

**DOKUZ EYLÜL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED
SCIENCES**

**TRAJECTORY PLANNING OF A SIX AXIS
SERIAL ROBOT BASED ON DYNAMIC
ANALYSIS**

**by
Şahin YAVUZ**

**July, 2012
İZMİR**

TRAJECTORY PLANNING OF A SIX AXIS SERIAL ROBOT BASED ON DYNAMIC ANALYSIS

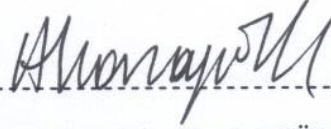
**A Thesis Submitted to the
Graduate School of Natural and Applied Sciences of Dokuz Eylül University
In Partial Fulfillment of the Requirements for the Degree of Master of Science
in Mechanical Engineering, Machine Theory and Dynamics Program**

**by
Şahin YAVUZ**

**July, 2012
İZMİR**

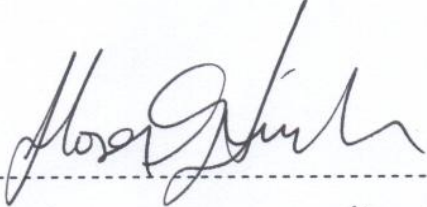
M.Sc THESIS EXAMINATION RESULT FORM

We have read the thesis entitled “TRAJECTORY PLANNING OF A SIX AXIS SERIAL ROBOT BASED ON DYNAMIC ANALYSIS” completed by SAHIN YAVUZ under supervision of PROF. DR. HİRA KARAGÜLLE and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.



PROF. DR. HİRA KARAGÜLLE

Supervisor



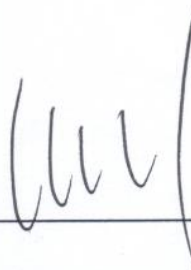
Assoc. Prof. Hasan ÖZTÜRK

(Jury Member)



Assist. Prof. Yavuz ŞENOL

(Jury Member)



Prof.Dr. Mustafa SABUNCU

Director

Graduate School of Natural and Applied Sciences

ACKNOWLEDGMENTS

I would like to express my sincere indebtedness and gratitude to my thesis consultant Prof. Dr. Hira KARAGÜLLE, for all his time and effort. His guidance and input at every stage of my work truly helped me navigate this endeavor.

I would also thank Dr. Murat AKDAG (Engineering Faculty of Dokuz Eylül University), my friends Armin AMINDARI and Serkan GÜLER for their encouragement and intellectual input during the entire course of this thesis.

Finally, I would like to thank my parents and my sister who have supported me all the way since the beginning of my studies. Their love gave me forces to make this work.

Şahin YAVUZ

TRAJECTORY PLANNING OF A SIX AXIS SERIAL ROBOT BASED ON DYNAMIC ANALYSIS

ABSTRACT

Serial robots are by far the most common robots used in industrial applications such as welding, pick and place, and various processes. Computers and engineering programs play important role to constitute motion analysis and design process. These days, engineers can analyze all parameters to produce the generous robots which are required for the process and find effective solutions quickly by using computer programs with the development of the technology.

In this study, a six axis serial robot is produced by using integrated approaches with parallel analysis in SolidWorks, CosmosMotion, VisualBasic, Abaqus and Adlink Motion Control Card. The parts of the robot are modeled in SolidWorks, rigidity is verified with finite element analysis in Abaqus and produced the real model of the robot. Kinematic analysis is studied in CosmosMotion by defining the velocity profile to the end effector point through the software algorithm developed in VisualBasic, which uses the application programming interface (API) capabilities. Different tasks are given to the robot in this study. The kinematic workspace and maximum motor velocities are evaluated by inverse kinematic analysis for each task. The kinetic workspace and maximum motor torques are evaluated by forward kinetic analysis. The rigidity workspace is evaluated by finite element analysis. The optimum trajectory is chosen after reviewing dynamic analysis and finite element results and the same trajectory is generated using ADLINK motion control card. Velocity mode is used. The maximum deviation for each motor is determined which is between the sending velocity profile data and the receiving feedback velocity profile data and compared whether the deviation values are in acceptable ranges.

Keywords: Robot programming, dynamic analysis, trajectory planning

TRAJECTORY PLANNING OF A SIX AXIS SERIAL ROBOT BASED ON DYNAMIC ANALYSIS

ÖZ

Seri robotlar günümüzde kaynak, taşıma ve çeşitli işlemler gibi sanayi uygulamalarında kullanılan en yaygın robotlardır. Hareket analizleri ve dizayn süreçlerini oluşturmak için bilgisayarlar ve mühendislik programları önemli rol oynamaktadır. Günümüzde, teknolojinin gelişmesiyle birlikte mühendisler süreç için gerekli olan verimli robotu üretmek için bilgisayar programlarını kullanarak bütün parametreleri analiz edebilmekte ve etkin çözümler bulabilmektedirler.

Bu çalışmada, SolidWorks, CosmosMotion, VisualBasic ve Adlink hareket kontrol kartın paralel analizi ile birlikte entegre yaklaşımlar kullanılarak 6 eksenli seri bir robot üretilmiştir. Robotun parçaları SolidWorks programında modellenmiş, sonlu elemanlar yöntemi ile Abaqus programında rijitliği doğrulanmış ve robotun gerçek modeli üretilmiştir. Uygulama programlama arayüzü (API) özelliklerini kullanan VisualBasic ortamında geliştirilen yazılım algoritması sayesinde uç noktaya hız profili tanımlanarak CosmosMotion programında kinematik analizi çalışması yapılmıştır. Bu çalışmada, robota farklı görevler verildi. Her görev için ters kinematik analizi ile kinematik çalışma alanı ve maksimum motor hızları değerlendirildi. İleri kinetik analizi ile kinetik çalışma alanı ve maksimum motor torkları değerlendirildi. Sonlu elemanlar analizi ile rijitlik çalışma alanı değerlendirildi. Dinamik analizi ve sonlu elemanlar analizi sonuçları incelendikten sonra en ideal yörünge seçildi ve aynı yörünge ADLINK hareket kontrol kartı kullanılarak üretildi. Hız modu fonksiyonu kullanıldı. Her motor için gönderilen hız profili datası ile alınan geri besleme hız profili datası arasındaki maksimum sapma miktarı hesaplandı ve kabul edilebilir sınırlar içinde olup olmadığı karşılaştırıldı.

Anahtar Sözcükler: Robot programlama, dinamik analiz, yörünge planlama

CONTENTS

	Pages
M.Sc THESIS EXAMINATION RESULT FORM.....	ii
ACKNOWLEDGMENTS	iii
ABSTRACT.....	iv
ÖZ	v
 CHAPTER ONE - INTRODUCTION	 1
1.1 Literature Survey	1
1.2 Thesis Organization.....	3
 CHAPTER TWO - INDUSTRIAL ROBOTS	 4
2.1 History of Industrial Robotics	4
2.2 Defining Parameters	5
2.3 Standard robot types	8
2.3.1 Articulate Robot.....	8
2.3.2 Cylindrical robots or spherical robots.....	10
2.3.3 Cartesian robots	12
 CHAPTER THREE - INTEGRATED DYNAMIC ANALYSIS	 14
3.1 Flow Chart.....	14
3.2 Modeling of members of a six axis serial robot	17
3.3 Integration Software (IS)	20
3.3.1 Database for Planned Work	20
3.3.2 Inverse Kinematics and Kinematic Workspace Evaluation in IS	21
3.3.3 Forward kinetics in IS	24

CHAPTER FOUR - SIMULATION RESULTS	26
4.1 Case Studies	26
4.2 Case 1	27
4.2.1 Simulation results for dynamic F.E ABAQUS analysis for case 1	28
4.3 Case 2	32
4.3.1 Simulation results for dynamic F.E ABAQUS analysis for case 2	34
4.4 Case-3	37
4.4.1 Simulation results for dynamic F.E ABAQUS analysis for case 3	39
 CHAPTER FIVE - EXPERIMENTAL RESULTS FOR MODAL ANALYSIS	42
 CHAPTER SIX - TRAJECTORY PLANNING WITH MOTION CONTROL	
CARD ADLINK 8366	47
 CHAPTER SEVEN - CONCLUSION	59
 REFERENCES.....	62

CHAPTER ONE

INTRODUCTION

According to the Robot Institute of America (1979) a robot is: "A reprogrammable, multifunctional manipulator designed to move material, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks".

A more inspiring definition can be found in Webster' Dictionary. According to Webster a robot is: "An automatic device that performs functions normally ascribed to humans or a machine in the form of a human."

1.1 Literature Survey

A significant amount of research has been reported concerning trajectory planning in robot manipulators (D.E Whitney, 1969). Most of them are based on the calculation of inverse kinematics employing a pseudo-inverse of the Jacobian matrix. Trajectory planning using the minimal-time criterion was proposed under the B-Spline assumption of the Cartesian path (J.E Borrow, 1988). A new method for time-optimal motion planning based on improved genetic algorithms was proposed, which incorporates kinematics constraints, dynamics constraints and control constraints of the robotic manipulator (W.M.YUN &Y.G Xi, 1990).

Arakelian and Baron dealt with the analytically tractable solution for input torques minimisation of two degrees of freedom serial manipulators based on minimum energy control and optimal redistribution of movable masses. They carried out the minimisation problem in two steps: at first, the optimal trajectory of the manipulator is defined as a function, which leads to the minimisation of energy consumption. Then, by introducing the obtained trajectory into dynamic equations, the torques are reduced by using the optimal redistribution of movable masses, which is carried out via an adaptive counterweight system.

A new approach to solve the inverse kinematics problem of redundant robot manipulators in environments cluttered with obstacles is presented by Nearchou. According to Nearchou, the physical problem is formulated as an optimization problem under constraints, and solved via a modified genetic algorithm (mGA). The mGA searches for successive robot configurations in the entire free space so that the robot moves its end-effector from an initial placement to a final desired. The objective of this optimization is to simultaneously minimize the end-effector's positional error and the robot's joint displacements.

One of the major objects of the research in the robotic systems is to evaluate dynamic analysis and to control the motion of the robot by using different methods. Some researchers carried out an adoptive learning strategy using neural an artificial network algorithm to control the motion of a 6 D.O.F manipulator robot and to overcome the inverse kinematics problem(A.T. Hasan & A.M.S. Hamouda & N. Ismail & H.M.A.A. Al-Assadi, 2006).Angeles and Ma studied on the systematic derivation of the $n \times n$ generalized inertia matrix of a general n -axis manipulator. They developed two methods to compute the joint accelerations from the governing equations of motion, which are based on the Cholesky decomposition of the generalized inertia matrix and a combination of minimum-norm and least-squares approximations of under determined and over determined linear algebraic systems. These two researchers developed also an algorithm based on Kane's dynamical equations for the numerical solution of the inverse dynamics of robotic manipulators of the serial type which produces the time histories of the n torques or forces required to drive the manipulator through the prescribed trajectory.

Nguyen and Pooran dealt with the design of the end effector and showed the effects of system parameters on the end-effector dynamics. The emphasis in their some researches is to employ Lagrangian approach to derive a set of equations of motion and Newton-Raphson method to solve forward kinematic.

The accuracy and pricision has been studied by a few researchers for the motion of the robot. A new motion control strategy for manipulators with redundant serial

kinematics is proposed by developing an algorithm which works in real time motion (Lotz & Bruhm & Czinki & Zalewski, 2011). Robots are often required to handle payloads with unknown dynamics. Song and Anderson studied on a adaptive path tracking control of a robot carrying an unknown payload. They investigated the inner product of the payload and used an alternative approach for stability analysis. An integrated design process presented by Karagülle and applied to a hexapod robot which can be used for medical operations.

1.2 Thesis Organization

In this study the evaluation of the kinematic and kinetic parameters is achieved with a main program developed in Visual Basic, which uses the application programming interface (API) capabilities.

Chapter II introduces the history of industrial robots followed by classification of the robots.

Chapter III explains the flowchart of the evaluation and modeling of the robot. Finite element analysis, inverse kinematics and forward kinetic analysis of the robot is discussed.

Chapter IV presents various simulation results for different cases.

Chapter V presents experimental results of the modal testing of the robot.

Chapter VI presents the motion of the robot with the optimum trajectory which is chosen as a result of one of the case studies.

CHAPTER TWO

INDUSTRIAL ROBOTS

2.1 History of Industrial Robotics

An industrial robot is defined by ISO as an automatically controlled, reprogrammable, multipurpose manipulator programmable in three or more axes. The field of robotics may be more practically defined as the study, design and use of robot systems for manufacturing (a top-level definition relying on the prior definition of robot).

Typical applications of robots include welding, painting, assembly, pick and place (such as packaging, palletizing and SMT), product inspection, and testing; all accomplished with high endurance, speed, and precision.

George Devol applied for the first robotics patents in 1954 (granted in 1961). The first company to produce a robot was Unimation, founded by 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. They used hydraulic actuators and were 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 GKN, 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.

In 1969 Victor Scheinman at Stanford University invented the Stanford arm, an all-electric, 6-axis articulated robot designed to permit an arm solution. This allowed it accurately to follow arbitrary paths in space and widened the potential use of the robot to more sophisticated applications such as assembly and welding. Scheinman then designed a second arm for the MITAI Lab, called the "MIT arm." Scheinman,

after receiving a fellowship from Unimation to develop his designs, sold those designs to Unimation who further developed them with support from General Motors and later marketed it as the Programmable Universal Machine for Assembly (PUMA).

Industrial robotics took off quite quickly in Europe, with both ABB Robotics and KUKA Robotics bringing robots to the market in 1973. ABB Robotics (formerly ASEA) introduced IRB 6, among the world's first commercially available all electric micro-processor controlled robot. The first two IRB 6 robots were sold to Magnusson in Sweden for grinding and polishing pipe bends and were installed in production in January 1974. Also in 1973 KUKA Robotics built its first robot, known as FAMULUS, also one of the first articulated robot to have six electromechanically driven axes.

Interest in robotics increased in the late 1970s and many US companies entered the field, including large firms like General Electric, and General Motors (which formed joint venture FANUC Robotics with FANUC LTD of Japan). U.S. startup companies included Automatix and Adept Technology, Inc. At the height of the robot boom in 1984, Unimation was acquired by Westinghouse Electric Corporation for 107 million U.S. dollars. Westinghouse sold Unimation to Stäubli Faverges SCA of France in 1988, which is still making articulated robots for general industrial and clean room applications and even bought the robotic division of Bosch in late 2004.

Only a few non-Japanese companies ultimately managed to survive in this market, the major ones being Adept Technology, Stäubli-Unimation, the Swedish-Swiss company ABB Asea Brown Boveri and the German company KUKA Robotics.

2.2 Defining Parameters

Number of axes – two axes are required to reach any point in a plane; three axes are required to reach any point in space. To fully control the orientation of the end of the arm (i.e. the *wrist*) three more axes (yaw, pitch, and roll) are required. Some designs

(e.g. the SCARA robot) trade limitations in motion possibilities for cost, speed, and accuracy.

Degrees of freedom – which is usually the same as the number of axes.

Working envelope – the region of space a robot can reach.

Kinematics – the actual arrangement of rigid members and joints in the robot, which determines the robot's possible motions. Classes of robot kinematics include articulated, cartesian, parallel and SCARA.

Carrying capacity or payload – how much weight a robot can lift.

Speed– how fast the robot can position the end of its arm. This may be defined in terms of the angular or linear speed of each axis or as a compound speed i.e. the speed of the end of the arm when all axes are moving.

Acceleration - how quickly an axis can accelerate. Since this is a limiting factor a robot may not be able to reach its specified maximum speed for movements over a short distance or a complex path requiring frequent changes of direction.

Accuracy – how closely a robot can reach a commanded position. When the absolute position of the robot is measured and compared to the commanded position the error is a measure of accuracy. Accuracy can be improved with external sensing for example a vision system or Infra-Red. See robot calibration. Accuracy can vary with speed and position within the working envelope and with payload (see compliance).

Repeatability - how well the robot will return to a programmed position. This is not the same as accuracy. It may be that when told to go to a certain X-Y-Z position that it gets only to within 1 mm of that position. This would be its accuracy which may be improved by calibration. But if that position is taught into controller memory and each time it is sent there it returns to within 0.1mm of the taught position then the repeatability will be within 0.1mm.

Accuracy and repeatability are different measures. Repeatability is usually the most important criterion for a robot and is similar to the concept of 'precision' in measurement - see Accuracy and precision. ISO 9283 sets out a method whereby

both accuracy and repeatability can be measured. Typically a robot is sent to a taught position a number of times and the error is measured at each return to the position after visiting 4 other positions. Repeatability is then quantified using the standard deviation of those samples in all three dimensions. A typical robot can, of course make a positional error exceeding that and that could be a problem for the process. Moreover the repeatability is different in different parts of the working envelope and also changes with speed and payload. ISO 9283 specifies that accuracy and repeatability should be measured at maximum speed and at maximum payload. But this results in pessimistic values whereas the robot could be much more accurate and repeatable at light loads and speeds. Repeatability in an industrial process is also subject to the accuracy of the end effector, for example a gripper, and even to the design of the 'fingers' that match the gripper to the object being grasped. For example if a robot picks a screw by its head the screw could be at a random angle. A subsequent attempt to insert the screw into a hole could easily fail. These and similar scenarios can be improved with 'lead-ins' e.g. by making the entrance to the hole tapered.

Motion control – for some applications, such as simple pick-and-place assembly, the robot need merely return repeatably to a limited number of pre-taught positions. For more sophisticated applications, such as welding and finishing (spray painting), motion must be continuously controlled to follow a path in space, with controlled orientation and velocity.

Power source – some robots use electric motors, others use hydraulic actuators. The former are faster, the latter are stronger and advantageous in applications such as spray painting, where a spark could set off an explosion; however, low internal air-pressure of the arm can prevent ingress of flammable vapours as well as other contaminants.

Drive – some robots connect electric motors to the joints via gears; others connect the motor to the joint directly (*direct drive*). Using gears results in measurable 'backlash' which is free movement in an axis. Smaller robot arms frequently employ high speed, low torque DC motors, which generally require high gearing ratios; this has the disadvantage of backlash. In such cases the harmonic drive is often used.

Compliance - this is a measure of the amount in angle or distance that a robot axis will move when a force is applied to it. Because of compliance when a robot goes to a position carrying its maximum payload it will be at a position slightly lower than when it is carrying no payload. Compliance can also be responsible for overshoot when carrying high payloads in which case acceleration would need to be reduced.

