-p1b1+p1t1),point2=(p1r2,-p1b1+p1t1))
sketch1.ArcByCenterEnds(center=(0,0),
direction=COUNTERCLOCKWISE,point1=(p1r2,0),point2=(-p1r2,0))
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p1r1,0))
part1=r06.Part(name='Part1',
dimensionality=THREE_D,type=DEFORMABLE_BODY)
part1.BaseSolidExtrude(sketch=sketch1,depth=p1t1)
del sketch1

part1.DatumPlaneByPrincipalPlane(offset=p1d1-
p1t3,principalPlane=XYPLANE)
f1=part1.datums[2];e1=part1.edges.findAt((p1r2,-de,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil1',sheetSize=s0,
    transform=part1.MakeSketchTransform(sketchPlane=f1,
sketchPlaneSide=SIDE1,
    sketchUpEdge=e1,sketchOrientation=RIGHT,origin=(0,0,0)))
part1.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.Line(point1=(p1r2,-p1b1+p1t1),point2=(p1r2,0))
sketch1.Line(point1=(-p1r2,-p1b1+p1t1),point2=(-p1r2,0))
sketch1.Line(point1=(-p1r2,-p1b1+p1t1),point2=(p1r2,-p1b1+p1t1))
sketch1.ArcByCenterEnds(center=(0,0),direction=COUNTERCLOCKWISE,
point1=(p1r2,0),point2=(-p1r2,0))
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p1r1,0))
part1.SolidExtrude(depth=p1t3,flipExtrudeDirection=OFF,
sketch=sketch1,sketchOrientation=RIGHT,sketchPlane=f1,
sketchPlaneSide=SIDE1,sketchUpEdge=e1)
del sketch1

part1.DatumPlaneByPrincipalPlane(offset=p1d2,principalPlane=XYPLANE)
f1=part1.datums[4];e1=part1.edges.findAt((p1r2,-de,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil1',sheetSize=s0,
transform=part1.MakeSketchTransform(sketchPlane=f1,
sketchPlaneSide=SIDE1,
    sketchUpEdge=e1,sketchOrientation=RIGHT,origin=(0,0,0)))
sketch1.Line(point1=(-p1a1-p1a2+p1r5,-p1b1+p1t1),
```

```
point2=(-p1a1-p1a2+p1r5,-p1b1+p1b3))
sketch1.Line(point1=(-p1a1-p1a2-p1r5,-p1b1+p1t1),
point2=(-p1a1-p1a2-p1r5,-p1b1+p1b3))
sketch1.Line(point1=(-p1a1-p1a2+p1r5,-p1b1+p1t1),
point2=(-p1a1-p1a2-p1r5,-p1b1+p1t1))
sketch1.ArcByCenterEnds(center=(-p1a1-p1a2,-
p1b1+p1b3),direction=COUNTERCLOCKWISE,
point1=(-p1a1-p1a2+p1r5,-p1b1+p1b3),
point2=(-p1a1-p1a2-p1r5,-p1b1+p1b3))
sketch1.CircleByCenterPerimeter(center=(-p1a1-p1a2,-p1b1+p1b3),
point1=(-p1a1-p1a2-p1r4,-p1b1+p1b3))
part1.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
part1.SolidExtrude(depth=p1t3,flipExtrudeDirection=OFF,
sketch=sketch1,sketchOrientation=RIGHT,sketchPlane=f1,
sketchPlaneSide=SIDE1,sketchUpEdge=e1)
del sketch1

f1=part1.faces.findAt((0,-p1r2-de,0));
e1=part1.edges.findAt((p1r2,-de,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil1',
sheetSize=s0, transform=part1.MakeSketchTransform(sketchPlane=f1,
sketchPlaneSide=SIDE1,
    sketchUpEdge=e1,sketchOrientation=RIGHT,origin=(0,0,0)))
part1.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p1r1,0))
sketch1.rectangle(point1=(-p1a1-p1a2-p1r5,p1b1-
p1t1),point2=(p1r2,p1b1))

part1.SolidExtrude(depth=p1d1,flipExtrudeDirection=ON,
sketch=sketch1,sketchOrientation=RIGHT,sketchPlane=f1,
sketchPlaneSide=SIDE1,sketchUpEdge=e1)
del sketch1