2.3 Standard robot types

The main characteristics of industrial robots are:

- Number of axes of motion
- Kinematic structure
- Work envelope
- Maximum payload
- Maximum speed
- Accuracy
- Drive train (actuators, remote vs. direct-drive).

Eight robot types are defined in the ISO standard 8373:1994 based on their kinematic structure and the coordinate frame that spans the workspace, Cartesian robots, cylindrical robots, spherical(or polar) robots, pendular robots, articulated (or anthropomorphic) robots, SCARA robots, spine robots and parallel robots. The preferred kinematic structure of industrial robots depends on the application at hand and is influenced by the required motion, payload, end-effector orientation, and other factors.

2.3.1 Articulate Robot

The articulate or jointed arm robot (or sometime called Anthropomorphic arms) closely resembles the human arm. The mechanical structure has at least three rotary

joints which forms a polar coordinate system. The Figure shows an articulate robot with 6 degree of freedom. The basic three rotary joints are Arm swivel, shoulder swivel and elbow rotations. Additional 3 revolute joints (Roll, Yaw, Pitch) and one prismatic joint allow the robot to point in many directions, and then reach out some radial distance.

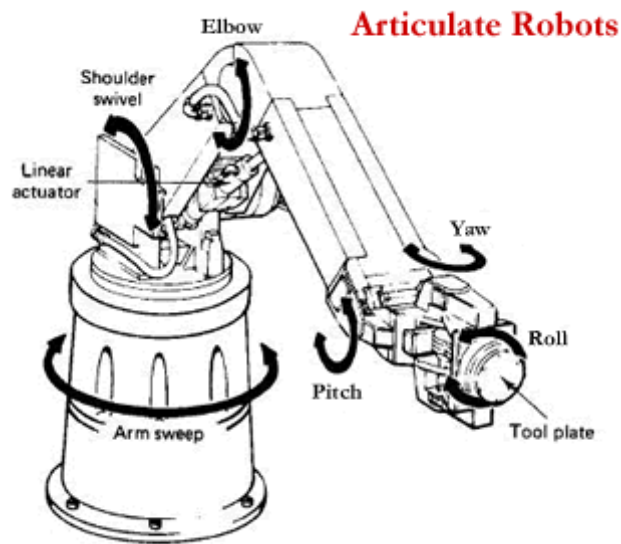


Figure 2.1 Articulated robot (Robotmatrix.org)

This structure is very flexible and has the ability to reach over obstructions. It can generally achieve any position and orientation within the working envelope. As such articulate robots are used for a wide range of applications including paint spraying, arc and spot welding, machine tending, etc. For examples, the articulate robot allows the welding torch to be manipulated in almost the same fashion as a human being would manipulate it. The torch angle and travel angle can be changed to make good quality welds in all positions. Articulate robots also allow the arc to weld in areas that are difficult to reach. In addition, articulate robots are compact and provide the largest work envelope relative to their size. Typical articulate robots have five or six free programmable arms or axes. As mentioned, the flexibility of the articulate robots makes them well suit for a wide variety of industrial application. But, it is not easy to control. When driving these robots in their natural co-ordinate system (joint space) the motion of the robot from one point to another can be difficult to visualize as the

robot will move each joint through the minimum angle required. This means that the motion of the tool will not be a straight line.

With different payload capability, reach and design, articulate robots are designed to employ in the following applications:

- Arc welding
- Spot welding
- Assembly
- cleaning/spraying
- Cutting
- Deburring
- Die casting
- Gluing/sealing
- Grinding/polishing
- Injection moulding
- Machine tending
- Material handling
- Packing
- Palletizing
- Picking
- Pre-machining
- Press brake tending

2.3.2 Cylindrical robots or spherical robots

A robot, which is used to control cylindrical mechanism and cylindrical coordinate system is called cylindrical robot. Cylindrical robot can work in four degree mechanism.

Advantages

- Simple kinematic model
- Easy to visualize
- Good access into cavities and machine openings
- Very powerful when hydraulic drives used

Disadvantages

- Restricted work space
- Prismatic guides difficult to seal from dust and liquids
- Back of robot can overlap work volume



Figure 2.2 Cylindrical robot (R19 ATHENA series, ST ROBOTICS)

2.3.3 Cartesian robots

A cartesian coordinate robot (also called linear robot) is an industrial robot whose three principal axes of control are linear (i.e. they move in a straight line rather than rotate) and are at right angles to each other. Among other advantages, this mechanical arrangement simplifies the Robot control arm solution. Cartesian coordinate robots with the horizontal member supported at both ends are sometimes called Gantry robots. They are often quite large.

A popular application for this type of robot is a computer numerical control machine (CNC machine). The simplest application is used in milling and drawing machines where a pen or router translates across an x-y plane while a tool is raised and lowered onto a surface to create a precise design.



Figure 2.3 Cartesian robot (Lexium MAX, Schneider Electric)

Advantages:

- Linear motion in three dimensions
- Simple kinematic model
- Rigid structure
- Easy to visualize

- Can use inexpensive pneumatic
- Drives for pick and place operation

Disadvantages:

- Requires a large volume to operate in
- Work space is smaller than robot volume
- Unable to reach areas under objects
- Guiding surfaces of prismatic joints
- Must be covered to prevent ingress of dust

Applications:

- Controlling cylindrical coordinate system
- Handling at machine tools
- Gearing and belt system ,
- Assembly operations
- Grinders and grinding machines
- Spot welding
- Wind Tunnel
- die-casting machines

CHAPTER THREE

INTEGRATED DYNAMIC ANALYSIS

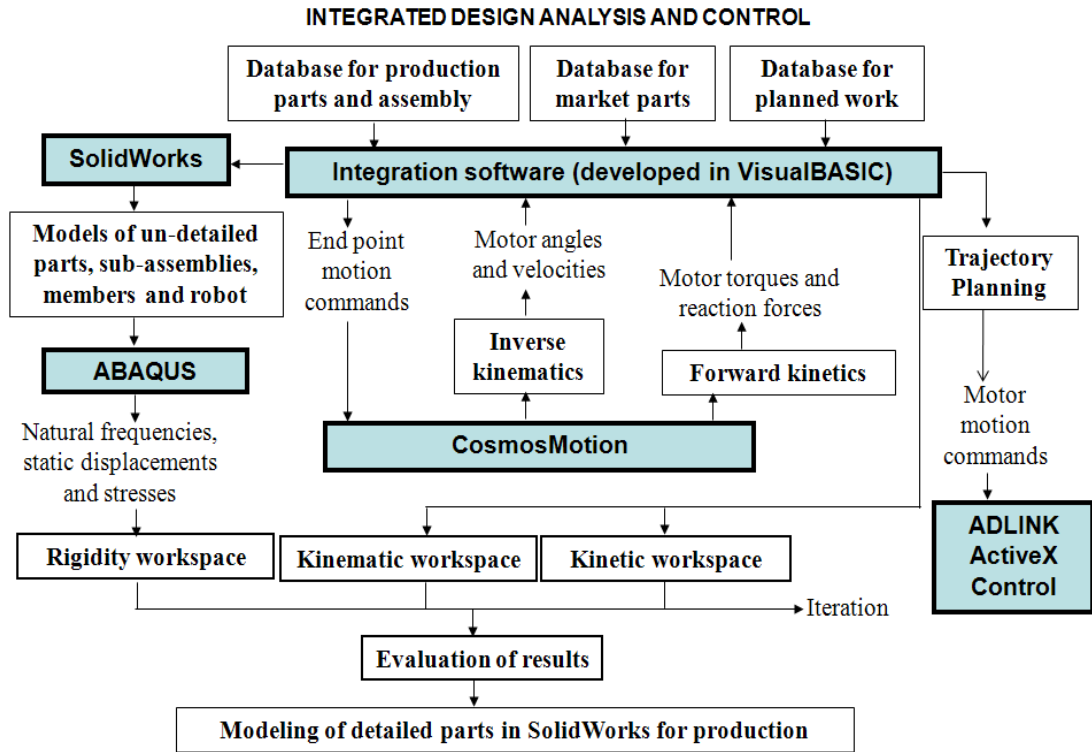


Figure 3.1 Flow chart of integrated design process

3.1 Flow Chart

The flow chart of the integrated design process developed in this study and applied to a six axis serial robot is shown in Fig. 3.1. The flow chart explains the processes of the robot from design to production.

Three main data blocks are created to develop the integration software (IS) in Visual Basic, which are the database for production parts and assembly, the database for market parts and the database for planned works. The database for production parts and assembly block includes the information of dimensions and angles which are necessary for modeling. The database for market parts block includes the

technical specifications of the actuation and load limits. The database for planned work block includes the input of the end-point motion.

The modeling software developed in Visual Basic reads the data from the databases and models the parts, sub-assemblies, members and the robot automatically, using the Application Programming Interface (API) capabilities of SolidWorks. SolidWorks + CosmosMotion + CosmosWorks package is used in this study. The robot model developed in SolidWorks is used automatically in CosmosMotion and CosmosWorks. Solid modeling of parts and assembly can also be done using the Graphical User Interface (GUI) capabilities of SolidWorks without the modeling software.

The interface has been developed in Visual Basic to analyze inverse kinematics and forward kinetics. IS reads the database for planned works, defines the velocity profile according to the parameters such as max. velocity and total time, and sends the end-point time based velocity profile as data points to CosmosMotion using API, when “Inverse Kinematics” option is clicked. The actuator’s motion parameters are set to free for inverse kinematics analysis and the actuator displacements and velocities are found. The time histories of the actuator displacements and velocities are transferred back to IS. The results are evaluated and shown on the IS whether the robot is reachable in its workspace and the percentage of each motor velocity. If the motion is possible, the IS shows the button as visible to pass to the other step “Forward Kinetics”.

The end-point motion is set to free and the actuator motion commands together with the inputs for the end-point work forces are sent to CosmosMotion, when “Forward Kinetics” option is clicked. The time histories of the actuator motion generator forces and reaction forces are found. The results are evaluated whether the kinetic limits (the actuator motion generation and joint force limits) are exceeded.

After the rigid body dynamics analyses are done, the finite element (FE) natural frequency and static displacement and strength analyses are performed in ABAQUS.

ABAQUS does not have API capabilities in the software package used in this study and these analyses are performed using GUI. The lowest natural frequency (f_{min}), maximum displacement (u_{max}) and maximum Von Mises stress (s_{max}) are evaluated whether the rigidity limits of the robot structure are exceeded. The higher values of f_{min} , and the lower values of u_{max} and s_{max} indicate higher static and dynamic rigidity. The precision of the work performed by the robot is related to these rigidity values.

The database for planned works are arranged for various works and simulations are performed for the kinematic, kinetic and rigidity workspace evaluations. The databases for modeling are changed and the simulations and evaluations are repeated until a final design is achieved.

Table 3.1 Design parameters in simulation

Inverse Kinematics	Reachability Velocity Limits
Forward Kinetics	Torque Limit Force Limit
Rigidity	Maximum Von Mises Stress Maximum Displacement Lowest Natural Frequency

The details such as holes, threads, tolerances are inserted to the un-detailed parts in SolidWorks after the final design, using GUI. Market parts usually have the detailed models, which can be downloaded from the manufacturer websites. Un-detailed models of market parts are produced by the designer. Un-detailed models must have equivalent rigid body dynamic properties and approximately the same rigidity in FE analyses.

After the actual robot and control unit are produced, the actuator motion commands are sent by IS to the actuator motion control unit using Adlink motor

control module, when “Move Traj.” option is clicked. This option is activated when the inverse kinematic and forward kinetic analysis are passed.

The integration software (IS) is used at the application stage when the actual robot is in use as well as at the design stage. IS is useful for the optimum usage of the robot as well as for the optimum design. The rigidity workspace evaluation takes time and may be omitted during the application if the end-load limits are not exceeded.

The limits are read by IS from text document which are based on real system such as motor torques, motor velocities, bearing forces. Limits are listed as shown below (Table 3.2).

Table 3.2 Limits for actual robot parameters

	Angle (degree)	Velocity (deg/s)	Motor Torque (Nm)	Bearing Force (N)
Motor-1	90	150	192	14186
Motor-2	80	120	478	14186
Motor-3	80	120	130	10000
Motor-4	90	150	51.2	4227
Motor-5	90	120	32	4227
Motor-6	270	120	16	4058

3.2 Modeling of members of a six axis serial robot

The model of the six axis robot with un-detailed parts designed with the process given in Section 3.1 as shown in Fig. 3.1. The model of the six axis robot assembly consists of motion sub-assemblies which are called as members. Each member consists of models of parts or sub-assemblies which have a common rigid body motion. Members are connected by joints. The models of the members with un-detailed parts are shown in Fig. 3.2.

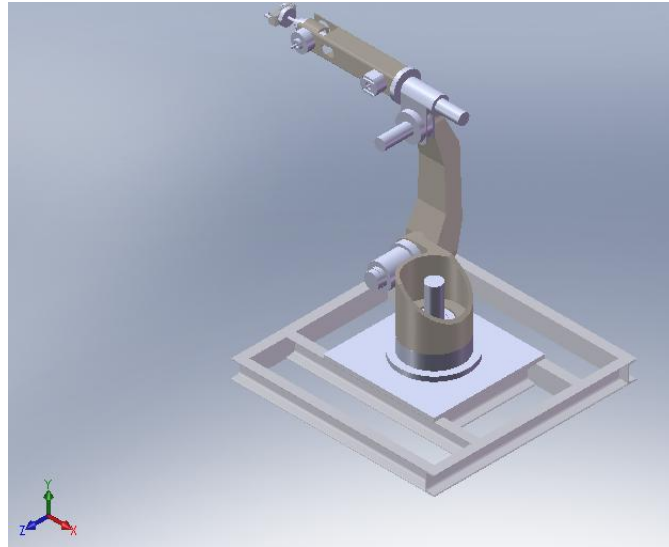


Figure 3.2 Model of the robot with un-detailed parts

The members of the six axis robot are titled as m1, m2, m3, m4, m5, m6 and gripper. The members are connected revolute joints. The sizes of the base platform are 1100x1100x50 mm. Its thickness is 10 mm. The end point is at a distance of 1655 mm to the axis of the first motor when the robot is at the maximum position in its workspace. The materials selected for all the parts are steel (AISI 1020). The weight of the six axis robot model is 235 kg. The rotary motors (Mitsubishi Model HC-KFS053, HC-KFS13, HC-KFS23, HC-KFS43, HC-KFS7 and HC-SFS102) and Harmonic Drive Gear (HFUC-14-2UH, HFUC-20-2UH, HFUC-32-2UH and HFUC-40-2UH) are market parts. The gear ratios are 80, 100, 100, 100, 80, 100 for Motor-1, Motor-2, Motor-3, Motor-4, Motor-5 and Motor-6, respectively. The actuators have 3.43×10^{-5} deg/pulse and 2.74×10^{-5} deg/pulse, depend on gear ratios, precision.

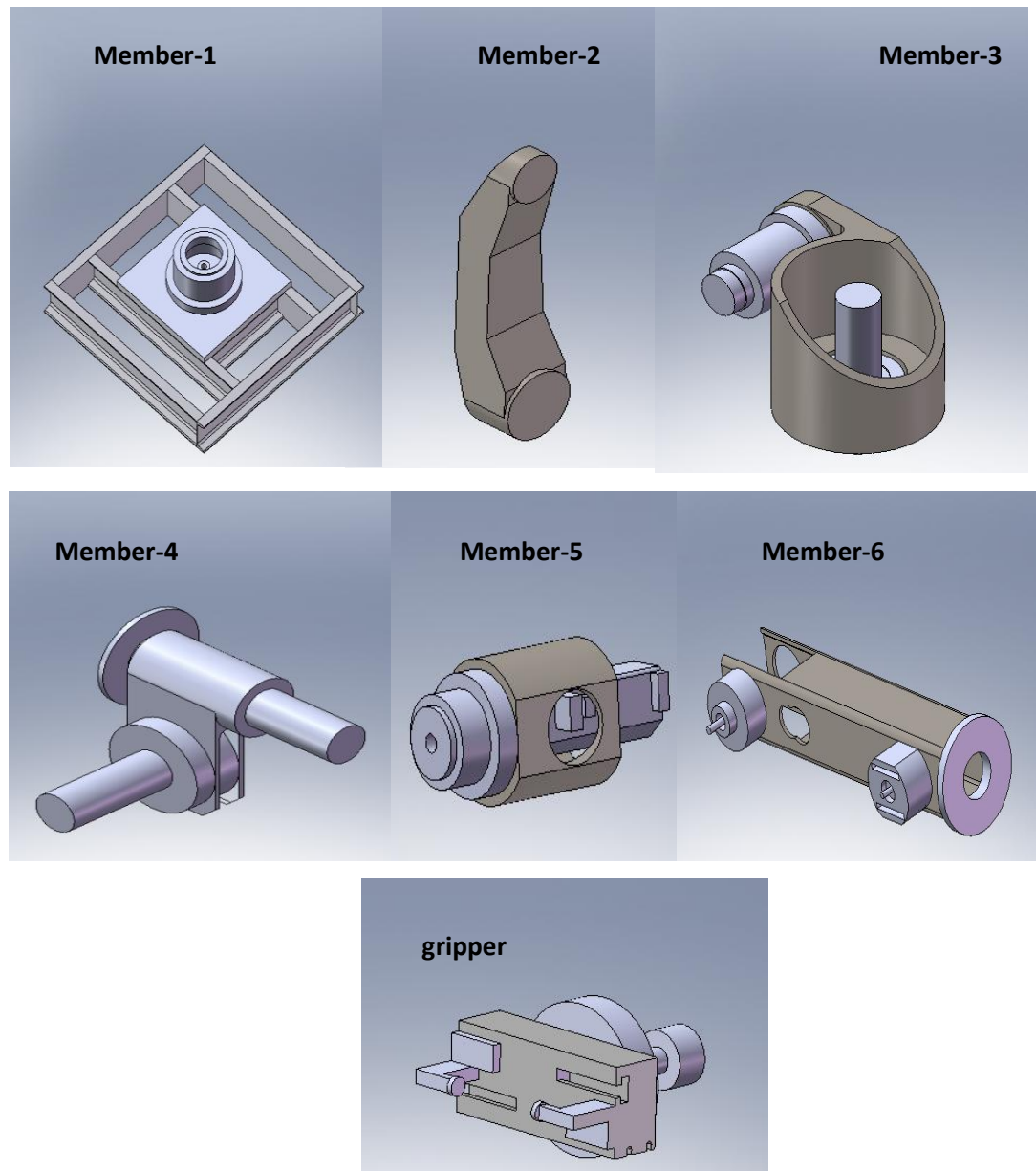


Figure 3.3 Members of the robot

The six axis robot can be modeled by using GUI after starting the recording of Visual Basic macros in SolidWorks. Then, the macros can be examined to determine the commands to develop codes for the modeling software, which uses API. The software performs the following tasks with sub-routines. First, the un-detailed parts are modeled using the input database. Then the parts are inserted to the assembly file for the member to be modeled. The parts are located to their positions. No mate commands are used for the modeling of the members. After modeling all the

members, the assembly files of the members are inserted to the assembly file of the six axis robot. The members are oriented and located to their approximate positions. The mates are defined between the members for joints. CosmosMotion defines revolute joints automatically after the ground and moving members are defined. The revolute joints on the linear actuator parts have one degree of freedom titled as RotateZ.

3.3 Integration Software (IS)

3.3.1 Database for Planned Work

The position of the robot is defined by the position of the moving end-point in Member-5. The incremental position vector of the end-point is defined as $\mathbf{q}_{ei} = [x_i, y_i, z_i, \theta_{xi}, \theta_{yi}, \theta_{zi}]^T$. x_i , y_i , and z_i are the incremental displacements and θ_{xi} , θ_{yi} and θ_{zi} are the incremental rotation angles of the end-point from its previous position in the x, y, and z directions, respectively. The time interval for an incremental displacement is defined as $\mathbf{t}_i = [t_1, t_2]$, where t_1 is the starting time and t_2 is the ending time for the incremental motion. The distances are given in mm, times are given in seconds, unless otherwise stated.

The motor rotation vector is defined as $\dot{\mathbf{q}}_{mi}(t) = [\dot{\theta}_{1i}(t), \dot{\theta}_{2i}(t), \dot{\theta}_{3i}(t), \dot{\theta}_{4i}(t), \dot{\theta}_{5i}(t), \dot{\theta}_{6i}(t)]^T$. $\dot{\theta}_{1i}(t)$, $\dot{\theta}_{2i}(t)$, $\dot{\theta}_{3i}(t)$, $\dot{\theta}_{4i}(t)$, $\dot{\theta}_{5i}(t)$ and $\dot{\theta}_{6i}(t)$ are the time histories of the motor velocities for the axes 1, 2, 3, 4, 5 and 6, respectively. The starting position of the end-point is defined as $\mathbf{q}_{es} = \mathbf{q}_{ea} + \mathbf{q}_{is}$, where the first term is the assembly position vector. The second term is the incremental displacement vector for the starting position, which is defined as $\mathbf{q}_{is} = [x_{is}, y_{is}, z_{is}]^T$. The assembly position is given as $\mathbf{q}_{ea} = [-944.44, 1003.52, -163.74]^T$.

The database for a planned task contains the information for the end point, $\mathbf{q}_{es}$; and $\mathbf{q}_{ei}$ and $\mathbf{t}_i$ for motion steps. An example motion input may be given as

0,4,-400,-800,0,0,0,0

Here, $t_1=0$ s starting time, $t_2=4$ s ending time, $x_1=-400$ mm, $y_1=-800$ mm, $z_1=0$ mm, $\theta_{x1}=0^\circ$, $\theta_{y1}=0^\circ$ and $\theta_{z1}=0^\circ$. The end-point motion follows the trapezoidal velocity profile which is created in IS, given in Fig. 3.5.

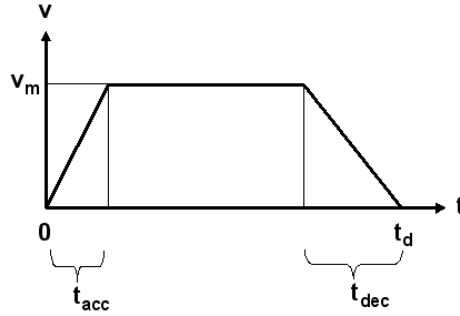


Figure 3.4 Trapezoidal velocity profile

3.3.2 Inverse Kinematics and Kinematic Workspace Evaluation in IS

In the inverse kinematics analysis, the incremental position vector $\mathbf{q}_{ei}$ is given and the time histories of motor velocities $\dot{\mathbf{q}}_{mi}(t)$ are found. The RotateZ components of all motors are set to “Free” for the inverse kinematic analysis. A motion object is assigned to the end point at Member-5. The location of the object is at the end-point, and the variable name “motione” is assigned to the motion object. motione has 6 degrees of freedom titled as TranslateX, TranslateY, TranslateZ, RotateX, RotateY, and RotateZ. The TranslateX, TranslateY, TranslateZ, RotateX, RotateY, and RotateZ components are set to the “Velocity” mode in motion type and “Spline” function to enter the data points of velocity profile. An example time history of the velocity profile which is created in IS, is given below.

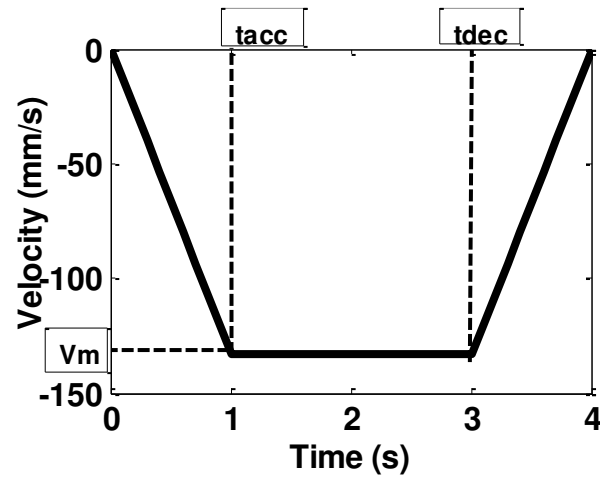


Figure 3.5 Trapezoidal velocity profile for displacement x

Table 3.3 Example motion input

$\mathbf{q}_{es}$	Step-1	
	$\mathbf{t}_i$	$\mathbf{q}_{ei}$
$[172.84, 1010.26, -164.14]^T$	[0,4]	[-400,-800,0,0,0,0]

Plot objects for the motor rotations are defined to observe the results. The results can be exported to the files and can be read. The time histories of motor angles and velocities are transferred to Visual Basic after the inverse kinematic analysis in CosmosMotion. Motor-3 angle and velocity time histories are shown in Fig. 3.6 for the path given in Table 3.3.

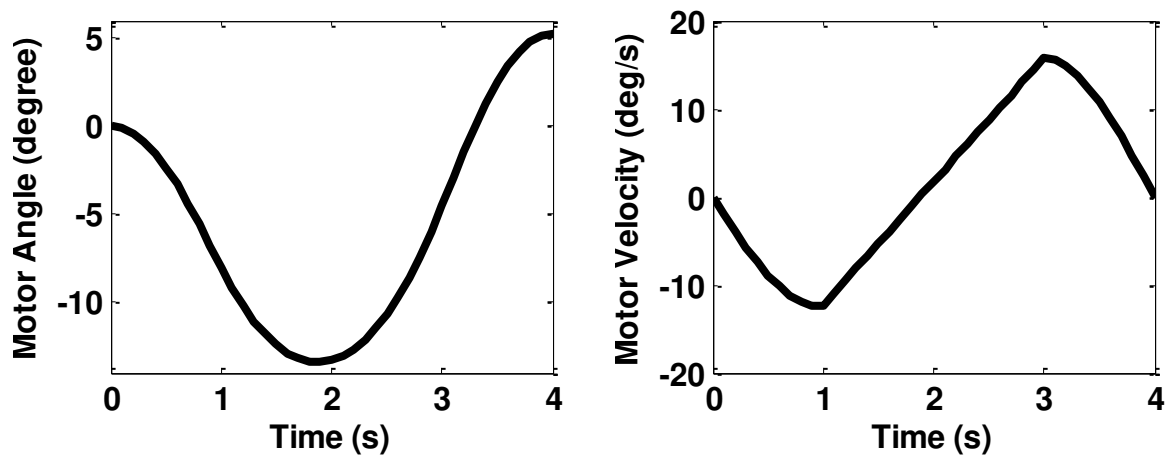


Figure 3.6 Motor-3 a) angle and b) velocity time histories (for path in Table 3.3)

The max. motor angles are determined and evaluated by the IS whether they exceed the limits. If the motor angles are within the limits, it means the robot is in the workspace, and the reachability button turns to green color in the kinematic evaluation chart shown in Fig 3.7. The red color appears if the task is not in the workspace. The angle limits are $\pm 90^\circ$, $\pm 80^\circ$, $\pm 80^\circ$, $\pm 90^\circ$, $\pm 90^\circ$ and $\pm 270^\circ$ for Motor-1, Motor-2, Motor-3, Motor-4, Motor-5 and Motor-6, respectively.



Figure 3.7 Results chart for evaluating kinematic parameters in IS
(Results are given for the path given in Table 3.3)

The motor velocities are checked whether they exceed the limits. Similarly, these buttons turn to the green or red colors. The percentages of the maximum motor velocities to the motor velocity limits are indicated in the chart. They must be below 100. The velocity limits are 150 deg/s, 120 deg/s, 120 deg/s, 150 deg/s, 120 deg/s and

120 deg/s for Motor-1, Motor-2, Motor-3, Motor-4, Motor-5 and Motor-6, respectively.

3.3.3 Forward kinetics in IS

All the components of the end-point motion (motione) are set to “Free” and the RotateZ components of Motor-1, Motor-2, Motor-3, Motor-4, Motor-5 and Motor-6 are set to the “Velocity” mode in motion type and “Spline” function to enter the data points of velocity profile in the forward kinetic analysis. The time histories of $\dot{\theta}_{1i}$, $\dot{\theta}_{2i}$, $\dot{\theta}_{3i}$, $\dot{\theta}_{4i}$, $\dot{\theta}_{5i}$ and $\dot{\theta}_{6i}$ found in the inverse kinematic analysis are assigned to the corresponding RotateZ components. Plot objects are defined to observe the kinetic outputs such as motor torques and reaction forces. An example time history of Motor-3 torque is shown in Fig 3.8 for the path given in Table 3.3.

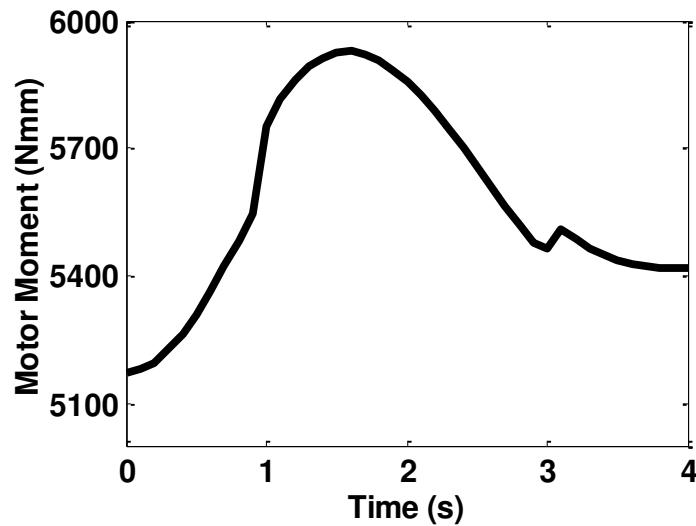


Figure 3.8 Motor-3 torque time history (for path in table 3.3)

The percentages of the maximum reaction forces and motor torques to the corresponding limit values are shown in the chart (Figure 3.9). The reaction force limits are 14186 N, 14186 N, 10000 N, 4227 N, 4227 N and 4058 N for the motor bearings-1, motor bearings-2, motor bearings-3, motor bearings-4, motor bearings-5 and motor bearings-6, respectively. The torque limits are 192 Nm, 478 Nm, 130 Nm, 51.2 Nm, 32 Nm and 16 Nm for Motor-1, Motor-2, Motor-3, Motor-4, Motor-5 and Motor-6, respectively. The results are shown with vertical colored lines. The line is

shown in green color if the percentage is less than 50. The brown color is used if the percentage is between 50.1 -100. The red color is used if the percentage is above 100, which indicates that the task is out of kinetic workspace.

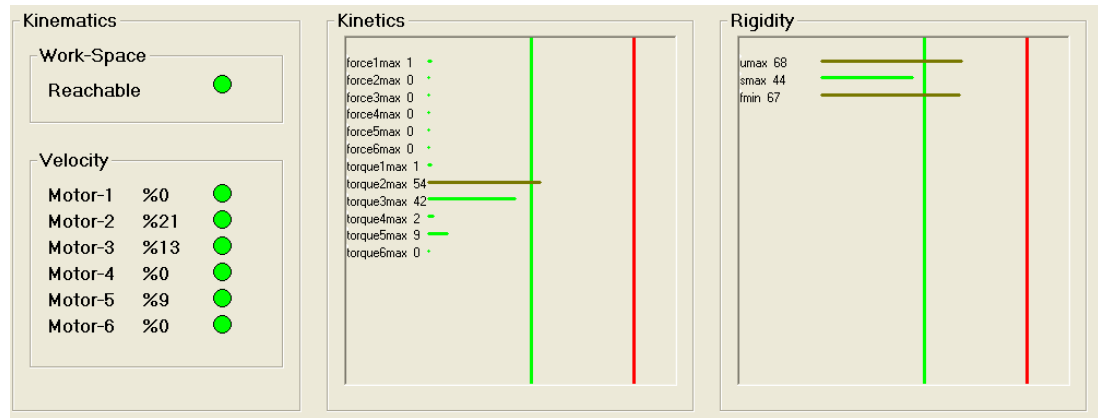


Figure 3.9. Results chart for evaluating kinematic, kinetic and rigidity parameters in IS (Results are given for the path given in Table 3.3)

CHAPTER FOUR

SIMULATION RESULTS

4.1 Case Studies

In this study, three cases are considered for moving the robot from starting position to ending position. It was supposed that there is an obstacle on the robot's motion trajectory and therefore robot will follow three different paths to reach the ending position. For all of the three cases, robot will start from the same starting position and reach to the same ending position.

Case 1: In this case the path is generated to pass over the obstacle as shown in Fig. 4.1. The database for planned work is given in Fig. 4.2.

Case 2: In this case the path is generated to pass under the obstacle as shown in Fig. 4.9. The database for planned work is given in Fig. 4.10.

Case 3: In this case the path is generated to pass behind the obstacle as shown in Fig. 4.17. The database for planned work is given in Fig. 4.18.

4.2 Case 1

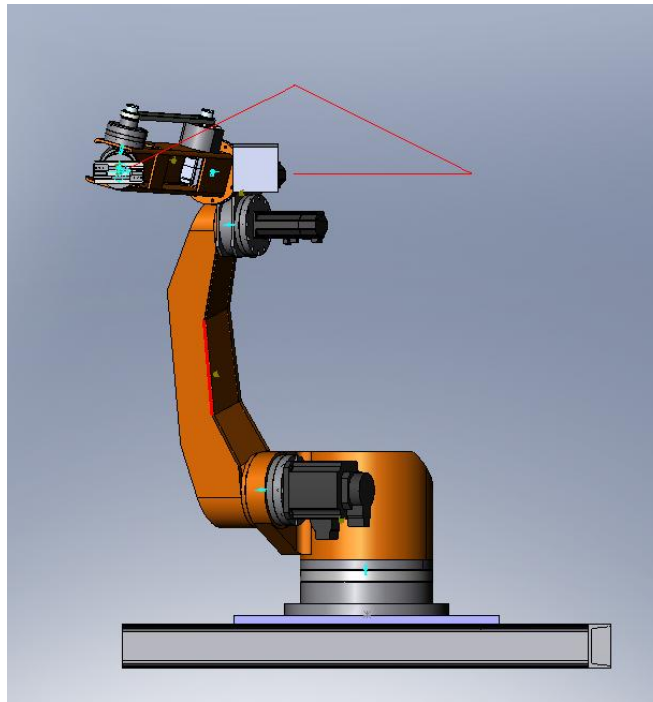


Figure 4.1 Case-1

Table 4.1 Coordinates of start and end points of the paths

	S.P	E.P
x	-945	-945
y	1000	1000
z	237	-563

S.P: start point of the path

E.P: end point of the path

```

nsim,0.1
0,0,0

inverse
0,0,0
0,3,0,0,400,0,0,0
4,6,0,200,-400,0,0,0
6,8,0,-200,-400,0,0,0
-1

```

Figure 4.2 Database for
planned work in case 1

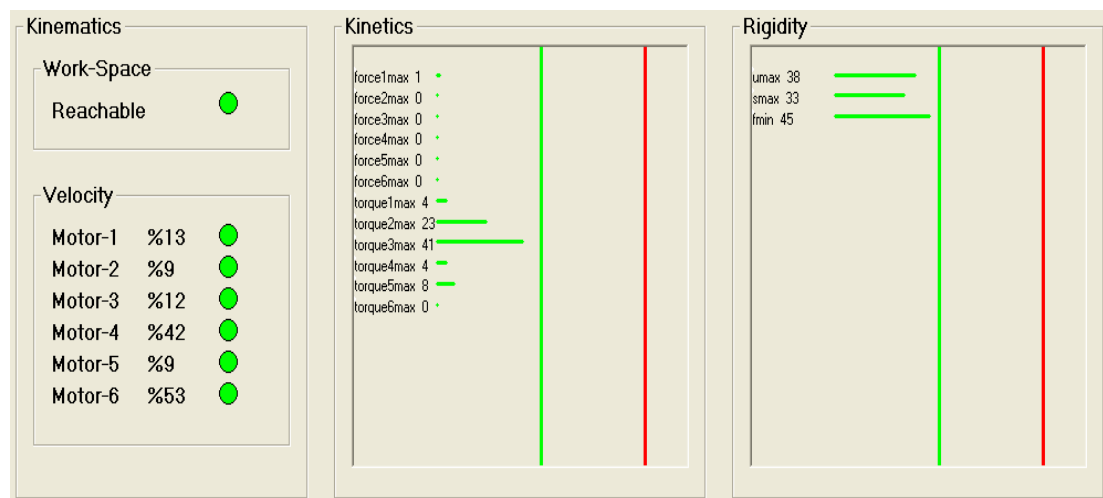


Figure 4.3 Simulation result for case 1

Robot is in workspace and all motor velocities and motor moments do not exceed limits. It's obviously seen that force values given by SolidWorks are too small compared to bearing forces on the actual system. In other words, the harmonic drives on the actual robot are selected quite large and it just caused rising the costs.