f1=part1.faces.findAt((0,-p1b1,de));
e1=part1.edges.findAt((0,-p1b1,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil1',sheetSize=s0,
transform=part1.MakeSketchTransform(sketchPlane=f1,
sketchPlaneSide=SIDE1,
    sketchUpEdge=e1,sketchOrientation=RIGHT,
origin=(-p1a1,-p1b1,0.5*p1d1)))
part1.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p1r3,0))

part1.SolidExtrude(depth=p1b2,flipExtrudeDirection=OFF,
sketch=sketch1,sketchOrientation=RIGHT,sketchPlane=f1,
sketchPlaneSide=SIDE1,sketchUpEdge=e1)
del sketch1

# Part2--------------------------------------------------------------

sketch1=r06.Sketch(name='Sketch1',sheetSize=s0)
sketch1.Line(point1=(0,-0.5*p2b1),point2=(l2,-0.5*p2b1))
sketch1.Line(point1=(0,0.5*p2b1),point2=(l2,0.5*p2b1))
sketch1.ArcByCenterEnds(center=(0,0),direction=COUNTERCLOCKWISE,
point1=(0,0.5*p2b1),point2=(0,-0.5*p2b1))
sketch1.ArcByCenterEnds(center=(l2,0),direction=COUNTERCLOCKWISE,
```

```
point1=(l2,-0.5*p2b1),point2=(l2,0.5*p2b1))
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p2r1,0))
sketch1.CircleByCenterPerimeter(center=(l2,0),point1=(l2+p2r2,0))
sketch1.CircleByCenterPerimeter(center=(p2a1,0),
point1=(p2a1+p2r3,0))
part2=r06.Part(name='Part2',
dimensionality=THREE_D,type=DEFORMABLE_BODY)
part2.BaseSolidExtrude(sketch=sketch1,depth=p2t1)
del sketch1

part2.DatumPlaneByPrincipalPlane(offset=p2d1-
p2t1,principalPlane=XYPLANE)
f1=part2.datums[2];e1=part2.edges.findAt((de,-0.5*p2b1,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil',sheetSize=s0,
    transform=part2.MakeSketchTransform(sketchPlane=f1,
    sketchPlaneSide=SIDE1,sketchUpEdge=e1,
    sketchOrientation=BOTTOM, origin=(0,0,0)))
part2.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.Line(point1=(0,-0.5*p2b1),point2=(l2,-0.5*p2b1))
sketch1.Line(point1=(0,0.5*p2b1),point2=(l2,0.5*p2b1))
sketch1.ArcByCenterEnds(center=(0,0),direction=COUNTERCLOCKWISE,
point1=(0,0.5*p2b1),point2=(0,-0.5*p2b1))
sketch1.ArcByCenterEnds(center=(l2,0),direction=COUNTERCLOCKWISE,
point1=(l2,-0.5*p2b1),point2=(l2,0.5*p2b1))
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p2r1,0))
sketch1.CircleByCenterPerimeter(center=(l2,0),point1=(l2+p2r2,0))
sketch1.CircleByCenterPerimeter(center=(p2a1,0),
point1=(p2a1+p2r3,0))
part2.SolidExtrude(depth=p2t1,flipExtrudeDirection=OFF,
sketch=sketch1,
sketchOrientation=BOTTOM,sketchPlane=f1,sketchPlaneSide=SIDE1,
sketchUpEdge=e1)
del sketch1

f1=part2.faces.findAt((0,p2r1+de,0));
e1=part2.edges.findAt((de,-0.5*p2b1,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil1',sheetSize=s0,
transform=part2.MakeSketchTransform(sketchPlane=f1,
sketchPlaneSide=SIDE1,
    sketchUpEdge=e1,sketchOrientation=BOTTOM,origin=(0,0,0)))
part2.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p2r1,0))
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p2r1+p2t2,0))
sketch1.CircleByCenterPerimeter(center=(-l2,0),point1=(-l2-p2r2,0))
sketch1.CircleByCenterPerimeter(center=(-l2,0),
point1=(-l2-p2r2-p2t2,0))
part2.SolidExtrude(depth=p2d1,flipExtrudeDirection=ON,
sketch=sketch1,
sketchOrientation=BOTTOM,sketchPlane=f1,sketchPlaneSide=SIDE1,
sketchUpEdge=e1)
del sketch1

part2.DatumPlaneByPrincipalPlane(offset=0,principalPlane=YZPLANE)
f1=part2.datums[5];e1=part2.edges.findAt((l2+0.5*p2b1,0,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0,name='profil1',sheetSize=s0,
transform=part2.MakeSketchTransform(sketchPlane=f1,
sketchPlaneSide=SIDE1,
```