4.2.1 Simulation results for dynamic F.E ABAQUS analysis for case 1

The path in case 1 is executed and the displacements for end point of the effector are graphed in ABAQUS. Due to having rigid analysis in SolidWorks, the displacements for rigid body motion in SolidWorks are subtracted the displacements for rigid + flexible motion in ABAQUS. Thus, the displacements for flexible motion of the end effector are found and graphed as shown in Fig. 4.4. The Von Mises

stresses are evaluated along the path and max. Von Mises stress is found in ABAQUS as shown in Fig. 4.8.

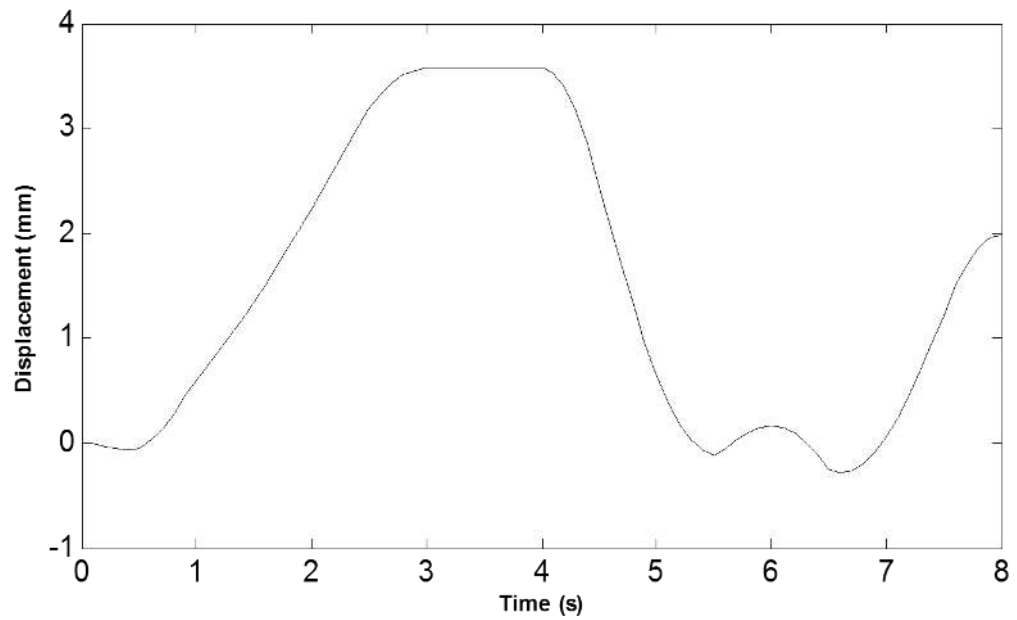


Figure 4.4 x- displacement for rigid + flexible motion graphed in ABAQUS

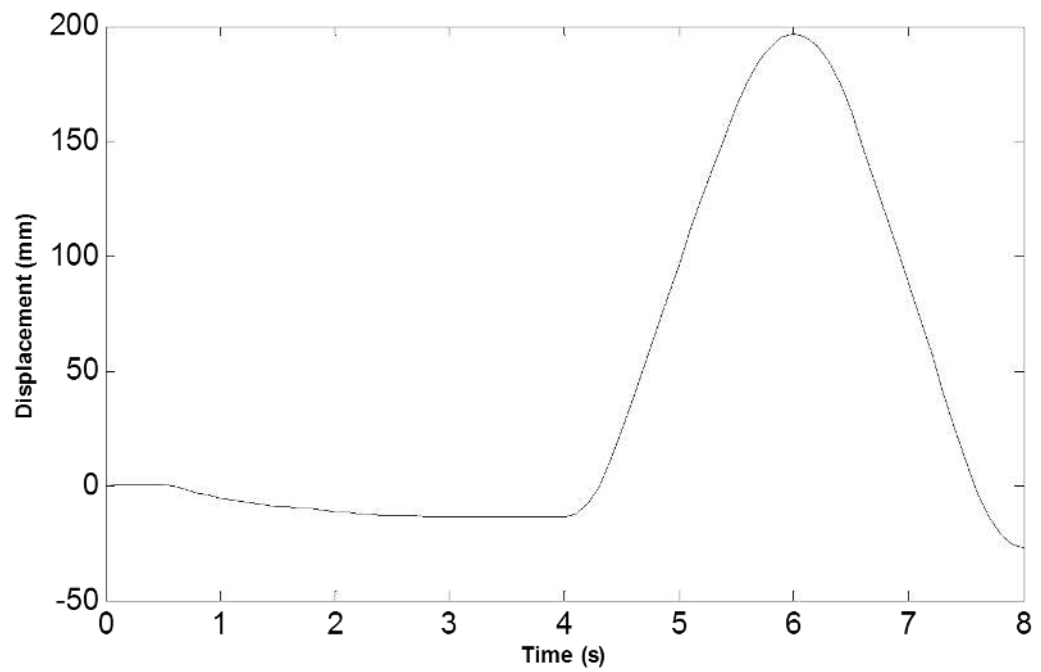


Figure 4.5 y- displacement for rigid + flexible motion graphed in ABAQUS

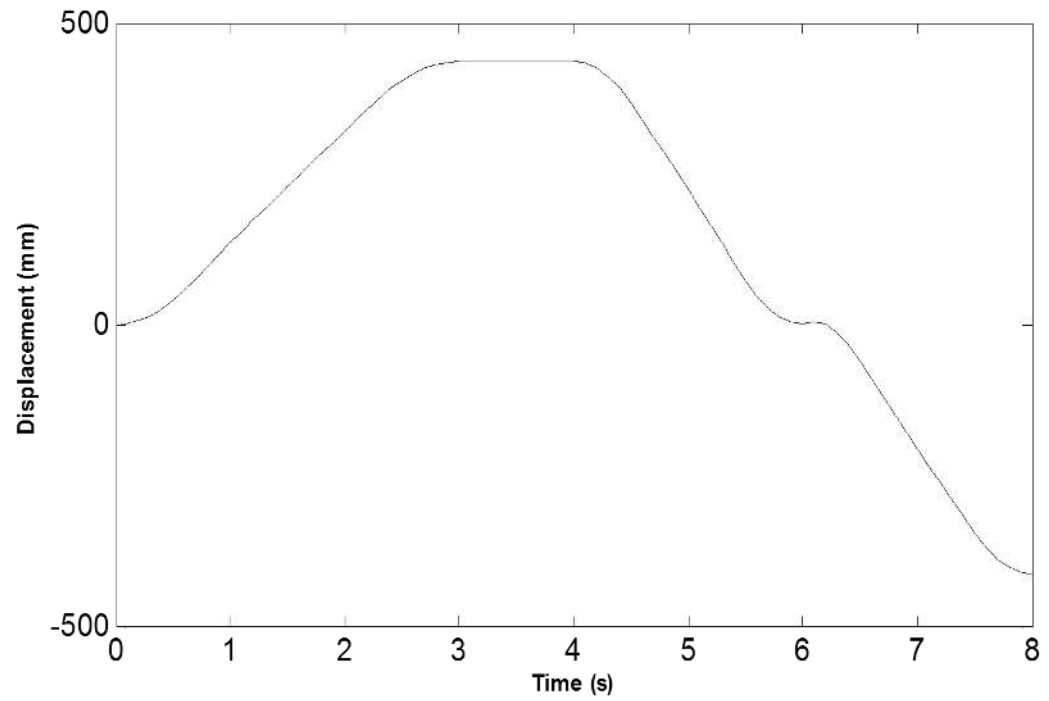


Figure 4.6 z- displacement for rigid+flexible motion graphed in ABAQUS

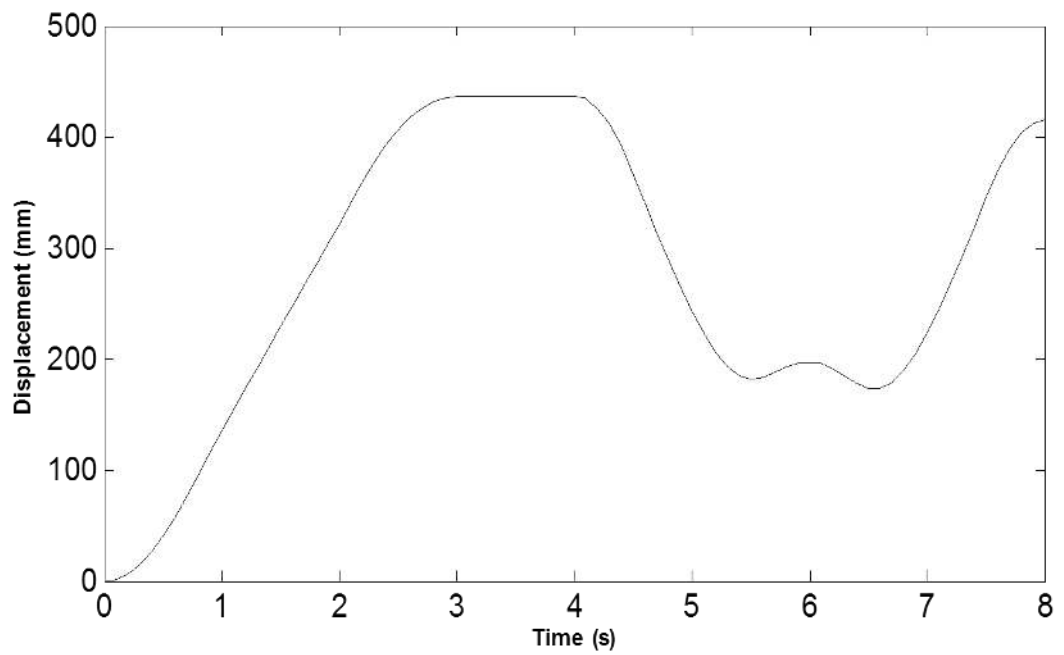


Figure 4.7 Magnitude of the displacements for rigid + flexible motion graphed in ABAQUS

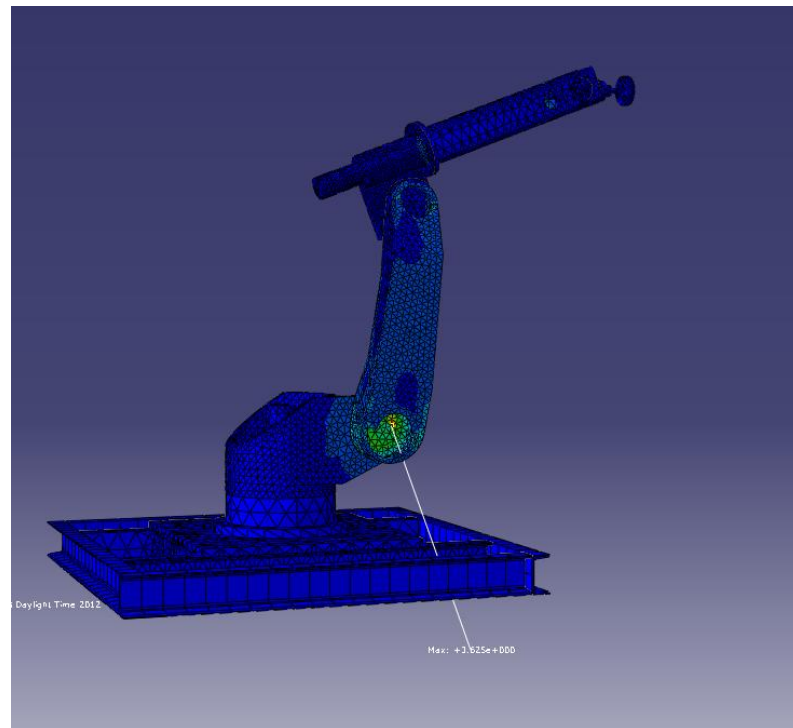


Figure 4.8 Max von mises stress for case 1

Max stress during the simulation for case 1 is at the point as shown in Fig. 4.8 which is 3.625 Mpa. Max. displacement is 94 μ .

Table 4.2 Rigidity results

	Case-1
Smax (maximum von mises stress)	3.625 MPa
Umax (maximum displacement)	94 μ
fmin (lowest natural frequencies)	13.59 Hz

4.3 Case 2

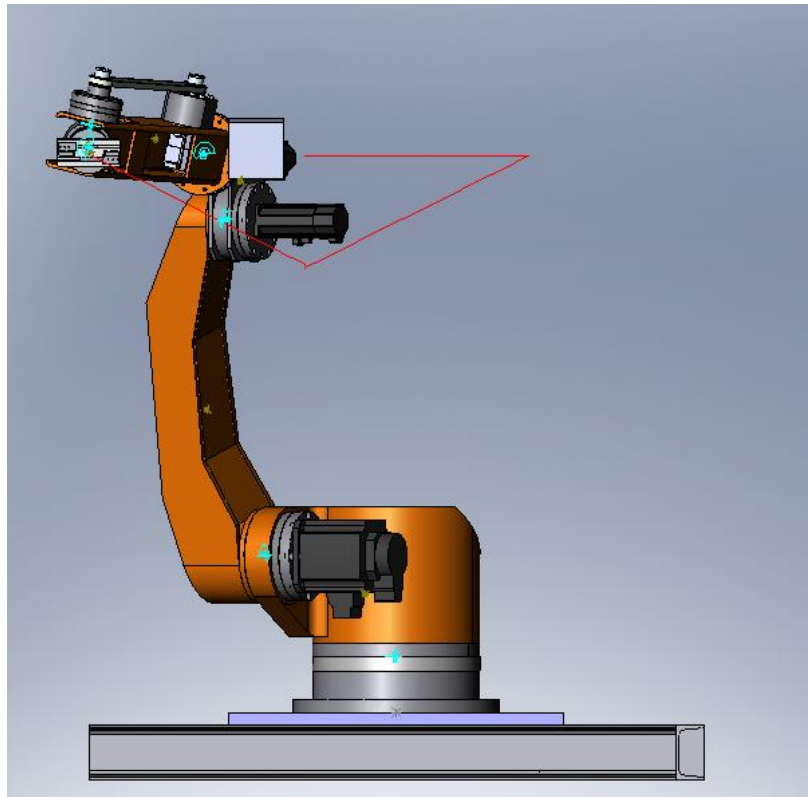


Figure 4.9 Case 2

Table 4.3 Coordinates of start and end points of the paths

	S.P	E.P
x	-945	-945
y	1000	1000
z	237	-563

S.P: start point of the path

E.P: end point of the path


```

nsim,0.1
0,0,0

inverse
0,0,0
0,3,0,0,400,0,0,0
4,6,0,-200,-400,0,0,0
6,8,0,200,-400,0,0,0
-1

```

Figure 4.10 Database for
planned work in case 2

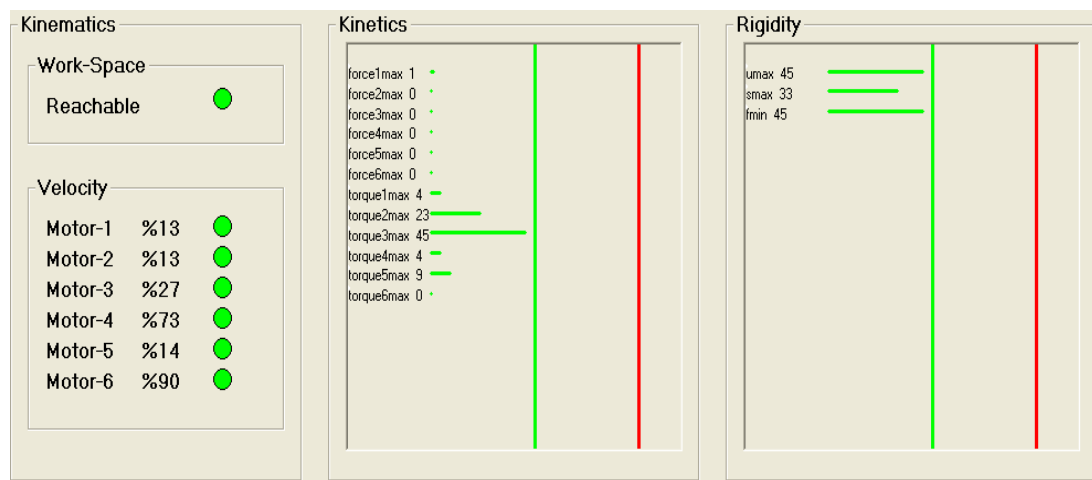


Figure 4.11 Simulation result for case 2

Robot is in workspace and all motor velocities and motor moments do not exceed limits. But it's seen that the velocity of motor-6 reached nearly the limit value. It's also obviously seen that force values given by SolidWorks are too small compared to bearing forces on the actual system. In other words, the harmonic drives on the actual robot are selected quite large and it just caused rising the costs.

4.3.1 Simulation results for dynamic F.E ABAQUS analysis for case 2

The path in case 2 is executed and the displacements for end point of the effector are graphed in ABAQUS. The Von Mises stresses are evaluated along the path and max. Von Mises stress is found in ABAQUS as shown in Fig. 4.16.

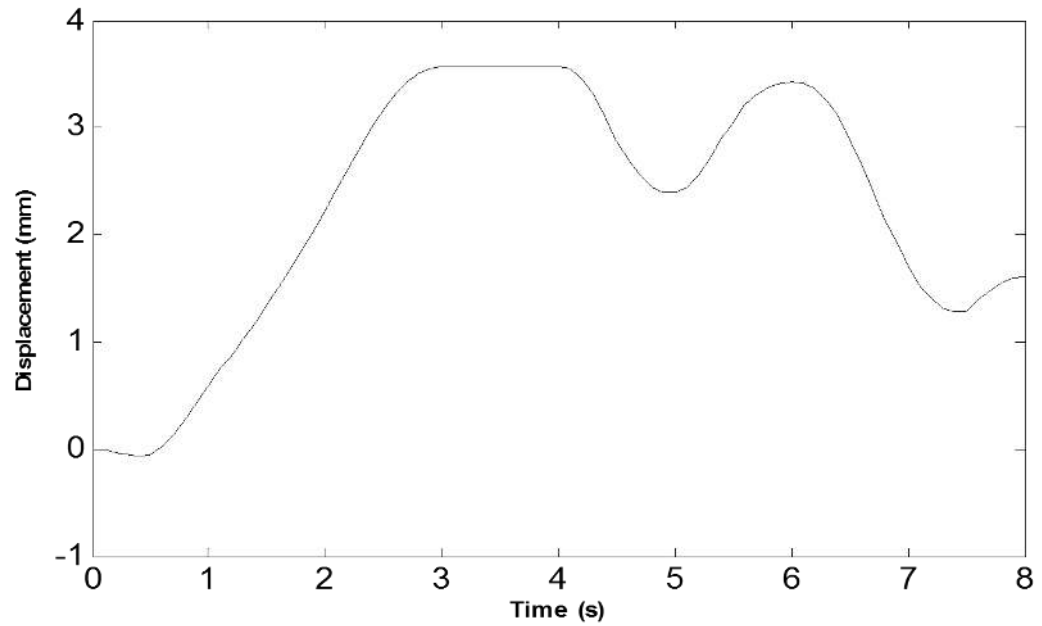


Figure 4.12 x- displacement for rigid + flexible motion graphed in ABAQUS

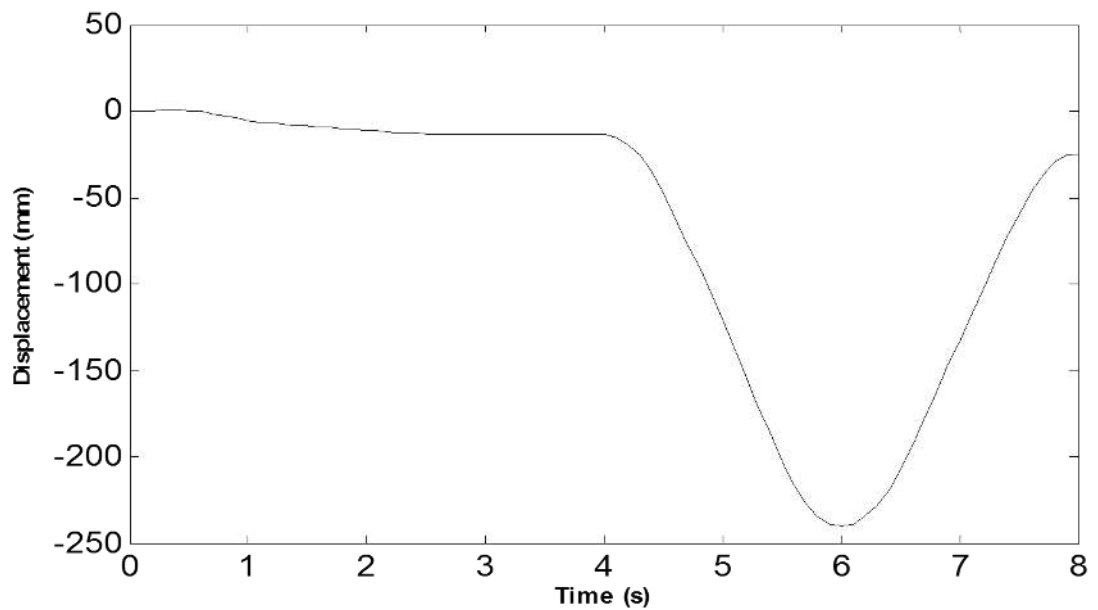


Figure 4.13 y- displacement for rigid + flexible motion graphed in ABAQUS

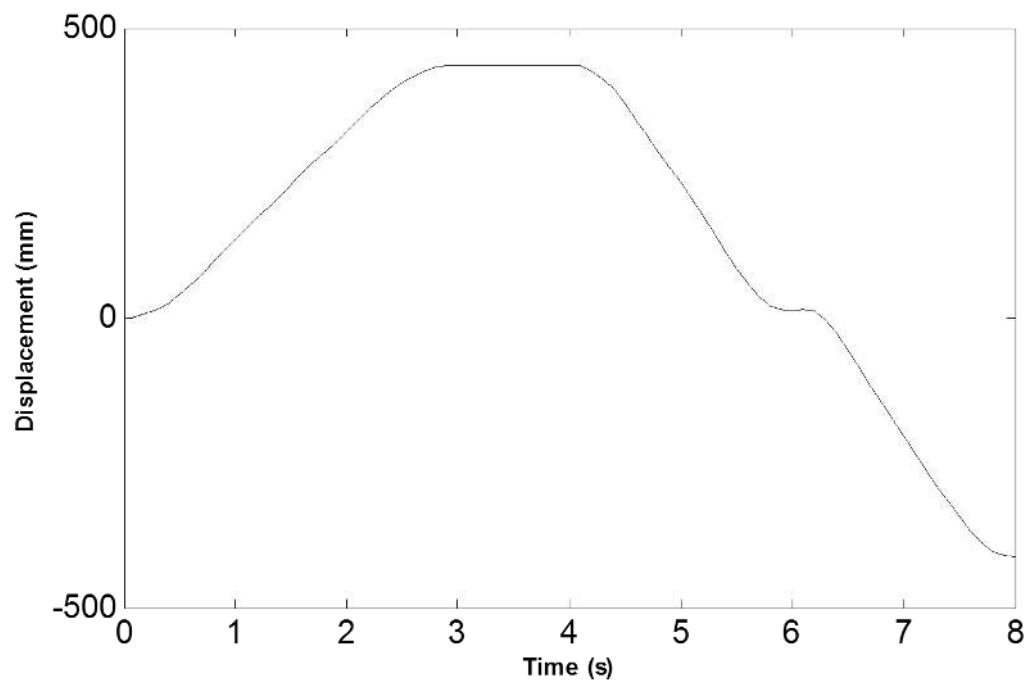


Figure 4.14 z- displacement for rigid + flexible motion graphed in ABAQUS

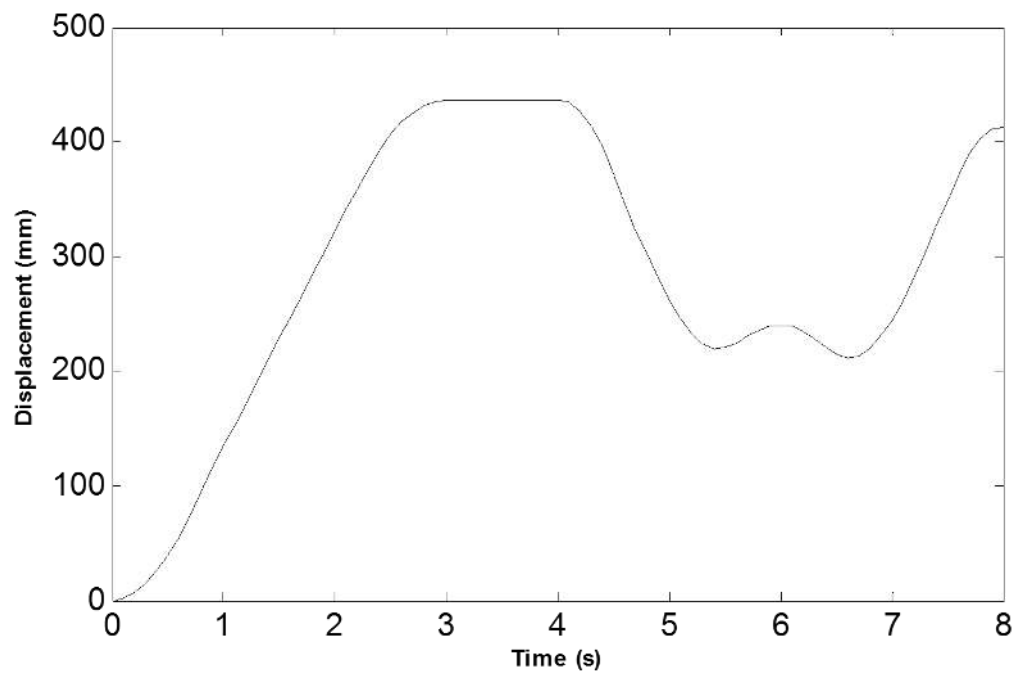


Figure 4.15 Magnitude of the displacements for rigid flexible motion graphed in ABAQUS

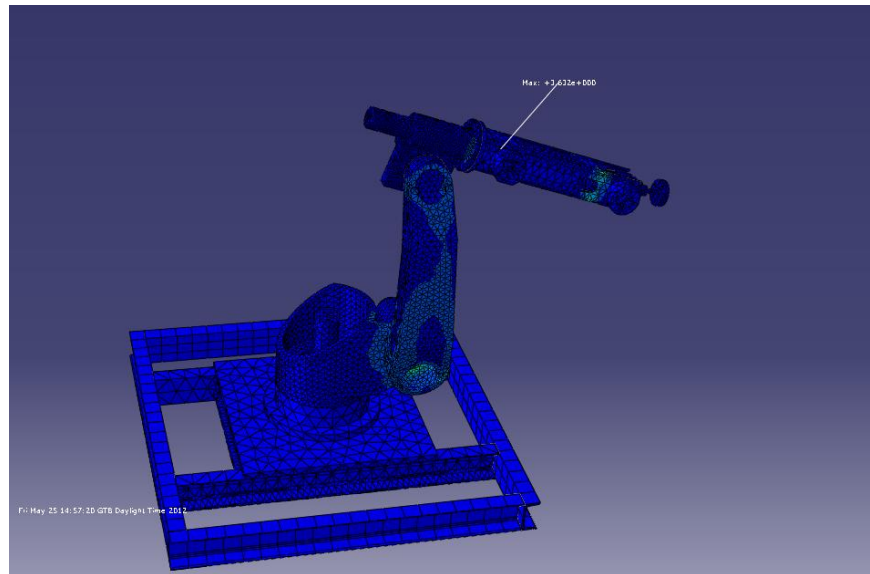


Figure 4.16 max von mises stress for case 2

Max stress during the simulation for case 2 is at the point as shown in Fig. 4.16 which is 3.632 Mpa. Max. displacement is 112 μ .

Table 4.4 Rigidity results

	Case-2
Smax (maximum von mises stress)	3.625 MPa
Umax (maximum displacement)	112 μ
fmin (lowest natural frequencies)	13.6 Hz

4.4 Case-3

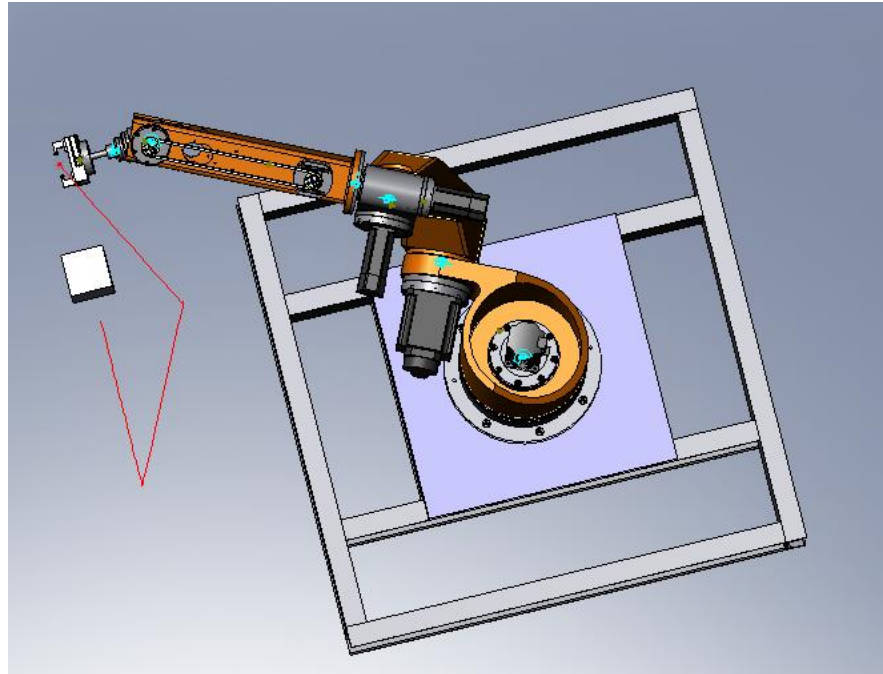


Figure 4.17 Case-3

Table 4.5 Coordinates of start and end points of the paths

	S.P	E.P
x	-945	-945
y	1000	1000
z	237	-563

S.P: start point of the path

E.P: end point of the path

```

nsim,0.1
0,0,0

inverse
0,0,0
0,3,0,0,400,0,0,0
4,6,200,0,-400,0,0,0
6,8,-200,0,-400,0,0,0
-1

```

Figure 4.18 Database for
planned work in case-3

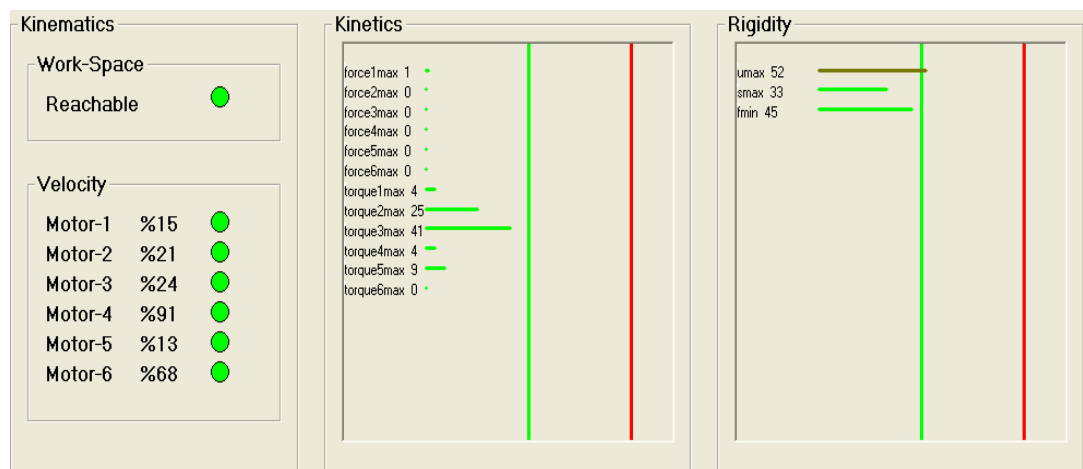


Figure 4.19 Simulation result for case-3

Robot is in workspace and all motor velocities and motor moments do not exceed limits. But it's seen that the velocity of motor-4 reaches nearly the limit value and max displacement exceeds the value of 50%. It's also obviously seen that force values given by SolidWorks are too small compared to bearing forces on the actual system.

When we consider all the three cases together, it's clearly seen that case-1 is optimum solution for the actual robot's trajectory plan.

4.4.1 Simulation results for dynamic F.E ABAQUS analysis for case 3

The path in case 3 is executed and the displacements for end point of the effector are graphed in ABAQUS. The Von Mises stresses are evaluated along the path and max. Von Mises stress is found in ABAQUS as shown in Fig. 4.24.