```
    sketchUpEdge=e1,sketchOrientation=RIGHT,origin=(0,0,0.5*p2d1)))
part2.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.rectangle(point1=(-0.5*p2d1+p2t1,0.5*p2b1),
point2=(0.5*p2d1-p2t1,0.5*p2b1-p2t3))
sketch1.rectangle(point1=(-0.5*p2d1+p2t1,-0.5*p2b1),
point2=(0.5*p2d1-p2t1,-0.5*p2b1+p2t3))
part2.SolidExtrude(depth=l2,flipExtrudeDirection=OFF,sketch=sketch1,
sketchOrientation=RIGHT,sketchPlane=f1,sketchPlaneSide=SIDE1,
sketchUpEdge=e1)
del sketch1

#   Part3-----------------------------------------------------------
b1g=p3b1-p3r3-p3t4

sketch1=r06.Sketch(name='Sketch1',sheetSize=s0)
sketch1.Line(point1=(-p3r2,0),point2=(-p3r2,b1g-p3t2))
sketch1.Line(point1=(p3r2,0),point2=(p3r2,b1g-p3t2))
sketch1.Line(point1=(-p3r2,b1g-p3t2),point2=(p3r2,b1g-p3t2))
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p3r1,0))
sketch1.ArcByCenterEnds(center=(0,0),direction=COUNTERCLOCKWISE,
point1=(-p3r2,0),point2=(p3r2,0))
part3=r06.Part(name='Part3',
dimensionality=THREE_D,type=DEFORMABLE_BODY)
part3.BaseSolidExtrude(sketch=sketch1,depth=p3t1)
del sketch1

part3.DatumPlaneByPrincipalPlane(offset=p3d1-
p3t1,principalPlane=XYPLANE)
f1=part3.datums[2];e1=part3.edges.findAt((p3r2,de,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0, name='profil', sheetSize=s0,

transform=part3.MakeSketchTransform(sketchPlane=f1,sketchPlaneSide=S
IDE1,
    sketchUpEdge=e1,sketchOrientation=RIGHT,origin=(0,0,0)))
part3.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.Line(point1=(-p3r2,0),point2=(-p3r2,b1g-p3t2))
sketch1.Line(point1=(p3r2,0),point2=(p3r2,b1g-p3t2))
sketch1.Line(point1=(-p3r2,b1g-p3t2),point2=(p3r2,b1g-p3t2))
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p3r1,0))
sketch1.ArcByCenterEnds(center=(0,0),direction=COUNTERCLOCKWISE,
point1=(-p3r2,0),point2=(p3r2,0))
part3.SolidExtrude(depth=p3t1,flipExtrudeDirection=OFF,
sketch=sketch1,
  sketchOrientation=RIGHT,
sketchPlane=f1,sketchPlaneSide=SIDE1,sketchUpEdge=e1)
del sketch1

f1=part3.faces.findAt((0,p3r1+de,0));
e1=part3.edges.findAt((p3r2,de,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0, name='profil', sheetSize=s0,
    transform=part3.MakeSketchTransform(sketchPlane=f1,
    sketchPlaneSide=SIDE1,sketchUpEdge=e1,sketchOrientation=RIGHT,
origin=(0,0,0)))
part3.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p3r1,0))
sketch1.rectangle(point1=(-p3r2,-b1g),point2=(p3r2,-b1g+p3t2))
```