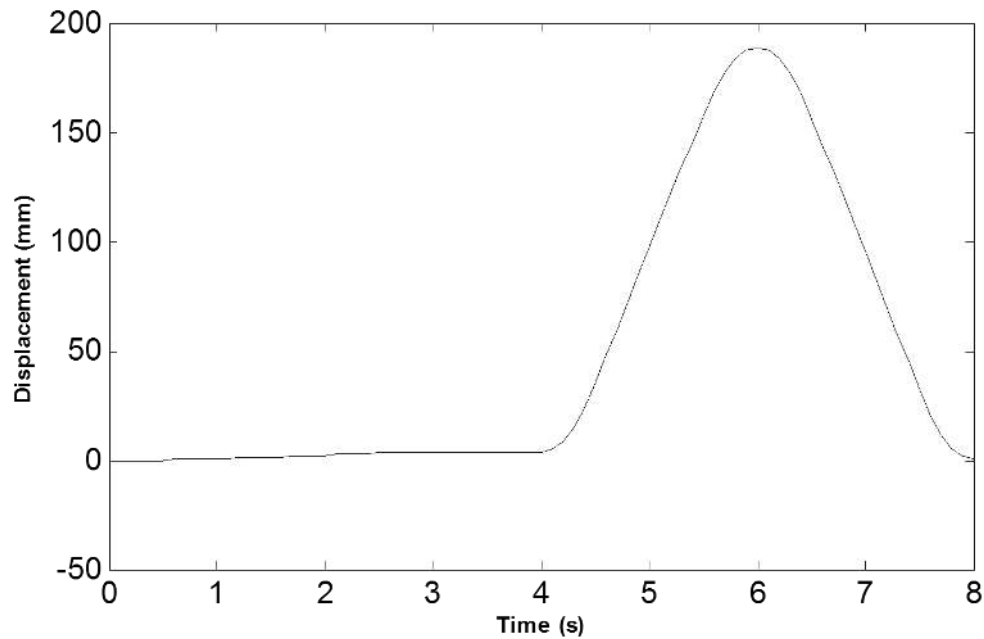


Figure 4.20 x- displacement for rigid + flexible motion graphed in ABAQUS

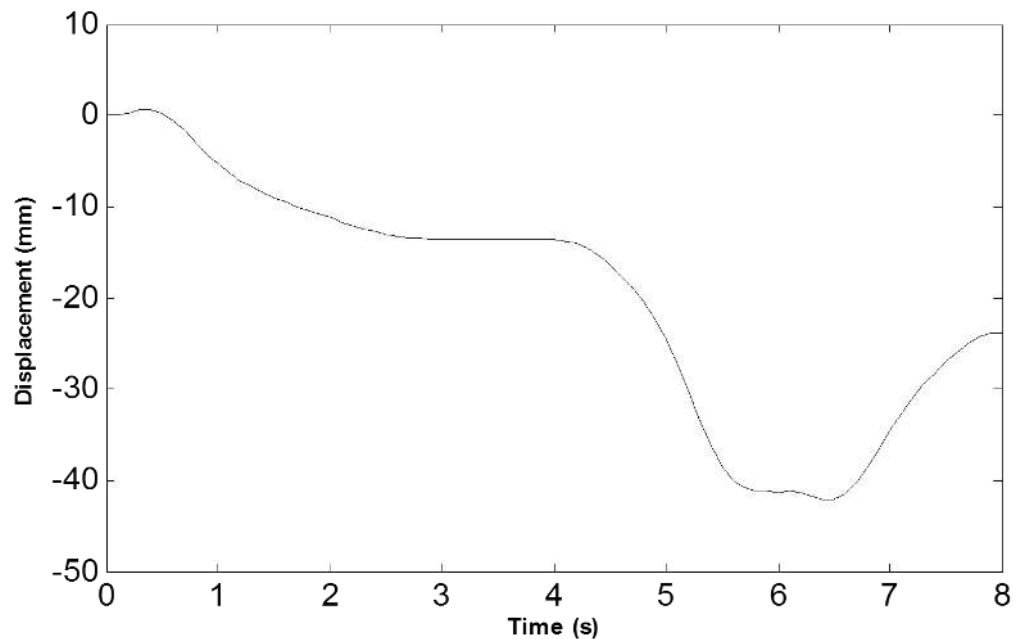


Figure 4.21 y- displacement for rigid + flexible motion graphed in ABAQUS

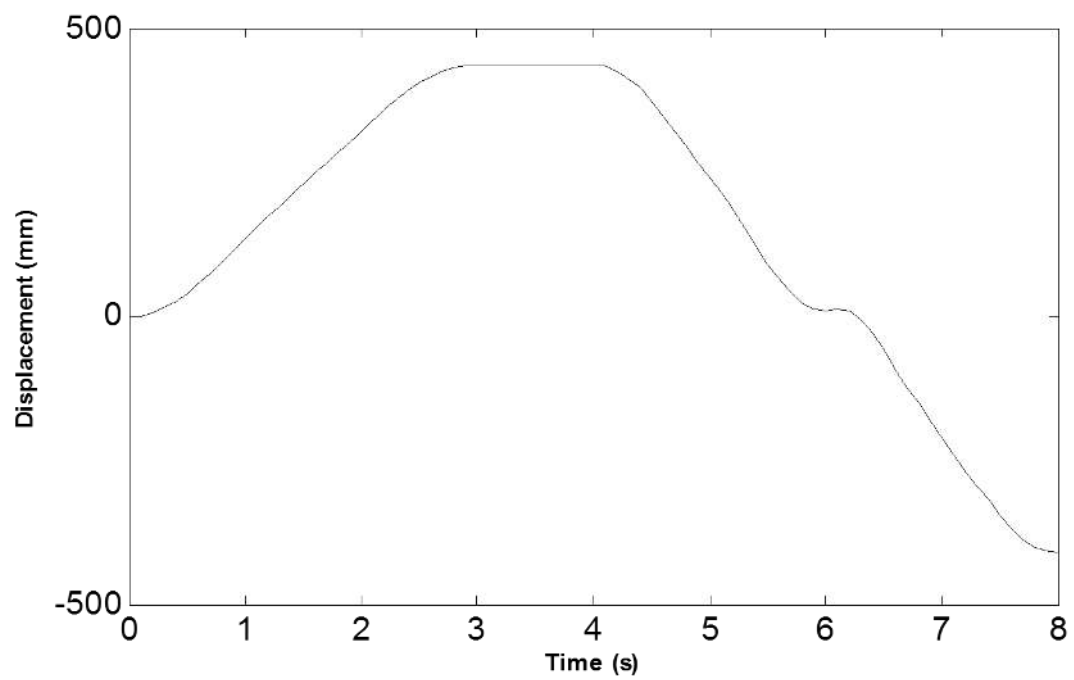


Figure 4.22 z- displacement for rigid + flexible motion graphed in ABAQUS

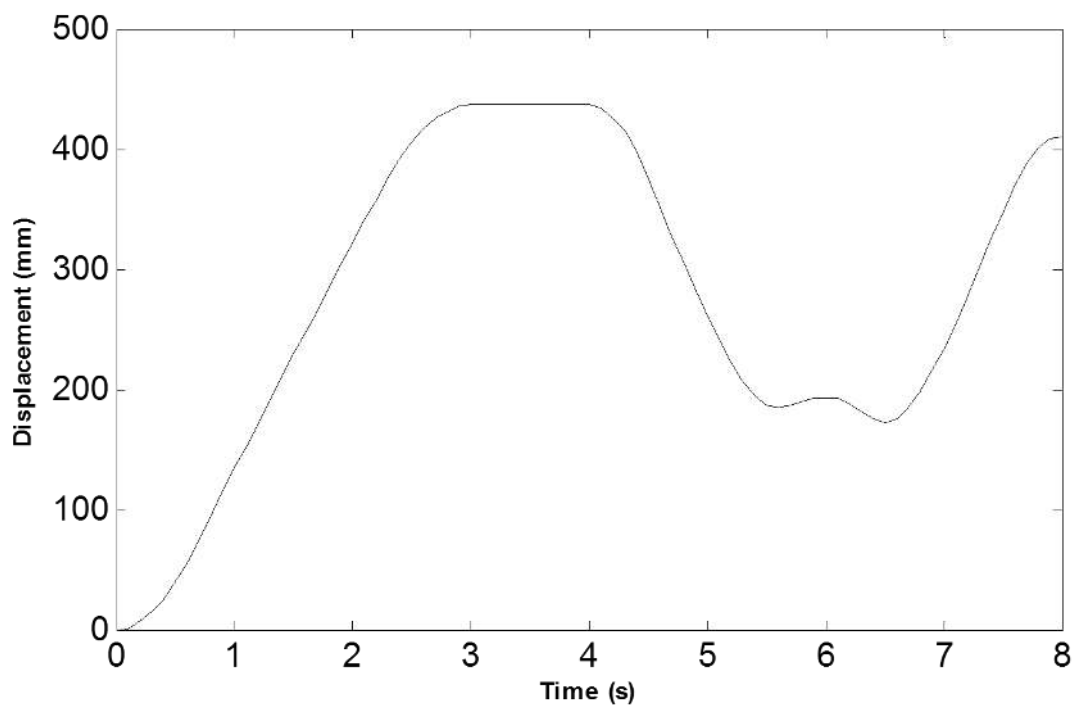


Figure 4.23 Magnitude of the displacements for rigid + flexible motion graphed in ABAQUS

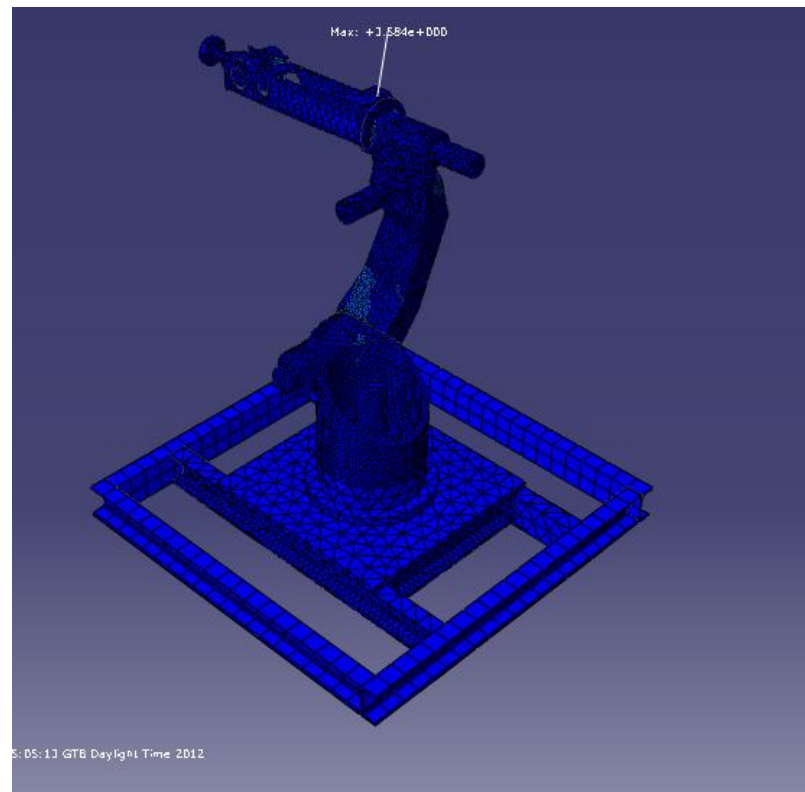


Figure 4.24 max von mises stress for case-3

Max stress during the simulation for case 3 is at the point as shown in Fig. 4.24 which is 3.584 Mpa. Max. displacement is 131 μ .

Table 4.6 Rigidity results

	Case-3
Smax (maximum von mises stress)	3.584MPa
Umax (maximum displacement)	131 μ
fmin (lowest natural frequencies)	13.55 Hz

CHAPTER FIVE

EXPERIMENTAL RESULTS FOR MODAL ANALYSIS

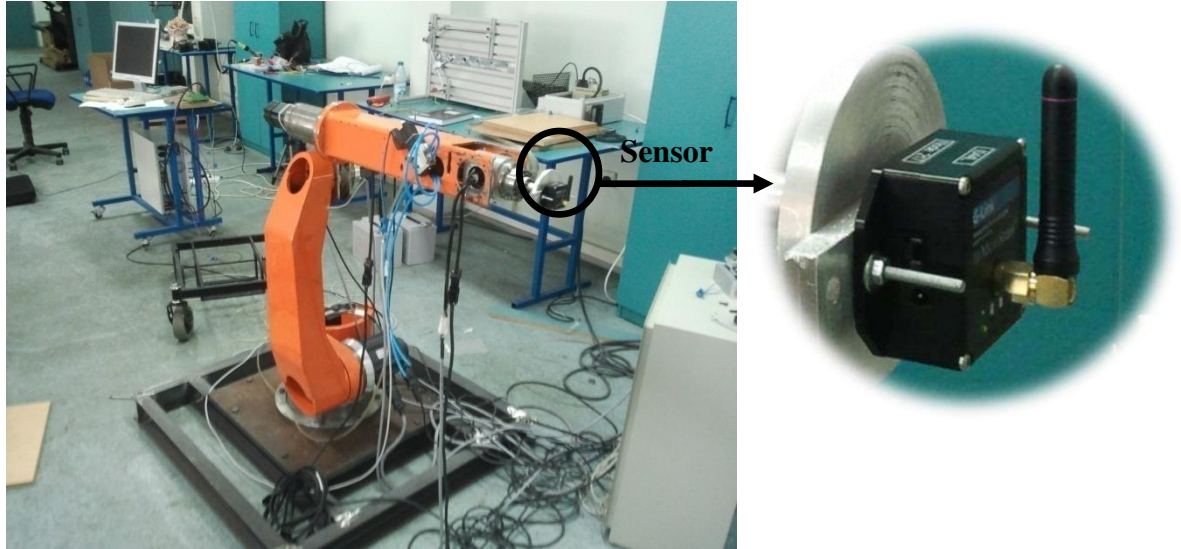


Figure 5.1 Measurement system of the robot

The measurement system for obtaining experimental results are shown in Fig. 5.2

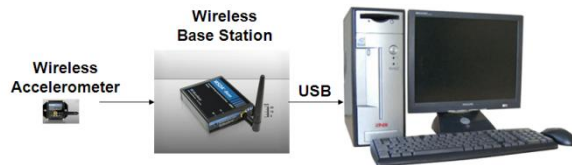


Figure 5.2 Measurement system

The wireless vibration sensor is a MicroStrain G-Link triaxial accelerometer (Accelerometer range: $\pm 10g$, 47 g, 58 mm x 43 mm x 26 mm). The wireless base station is MicroStrain Gateway WSDA-Base-104. MicroStrain wireless systems are capable coordinating and supporting tens to hundreds highly synchronized nodes in a system. Nodes sense events and wirelessly relay sensor signals. Two types of record option are available in the Node Commander Software: Streaming and Trigger. Streaming supports real time streaming from node on the software screen. When

streaming are started, the amplitudes of acceleration signal will be graphed on the screen in real time until the streaming is stopped. Trigger option records data in the node with respect to specified sampling rate. The recorded data can be downloaded to the computer via base station and plotted by various software like MatLAB.

The data received by the base station is transmitted to the computer through Universal Serial Bus (USB) and recorded by using Node Commander Software. The sampling rate is selected 2048 Hz. The frame is excited by a hammer impact. The FFTs (Fast Fourier Transforms) of the recorded signals in the x, y, and z directions are taken in MatLAB (Fig. 5.3). The frequencies where the peaks appear are determined for the natural frequencies.

Due to not taking into account the flexibility of the motors by SolidWorks, modal analysis of the robot is performed in ABAQUS. Experimental results are compared with simulation results and the comparison is given in Table 5.1