```
part3.SolidExtrude(depth=p3d1,flipExtrudeDirection=ON,
sketch=sketch1,
    sketchOrientation=RIGHT, sketchPlane=f1,sketchPlaneSide=SIDE1,
sketchUpEdge=e1)
del sketch1

f1=part3.faces.findAt((-p3r2,de,de));
e1=part3.edges.findAt((-p3r2,de,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0, name='profil', sheetSize=s0,
    transform=part3.MakeSketchTransform(sketchPlane=f1,
    sketchPlaneSide=SIDE1,sketchUpEdge=e1,sketchOrientation=RIGHT,
    origin=(0,0,0.5*p3d1)))
part3.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,-b1g-p3t4-p3r3),
point1=(p3r3,-b1g-p3t4-p3r3))
sketch1.Line(point1=(-p3r3-p3t3,-b1g),
point2=(-p3r3-p3t3,-b1g-p3t4-p3r3))
sketch1.Line(point1=(p3r3+p3t3,-b1g),
point2=(p3r3+p3t3,-b1g-p3t4-p3r3))
sketch1.Line(point1=(-p3r3-p3t3,-b1g),point2=(p3r3+p3t3,-b1g))
sketch1.ArcByCenterEnds(center=(0,-b1g-p3t4-p3r3),
direction=CLOCKWISE,point1=(p3r3+p3t3,-b1g-p3t4-p3r3),
point2=(-p3r3-p3t3,-b1g-p3t4-p3r3))
part3.SolidExtrude(depth=2*p3r2,flipExtrudeDirection=ON,
sketch=sketch1,
    sketchOrientation=RIGHT, sketchPlane=f1,
sketchPlaneSide=SIDE1, sketchUpEdge=e1)
del sketch1

f1=part3.faces.findAt((p3r2,de,de));
e1=part3.edges.findAt((-p3r2,de,0));
sketch1=r06.Sketch(gridSpacing=0.1*s0, name='profil', sheetSize=s0,
    transform=part3.MakeSketchTransform(sketchPlane=f1,
    sketchPlaneSide=SIDE1,sketchUpEdge=e1,sketchOrientation=RIGHT,
    origin=(0,b1g+p3t4+p3r3,0.5*p3d1)))
part3.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p3r3,0))
sketch1.CircleByCenterPerimeter(center=(0,0),
point1=(p3r3+p3t3+p3t5,0))
part3.SolidExtrude(depth=p3t6,flipExtrudeDirection=OFF,
sketch=sketch1,
    sketchOrientation=RIGHT, sketchPlane=f1,sketchPlaneSide=SIDE1,
sketchUpEdge=e1)
del sketch1

#   Part4-----------------------------------------------------------
x2=p4x2
sketch1=r06.Sketch(name='Sketch1',sheetSize=s0)
sketch1.HorizontalConstructionLine(point=(0.0, 0.0))
sketch1.Line(point1=(0,p4r1-p4t1),point2=(x2,p4r1-p4t1))
sketch1.Line(point1=(x2,p4r1-p4t1),point2=(x2,p4r1+p4t5))
sketch1.Line(point1=(x2,p4r1+p4t5),point2=(x2-p4t4,p4r1+p4t5))
sketch1.Line(point1=(x2-p4t4,p4r1+p4t5),point2=(x2-p4t4,p4r1))
sketch1.Line(point1=(x2-p4t4,p4r1),point2=(p4t3,p4r1))
sketch1.Line(point1=(p4t3,p4r1),point2=(p4t3,p4r1+p4t2))
sketch1.Line(point1=(p4t3,p4r1+p4t2),point2=(0,p4r1+p4t2))
sketch1.Line(point1=(0,p4r1+p4t2),point2=(0,p4r1-p4t1))
```

```
part4=r06.Part(name='Part4',
dimensionality=THREE_D,type=DEFORMABLE_BODY)
part4.BaseSolidRevolve(angle=360.0,flipRevolveDirection=OFF,
sketch=sketch1)
del sketch1

sketch1=r06.Sketch(gridSpacing=0.1*s0, name='profil', sheetSize=s0,
    transform=part4.MakeSketchTransform(sketchPlane=part4.faces[6],
    sketchPlaneSide=SIDE1,sketchUpEdge=part4.edges[12],
    sketchOrientation=RIGHT,origin=(0,0,0)))
part4.projectReferencesOntoSketch(filter=COPLANAR_EDGES,
sketch=sketch1)
sketch1.CircleByCenterPerimeter(center=(0,0),point1=(p4r1,0))
part4.SolidExtrude(depth=p4t3,flipExtrudeDirection=ON,
sketch=sketch1,
    sketchOrientation=RIGHT, sketchPlane=part4.faces[6],
    sketchPlaneSide=SIDE1, sketchUpEdge=part4.edges[12])

#   Property-------------------------------------------------------
part0.SectionAssignment(region=Region(cells=part0.cells[0:1]),
sectionName='section2')
part1.SectionAssignment(region=Region(cells=part1.cells[0:1]),
sectionName='section2')
part2.SectionAssignment(region=Region(cells=part2.cells[0:1]),
sectionName='section2')
part3.SectionAssignment(region=Region(cells=part3.cells[0:1]),
sectionName='section2')
part4.SectionAssignment(region=Region(cells=part4.cells[0:1]),
sectionName='section2')



#   Mesh-----------------------------------------------------------

mdb.models['r06'].parts['Part0'].seedPart(deviationFactor=0.1,
size=0.02)
mdb.models['r06'].parts['Part0'].setMeshControls(elemShape=TET,
regions=mdb.models['r06'].parts['Part0'].cells[0:1], technique=FREE)
mdb.models['r06'].parts['Part0'].generateMesh()

mdb.models['r06'].parts['Part1'].seedPart(deviationFactor=0.1,
size=0.008)
mdb.models['r06'].parts['Part1'].setMeshControls(elemShape=TET,
regions=mdb.models['r06'].parts['Part1'].cells[0:1], technique=FREE)
mdb.models['r06'].parts['Part1'].generateMesh()

mdb.models['r06'].parts['Part2'].seedPart(deviationFactor=0.1,
size=0.006)
mdb.models['r06'].parts['Part2'].setMeshControls(elemShape=TET,
regions=mdb.models['r06'].parts['Part2'].cells[0:1], technique=FREE)
mdb.models['r06'].parts['Part2'].generateMesh()

mdb.models['r06'].parts['Part3'].seedPart(deviationFactor=0.1,
size=0.006)
mdb.models['r06'].parts['Part3'].setMeshControls(elemShape=TET,
regions=mdb.models['r06'].parts['Part3'].cells[0:1], technique=FREE)
mdb.models['r06'].parts['Part3'].generateMesh()
```

```
mdb.models['r06'].parts['Part4'].seedPart(deviationFactor=0.1,
size=0.01)
mdb.models['r06'].parts['Part4'].setMeshControls(elemShape=TET,
regions=mdb.models['r06'].parts['Part4'].cells[0:1], technique=FREE)
mdb.models['r06'].parts['Part4'].generateMesh()

fileinp=open('kolacilar_abaqus.txt','r') #End point coordinates file
lignes=fileinp.readlines()

for i in range (1,21):
 xx=3*i-2
 yy=3*i-1
 zz=3*i

 th1=float(lignes[xx])
 th2=float(lignes[yy])
 th3=float(lignes[zz])     # Links Angles

 is_ismi='r06statik'
 mylist=[i]
 mystring=str(mylist)
 aaa=mystring
 bbb=aaa[1:-1]
 jobname=is_ismi+bbb
 uzanti='.odb'
 odbname=jobname+uzanti

 #   Assembly----------------------------------------------------------

 r06.rootAssembly.DatumCsysByDefault(CARTESIAN)
 part0i=r06.rootAssembly.Instance(dependent=ON,name='Part0i',
part=part0)
 part1i=r06.rootAssembly.Instance(dependent=ON,name='Part1i',
part=part1)
 part2i=r06.rootAssembly.Instance(dependent=ON,name='Part2i',
part=part2)
 part3i=r06.rootAssembly.Instance(dependent=ON,name='Part3i',
part=part3)
 part4i=r06.rootAssembly.Instance(dependent=ON,name='Part4i',
part=part4)


 x0p1=p1a1;y0p1=p0b1+b1+p1b1;z0p1=-0.5*p1d1
 x0p2=x0p1;y0p2=y0p1;z0p2=-0.5*p2d1-d1
 x0p3=x0p2;y0p3=y0p2+l2;z0p3=-0.5*p3d1+d2-d1;
 x0p4=x0p3-p3a1+p3r2;y0p4=y0p3+p3b1;z0p4=d2-d1
 part0i.rotateAboutAxis(angle=- 90,axisDirection=(s0,0,0),
 axisPoint=(0,0,0))
 part0i.rotateAboutAxis(angle=180,axisDirection=(0,s0,0),
 axisPoint=(0,0,0))
 part1i.translate(vector=(x0p1,y0p1,z0p1))
 part2i.translate(vector=(x0p2,y0p2,z0p2))
 part2i.rotateAboutAxis(angle=90,axisDirection=(0,0,s0),
 axisPoint=(x0p2,y0p2,0))
 part3i.translate(vector=(x0p3,y0p3,z0p3))
 part4i.translate(vector=(x0p4,y0p4,z0p4))

 part3i.rotateAboutAxis(angle=th3,axisDirection=(0,0,s0),
 axisPoint=(x0p3,y0p3,0))
```