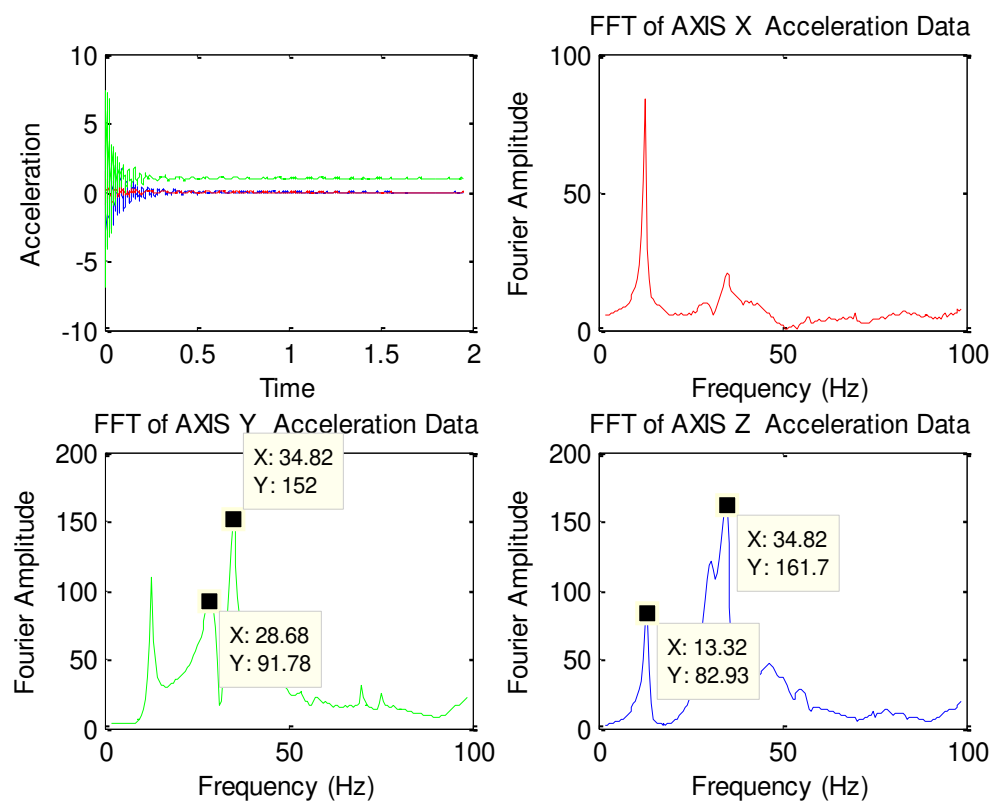


Figure 5.3 Amplitude of acceleration and FFT signals

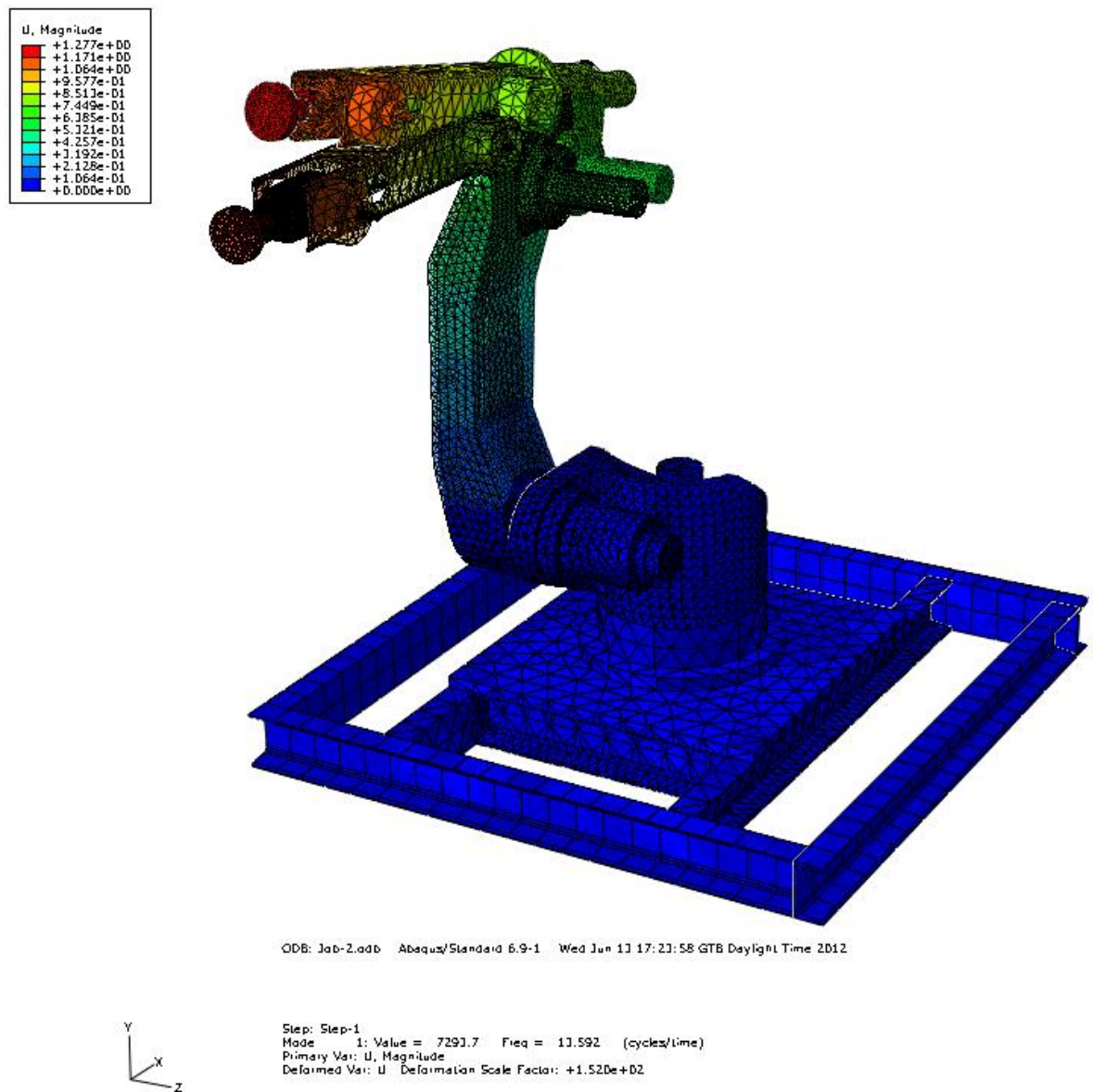


Figure 5.4 First mode of natural frequency 13.59 Hz

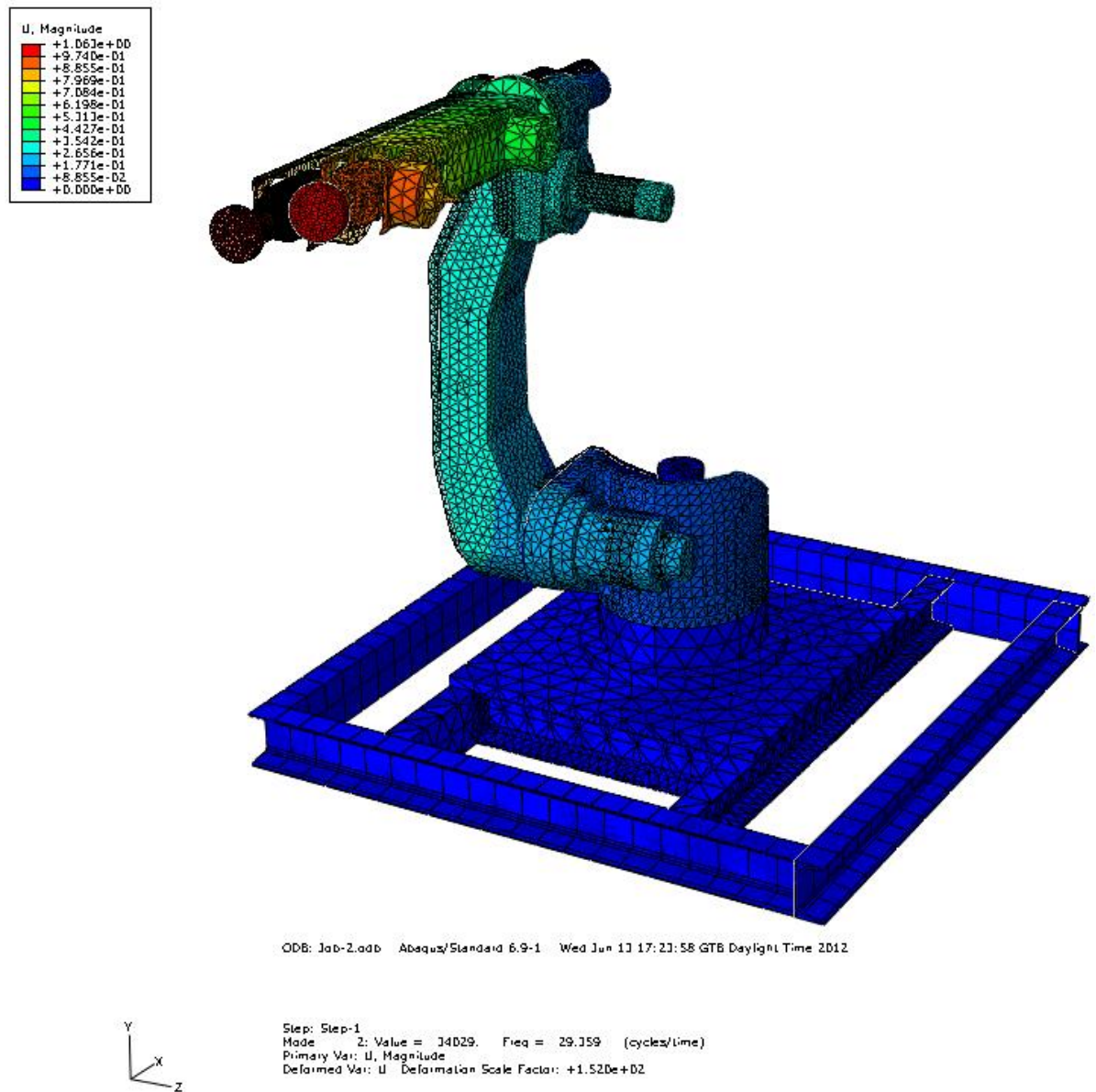


Figure 5.5 Second mode of natural frequency 29.36 Hz

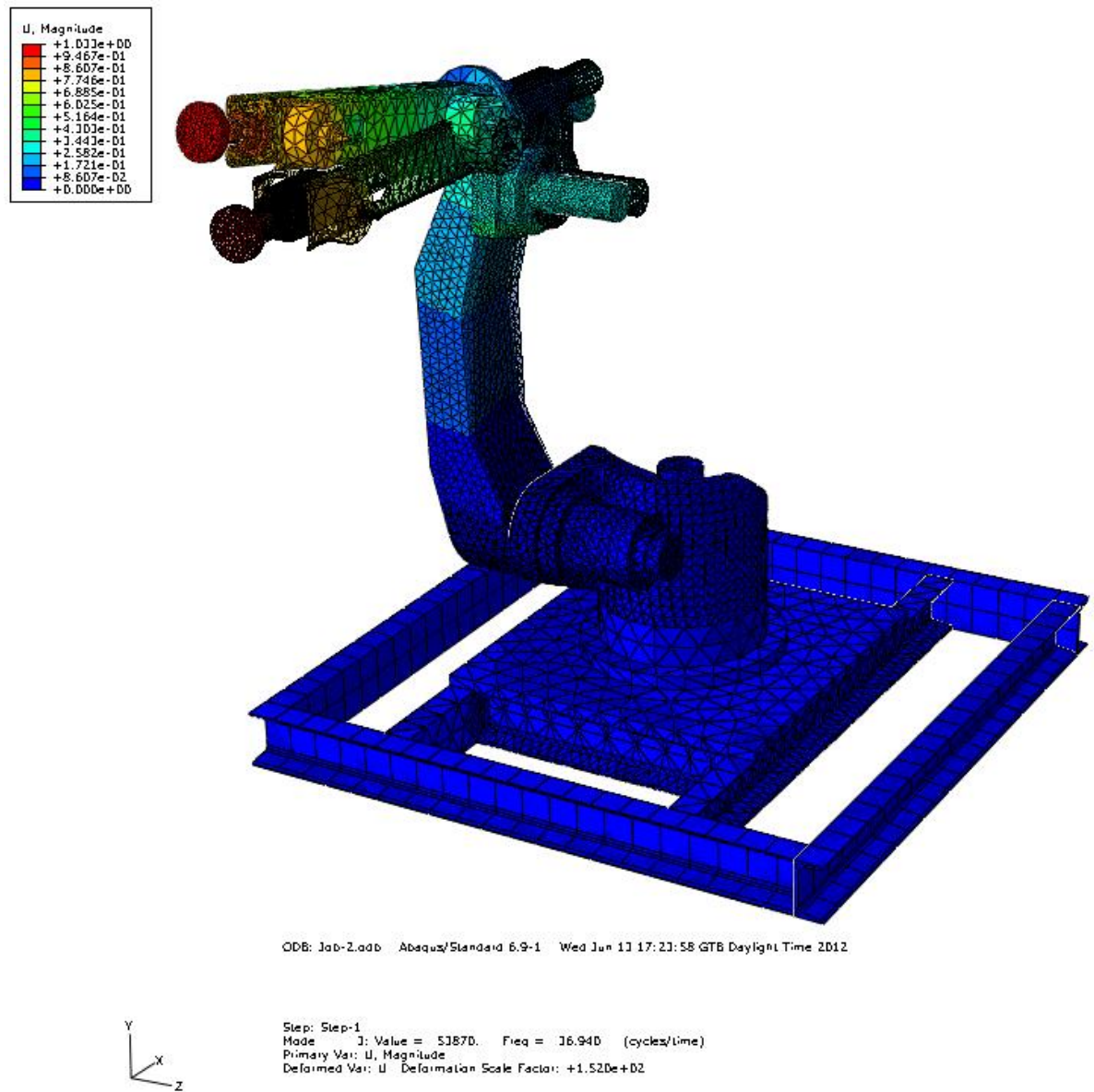


Figure 5.6 Third mode of natural frequency 36.94 Hz

Table 5.1 Simulation and experimental results of modal analysis

	Experiment	Simulation
First natural frequency	13.32	13.59
Second natural frequency	28.68	29.36
Third natural frequency	34.82	36.94

CHAPTER SIX

TRAJECTORY PLANNING WITH MOTION CONTROL CARD

ADLINK 8366

In this chapter, motion control and trajectory planning are discussed on the actual robot by using the same planned work of case-1. ADLINK PCI 8366 motion control card is used for sending the created path from SolidWorks to actual robot.

PCI-8366 is a PCI bus interface card designed for personal computer or industrial computer accompanied with a Mitsubishi MR-J2S-B type or SSCNET type servo amplifier. PCI-8366 can control up to 6 servo amplifiers. The connection between the motion control board and the amplifier is done via high-speed serial communication of the SSCNet II protocol. SSCNet II connections offer the following advantages over pulse train type connections:

- Wiring is simplified because servo amplifiers are connected by multi-drop method and the communication distance is up to 30 meters.
- Parameter management and the construction of absolute positioning system (ABS) are greatly simplified.
- Since commands are transmitted in serial data format, noise-reduction is better, thus reliability is improved. Also the control resolution is increased.
- Users can retrieve abundant information from the servo system through SSCNet II. No longer are you restricted to commands and feedback. You can now also monitor servo status, alarm status and tuning servo parameters.

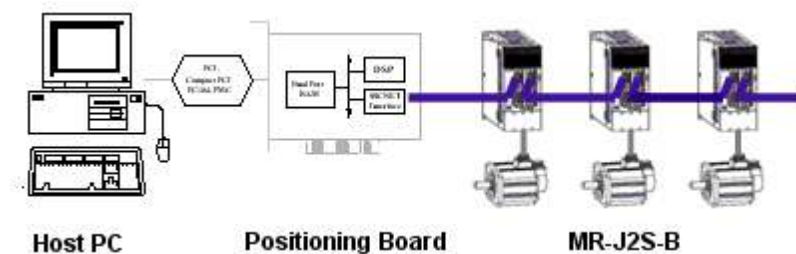


Figure 6.1 SSCNet II High-Speed Connections

The path data of case-1 is used for processing time based velocity data of all motors from SolidWorks. Differently, we added 1 second waiting time between the motion steps as shown in Fig. 6.2

```
nsim,0.1
0,0,0

inverse
0,0,0
0,3,0,0,400,0,0,0
4,6,0,200,-400,0,0,0
7,9,0,-200,-400,0,0,0
-1
```

Figure 6.2 Database for evaluating trajectory planning

After saving input data to “motion1.txt”, Inverse Kinematics and Forward Kinetics buttons are clicked, respectively. As mentioned in Chapter 4, it is seen that the path of case-1 is optimum solution in respect of other cases. When kinematic and kinetic analysis finished, SolidWorks calculate time based velocity profile data for all motors and we can get those data as .csv file by using IS. IS gets time based velocity data of all motors using the following code:

Call elem.ExportCSVFile (fl)

Time based velocity profile graphics are seen as degree/sec in GUI, but the velocity data are saving as rad/sec while exporting data from SolidWorks to csv file by IS. Therefore, all data must be converted to degree/sec by adding a button on the Integrated Software’s interface which is named as “Convert Inp_forw”. (Fig. 6.3)

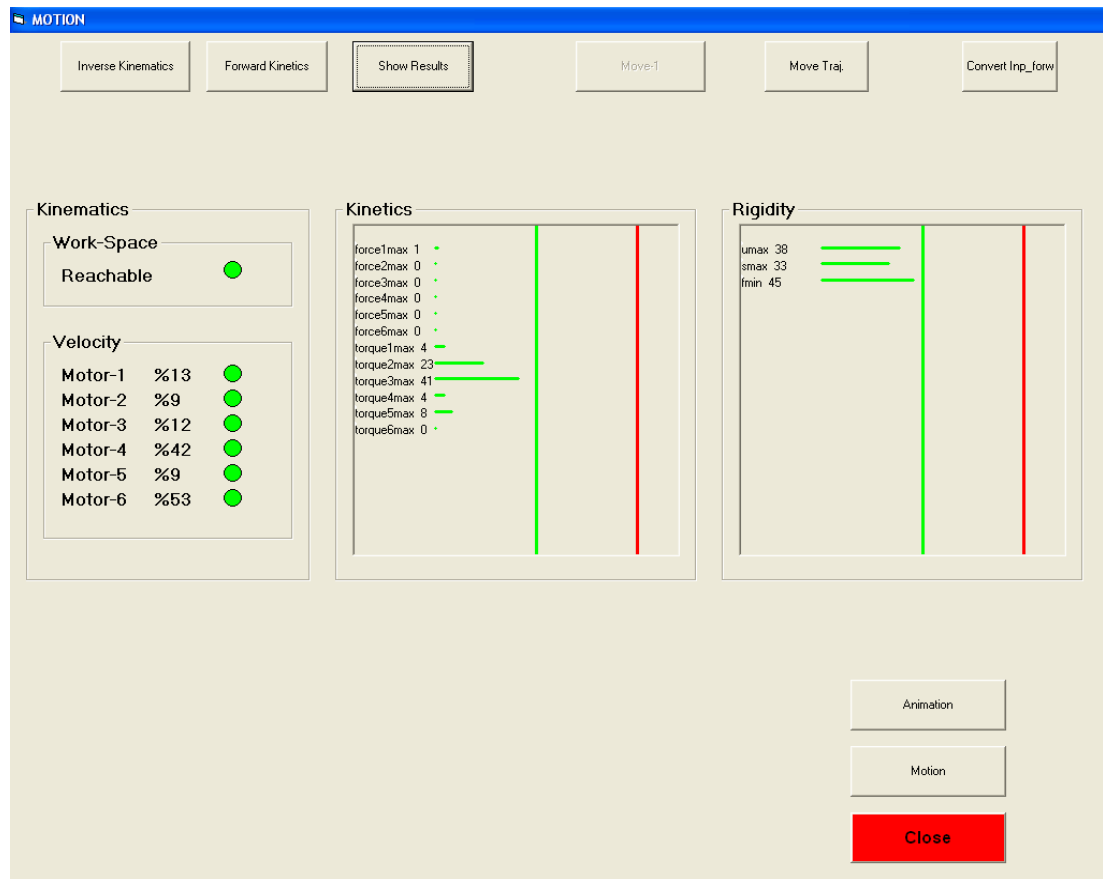


Figure 6.3 The interface of analysis program. “Convert Inp_forw” button converts the velocity values from rad/sec to degree/sec.

Next step is changing interface in IS by clicking “Motion” button to communicate the software with ADLINK motion control card (Fig. 6.3). After clicking “Motion” button, all servo motor drivers are on and ready to drive. On the “Motion” interface, there are many buttons to move the actual robot such as manual motion commands for each motor, demo motion and trajectory moving for following the planned path by getting the data from SolidWorks. For manual motion, we can move specified axis with a specified speed and acceleration/deceleration time. Manual motion uses “start_tr_move” function.

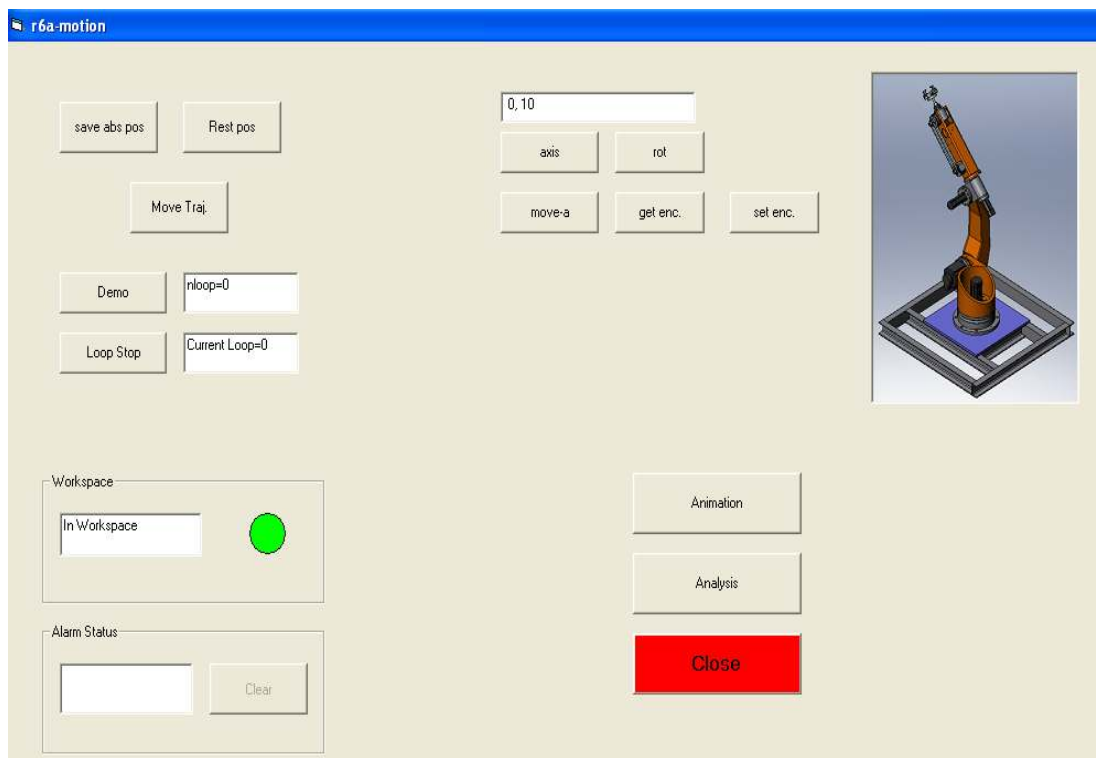


Figure 6.4 “Motion” Interface on the IS.

In this section, it is described how to send velocity commands to motion control card by IS. For trajectory motion, “single axis velocity motion” and “change velocity on the fly” functions are used. Single axis velocity motion function uses “tv_move” command to accelerate the axis from a starting velocity to a specified maximum velocity (Fig 6.5).

tv_move (Axis, StrVel, MaxVel, Tacc)

Here;

StrVel: Starting velocity

MaxVel: Maximum velocity

Tacc: Acceleration time

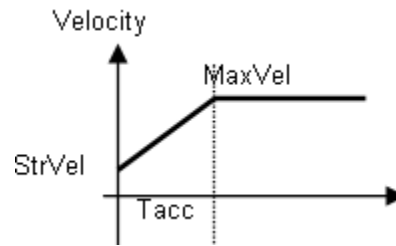


Figure 6.5 tv_move command

Change velocity on the fly function uses the following commands as shown below.

tv_change (Axis, SpeedFactor, Tacc)

tv_stop (Axis, Tdec)

emg_stop (Axis)

“tv_stop” command stops the specified axis with a deceleration time period (Tdec) and a trapezoidal velocity profile during deceleration. “emg_stop” command stops the specified axis instantly.(Fig. 6.6).