```
 part4i.rotateAboutAxis(angle=th3,axisDirection=(0,0,s0),
 axisPoint=(x0p3,y0p3,0))
 part2i.rotateAboutAxis(angle=th2,axisDirection=(0,0,s0),
 axisPoint=(x0p2,y0p2,0))
 part3i.rotateAboutAxis(angle=th2,axisDirection=(0,0,s0),
 axisPoint=(x0p2,y0p2,0))
 part4i.rotateAboutAxis(angle=th2,axisDirection=(0,0,s0),
 axisPoint=(x0p2,y0p2,0))
 part1i.rotateAboutAxis(angle=th1,axisDirection=(0,s0,0),
 axisPoint=(0,0,0))
 part2i.rotateAboutAxis(angle=th1,axisDirection=(0,s0,0),
 axisPoint=(0,0,0))
 part3i.rotateAboutAxis(angle=th1,axisDirection=(0,s0,0),
 axisPoint=(0,0,0))
 part4i.rotateAboutAxis(angle=th1,axisDirection=(0,s0,0),
 axisPoint=(0,0,0))

#   Step-------------------------------------------------------------

 r06.StaticStep(name='Step-1',previous='Initial')
 r06.FrequencyStep(name='Step-2', numEigen=3, previous='Step-1')

#   Interaction------------------------------------------------------
 r06.ConnectorProperty(assembledType=WELD,name='weld',
 rotationalType=None,
 translationalType=None)
 r06.ConnectorProperty(assembledType=HINGE,name='Hinge',
 rotationalType=None,
 translationalType=None)

 r06.rootAssembly.ReferencePoint(point=(0,0.5*p0b1,0))
 r06.rootAssembly.ReferencePoint(point=(x0p1-p1a1,y0p1-p1b1-
 0.5*p1b2,0))

r06.rootAssembly.ReferencePoint(point=(x0p1*cos(th1*3.141593/180),y0
p1,- x0p1*sin(th1*3.141593/180)))
 r06.rootAssembly.ReferencePoint(point=(x0p2*cos(th1*3.141593/180)-
 d1*sin(th1*3.141593/180),y0p2,-x0p2*sin(th1*3.141593/180)-
 d1*cos(th1*3.141593/180)))
 r06.rootAssembly.ReferencePoint(point=((x0p2-
 l2*sin(th2*3.141593/180))*cos(th1*3.141593/180),
        y0p2+l2*cos(th2*3.141593/180),
        (-x0p2+l2*sin(th2*3.141593/180))*sin(th1*3.141593/180)))
 r06.rootAssembly.ReferencePoint(point=((x0p2-
 l2*sin(th2*3.141593/180))*cos(th1*3.141593/180),
        y0p2+l2*cos(th2*3.141593/180),
        (-x0p2+l2*sin(th2*3.141593/180))*sin(th1*3.141593/180)))

 r06.rootAssembly.ReferencePoint(point=((x0p2-
l2*sin(th2*3.141593/180)-
 p3b1*sin(th2*3.141593/180+th3*3.141593/180)-
(p3r2+p3t6)*cos(th2*3.141593/180+th3*3.141593/180))*cos(th1*3.141593
/180),
y0p2+l2*cos(th2*3.141593/180)+p3b1*cos(th2*3.141593/180+th3*3.141593
/180)- (p3r2+p3t6)*sin(th2*3.141593/180+th3*3.141593/180),
 (-x0p2-l2*sin(th2*3.141593/180)-
p3b1*sin(th2*3.141593/180+th3*3.141593/180)-
p3r2+p3t6)*cos(th2*3.141593/180+th3*3.141593/180))*sin(th1*3.141593/
180)))
```

```
 r06.rootAssembly.ReferencePoint(point=((x0p2-
l2*sin(th2*3.141593/180)-
p3b1*sin(th2*3.141593/180+th3*3.141593/180)-
(p3r2+p3t6)*cos(th2*3.141593/180+th3*3.141593/180))*cos(th1*3.141593
/180),
y0p2+l2*cos(th2*3.141593/180)+p3b1*cos(th2*3.141593/180+th3*3.141593
/180)-(p3r2+p3t6)*sin(th2*3.141593/180+th3*3.141593/180),
(-x0p2-l2*sin(th2*3.141593/180)-
p3b1*sin(th2*3.141593/180+th3*3.141593/180)-
(p3r2+p3t6)*cos(th2*3.141593/180+th3*3.141593/180))*sin(th1*3.141593
/180)))

 r06.rootAssembly.ReferencePoint(point=((x0p2-
l2*sin(th2*3.141593/180)-
p3b1*sin(th2*3.141593/180+th3*3.141593/180)-
l3*cos(th2*3.141593/180+th3*3.141593/180))*cos(th1*3.141593/180),
y0p2+l2*cos(th2*3.141593/180)+p3b1*cos(th2*3.141593/180+th3*3.141593
/180)-l3*sin(th2*3.141593/180+th3*3.141593/180),
(-x0p2-l2*sin(th2*3.141593/180)-
p3b1*sin(th2*3.141593/180+th3*3.141593/180)-
l3*cos(th2*3.141593/180+th3*3.141593/180))*sin(th1*3.141593/180)))


 r06.rootAssembly.DatumCsysByTwoLines(CARTESIAN,
line1=r06.rootAssembly.instances['Part1i'].edges[49],
   line2=r06.rootAssembly.instances['Part1i'].edges[9], name='Teta1
Ekseni')
 r06.rootAssembly.DatumCsysByTwoLines(CARTESIAN,
line1=r06.rootAssembly.instances['Part1i'].edges[9],
   line2=r06.rootAssembly.instances['Part1i'].edges[38],
name='Teta2-3 Ekseni')


 r06.Connector(name='Conn-1',
orientation1=r06.rootAssembly.datums[21+(i-1)*43],
point1=r06.rootAssembly.referencePoints[12+(i-1)*43],
point2=r06.rootAssembly.referencePoints[13+(i-
1)*43],property='weld')
 r06.Connector(name='Conn-2',
orientation1=r06.rootAssembly.datums[22+(i-1)*43],
point1=r06.rootAssembly.referencePoints[14+(i-1)*43],
point2=r06.rootAssembly.referencePoints[15+(i-
1)*43],property='weld')
 r06.Connector(name='Conn-3',
orientation1=r06.rootAssembly.datums[22+(i-1)*43],
point1=r06.rootAssembly.referencePoints[16+(i-1)*43],
point2=r06.rootAssembly.referencePoints[17+(i-
1)*43],property='weld')
 r06.Connector(name='Conn-4',
orientation1=r06.rootAssembly.datums[21+(i-1)*43],
point1=r06.rootAssembly.referencePoints[18+(i-1)*43],
point2=r06.rootAssembly.referencePoints[19+(i-
1)*43],property='weld')

r06.rootAssembly.Set(name='uc_nokta',
referencePoints=(r06.rootAssembly.referencePoints[20+(i-1)*43], ))
```

```
r06.historyOutputRequests.changeKey(fromName='H-Output-
1',toName='cikti')

r06.historyOutputRequests['cikti'].setValues(rebar=EXCLUDE,region=r0
6.rootAssembly.sets['uc_nokta'],sectionPoints=DEFAULT,variables=('U1
','U2','U3'))

r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
referencePoints[12+(i-1)*43],)),couplingType=KINEMATIC,
influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-1',
surface=Region(side1Faces=r06.rootAssembly.instances['Part0i'].faces
[1:2]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

 r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
 referencePoints[13+(i-1)*43],)),couplingType=KINEMATIC,
 influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-2',

surface=Region(side1Faces=r06.rootAssembly.instances['Part1i'].faces
[0:1]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

 r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
 referencePoints[14+(i-1)*43],)),couplingType=KINEMATIC,
 influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-3',

surface=Region(side1Faces=r06.rootAssembly.instances['Part1i'].faces
[2:3]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

 r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
 referencePoints[15+(i-1)*43],)),couplingType=KINEMATIC,
 influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-4',

surface=Region(side1Faces=r06.rootAssembly.instances['Part2i'].faces
[12:13]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

 r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
 referencePoints[16+(i-1)*43],)),couplingType=KINEMATIC,
 influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-5',

surface=Region(side1Faces=r06.rootAssembly.instances['Part2i'].faces
[11:12]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

 r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
 referencePoints[17+(i-1)*43],)),couplingType=KINEMATIC,
 influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-6',

surface=Region(side1Faces=r06.rootAssembly.instances['Part3i'].faces
[11:12]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

 r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
 referencePoints[18+(i-1)*43],)),couplingType=KINEMATIC,
 influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-7',
```

```
surface=Region(side1Faces=r06.rootAssembly.instances['Part3i'].faces
[2:3]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

 r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
 referencePoints[19+(i-1)*43],)),couplingType=KINEMATIC,
 influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-8',

surface=Region(side1Faces=r06.rootAssembly.instances['Part4i'].faces
[1:2]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

 r06.Coupling(controlPoint=Region(referencePoints=(r06.rootAssembly.
 referencePoints[20+(i-1)*43],)),couplingType=KINEMATIC,
 influenceRadius=WHOLE_SURFACE,localCsys=None,name='Constraint-9',

surface=Region(side1Faces=r06.rootAssembly.instances['Part4i'].faces
[7:8]),
 u1=ON, u2=ON, u3=ON, ur1=ON, ur2=ON, ur3=ON)

#   Load-----------------------------------------------------------

 r06.Gravity(comp2=-9.81,createStepName='Step-1',name='Load-2')
 r06.ConcentratedForce(cf1=fx,cf2=fy,cf3=fz,
 createStepName='Step-1',localCsys=None,name='Load-1',
 region=Region(referencePoints=(r06.rootAssembly.
 referencePoints[20+(i-1)*43],)))

 f1=part0i.faces[2:3]
 r06.EncastreBC(createStepName='Initial',
name='BC-1',region=Region(faces=f1))

#    Job------------------------------------------------------------

 mdb.Job(name=jobname, model='r06',
type=ANALYSIS,explicitPrecision=SINGLE,
 nodalOutputPrecision=SINGLE, description='',
     parallelizationMethodExplicit=DOMAIN,
 multiprocessingMode=DEFAULT,numDomains=1,userSubroutine='',
 numCpus=1,preMemory=2000.0,
     standardMemory=2000.0, standardMemoryPolicy=MODERATE,
scratch='',echoPrint=OFF,modelPrint=OFF,contactPrint=OFF,historyPrin
t=OFF)
 mdb.jobs[jobname].submit()
 mdb.jobs[jobname].waitForCompletion()



 #      odb-----------------------------------------------------------
 odb = visualization.openOdb(odbname , readOnly=TRUE)
 odb.close()

 del mdb.models['r06'].rootAssembly.features['RP-1']
 del mdb.models['r06'].rootAssembly.features['RP-2']
 del mdb.models['r06'].rootAssembly.features['RP-3']
 del mdb.models['r06'].rootAssembly.features['RP-4']
 del mdb.models['r06'].rootAssembly.features['RP-5']
 del mdb.models['r06'].rootAssembly.features['RP-6']
```

```
del mdb.models['r06'].rootAssembly.features['RP-7']
del mdb.models['r06'].rootAssembly.features['RP-8']
del mdb.models['r06'].rootAssembly.features['RP-9']

del mdb.models['r06'].rootAssembly.features['Datum csys-1']
del mdb.models['r06'].rootAssembly.features['Teta1 Ekseni']
del mdb.models['r06'].rootAssembly.features['Teta2-3 Ekseni']

del mdb.models['r06'].connectors['Conn-1']
del mdb.models['r06'].connectors['Conn-2']
del mdb.models['r06'].connectors['Conn-3']
del mdb.models['r06'].connectors['Conn-4']

del mdb.models['r06'].loads['Load-1']
del mdb.models['r06'].loads['Load-2']
del mdb.models['r06'].boundaryConditions['BC-1']

print i
```