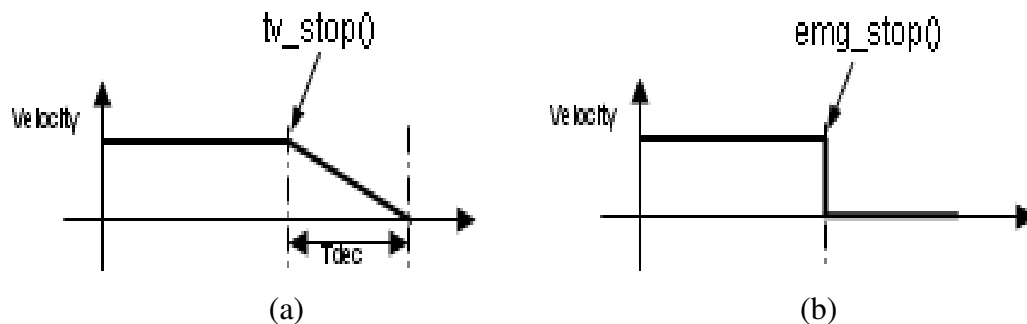


Figure 6.6 (a) tv_stop and (b) emg_stop commands

“tv_change” command changes the moving speed of a specified axis with acceleration time period (Tacc) and a trapezoidal velocity profile during acceleration. The second parameter “Speed- Factor” is used to define the new speed. For example, if the specified axis starts its motion using tv_move command and the “MaxVel” is set to be 10 mm/sec. Then tv_change is applied with “Speed- Factor” = 1.5. The new speed is $1.5 * 10 = 15$ mm/sec (Fig.6.7).

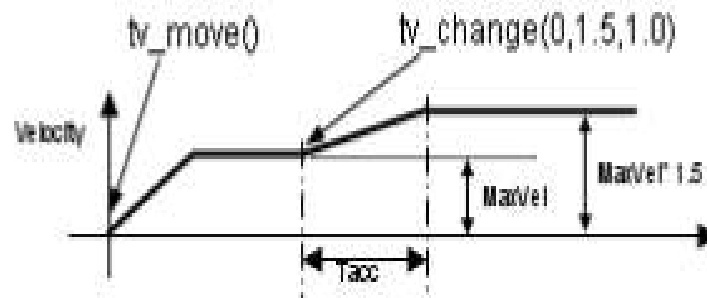


Figure 6.7 tv_change command

“Move Traj.” button on the IS applies the developed algorithm with the following order:

1- IS reads the velocity data of all axis from the text files and finds the maximum velocity value of each axis and checks whether the maximum velocity value given by SolidWorks is smaller than the maximum velocity of the actual motor. If the maximum velocity value in simulation is higher than the maximum velocity of the actual motor, the message box appears on the screen and gives a warning “Max. Velocity!” Program stops running and doesn’t continue applying the algorithm. (Fig. 6.8)

```

For naxis = 1 To 6
  velmax(naxis) = 0
  For n = 1 To nf
    If velmax(naxis) < Abs(vel(naxis, n)) Then velmax(naxis) = Abs(vel(naxis, n))
  Next n
Next naxis

For naxis = 1 To 6
  If velmax(naxis) > wmax1(naxis) Then MsgBox ("Max Velocity!"): Exit Sub
Next naxis

```

Figure 6.8 IS finds maximum velocity and checks whether it exceeds the limit

2- Speed-Factor needs the ratio of new velocity to starting velocity. For example, motor starts the motion with the speed of 100 mm/sec and the new velocity is wanted to be 200 mm/sec. Then Speed-Factor must be $200/100=2$.

```
tv_move(0,0,100,0.1)
```

```
tv_change(0,2,0.1)
```

It is given in motion input data that the time intervals in simulation are 0.1 second. So that, total simulation time is 9 seconds and 91 velocity data for each axis are exported from SolidWorks. IS calculates all the speed factor values of each axis to use it in tv_change command in the following steps.

3- All motor velocity data are taken from motor at the first step (t=0 s) and written to text file “axis(naxis)_feedback.txt”

```
For naxis = 0 To 5
a = get_velocity(naxisa(naxis), vel_enc, vel_com)
Print #naxis + 1, t(naxis, 1), Round((vel_enc / kpuls(naxis)), 5),
Round((vel_com / kpuls(naxis)), 5), vel(naxis, 1)
Next naxis
```

Figure 6.9 Reading velocities from motor

4- Timer is started to calculate total motion time.

5- tv_move command is used to start the motion of all axis. (t=0.1 s)

```
For naxis = 1 To 6
```

```
a = tv_move(naxisa(naxis), 0, Int(vel(naxis, 2) * kpuls(naxis)), t(naxis, 2))
```

```
Next naxis
```

6- IS changes the velocities of each axis during the motion until the velocity sign changes from negative to positive or from positive to negative. tv_change function doesn't support changing the velocity to other sign. Thus, IS controls the sign of velocities in every steps and the new command is used if the sign of the velocity changes in the next step.

```
tv_stop(naxisa(naxis), delt / 2)
```

```
emg_stop(naxisa(naxis))
```

```
tv_move(naxisa(naxis), 0, Int(Sgn(vel(naxis, ch)) * (Abs(vel(naxis, 2)) *
kpuls(naxis))), delt / 2)
```

tv_stop command stops the motion with the time interval of $t/2$ (0.05s), emg_stop prepares the motor to the new motion and tv_move moves the motor again with the time interval of $t/2$ (0.05s). So, the sign of the motor is changed in the time period of 0.1s.

Changing the sign and the value of the velocities and writing the feedback velocities from the motors for all axis continue until the final time step ($t=9$ s). At this time, all axes are stopped by IS.

7- Timer is stopped and written on the screen how long it takes.

For this motion, IS dedicated the motion time as 9.053 seconds. This means that there is 53 ms delay time compared to simulation time. SSCNet has the property of the deterministic time, which is 0.888 ms. Theoretically, the motion command will be passed down to DSP with hand-shaking way. It takes two or three cycle times to complete the delivery and execute the motion command. Consequently, it is sure to waste some time and has slower response.

VisualBasic uses various commands for waiting time such as timegettime(), timer object, time control and gettickcount(). Timegettime() command is more precise than the others but the precision of this command seems to vary by machine (15.625ms to 30 ms). To accurate this command only to 1 ms, we should call the functions of “timebeginperiod” and “timeendperiod” from “winm.dll” library and write the algorithm between these two functions. As a result, SSCNet sends the commands with the period of 0.888ms and VB counts the time with the period of 1ms, there is 0.112 ms delay time. There are 6 axis and 90 frames for each axis, so $90 \times 6 = 540$ data to send on the ADLINK card to the driver. It takes $0.112 \times 540 = 60.48$ ms average delay time, and IS gave us 53ms delay time in the total time.

Finally, all time based velocity feedback data for each axis are written to the text file and compared the velocity data given by SolidWorks. Both velocity profile data are graphed by MATLAB (Fig 6.10).

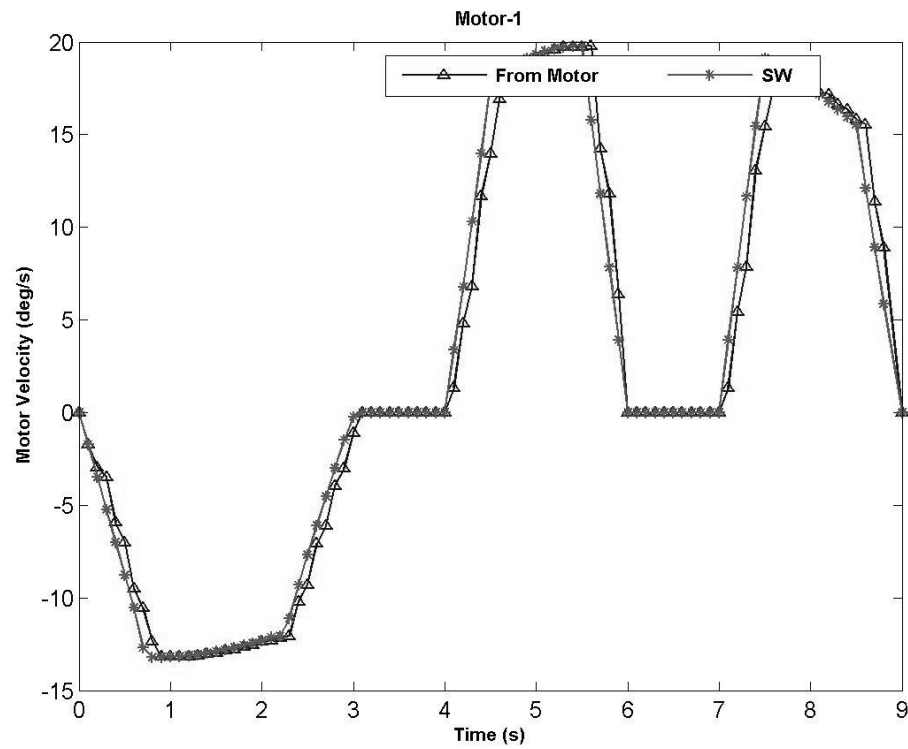


Figure 6.10 (a) Velocity profile for actual motors and motors in simulation (Motor-1)

The ratio of the maximum deviation between reference data (taken by SolidWorks) and receiving data (taken by actual motor) is %0.0134 (for motor-1).

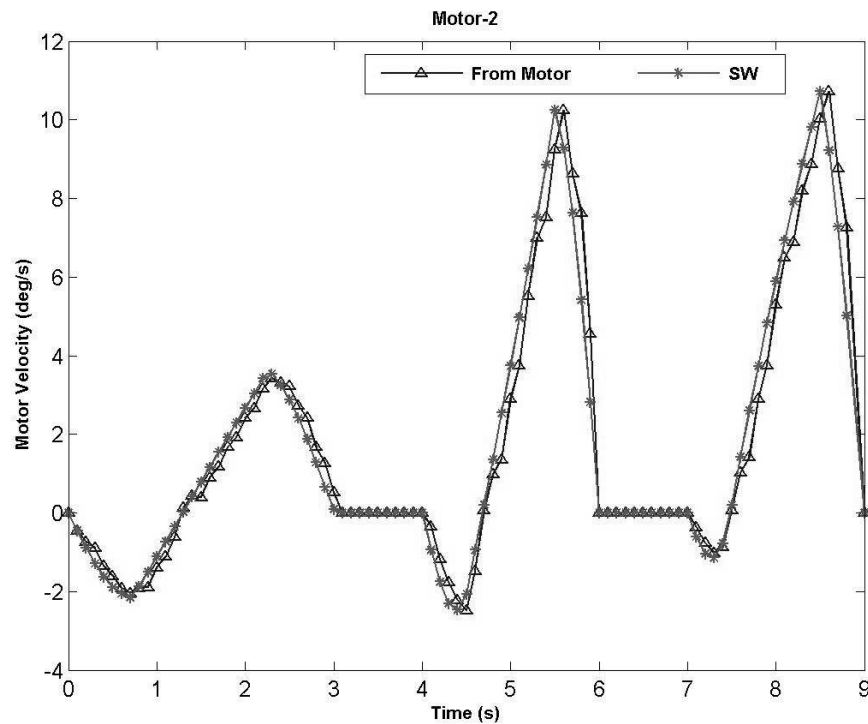


Figure 6.10 (b) Velocity profile for actual motors and motors in simulation (Motor-2)

The ratio of the maximum deviation between reference data (taken by SolidWorks) and receiving data (taken by actual motor) is %0.1285 (for motor-2).

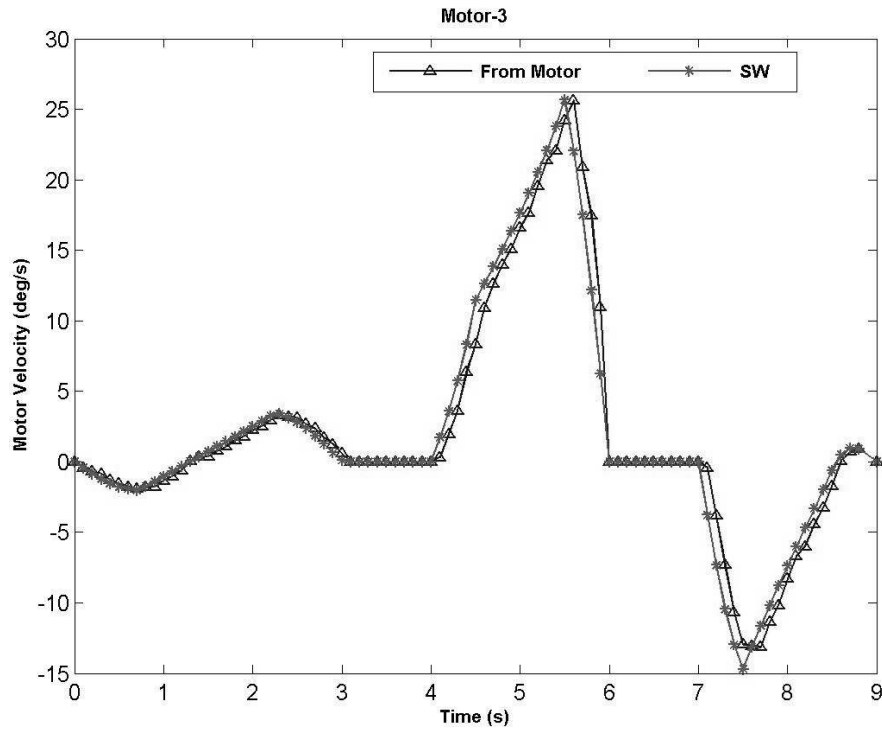


Figure 6.10 (c) Velocity profile for actual motors and motors in simulation (Motor-3)

The ratio of the maximum deviation between reference data (taken by SolidWorks) and receiving data (taken by actual motor) is %0.0639 (for motor-3).

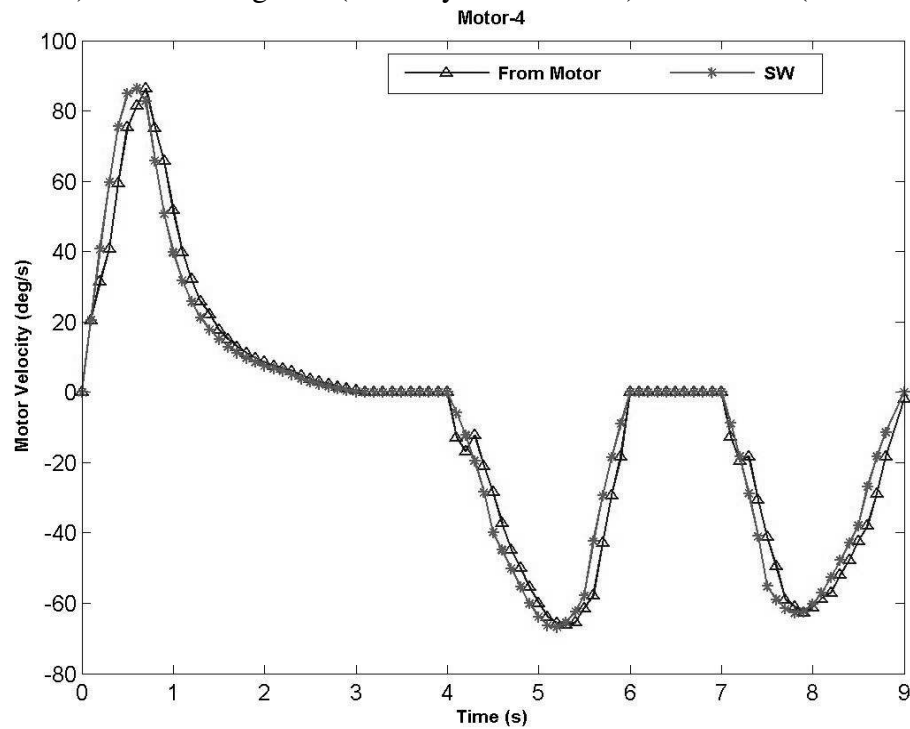


Figure 6.10 (d) Velocity profile for actual motors and motors in simulation (Motor-4)

The ratio of the maximum deviation between reference data (taken by SolidWorks) and receiving data (taken by actual motor) is %0.0005 (for motor-4).

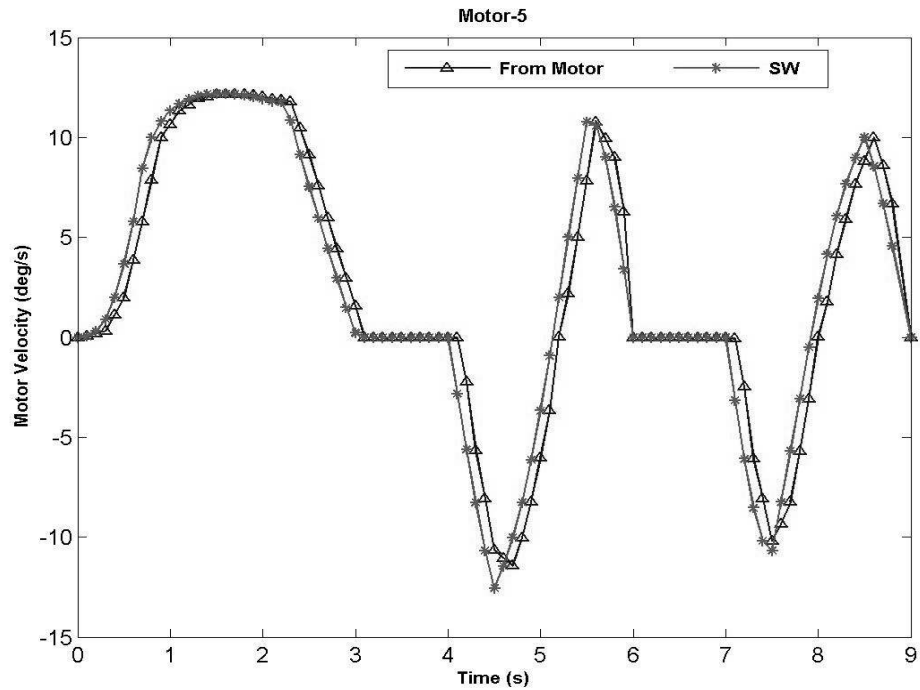


Figure 6.10 (e) Velocity profile for actual motors and motors in simulation (Motor-5)

The ratio of the maximum deviation between reference data (taken by SolidWorks) and receiving data (taken by actual motor) is %0.115 (for motor-5).

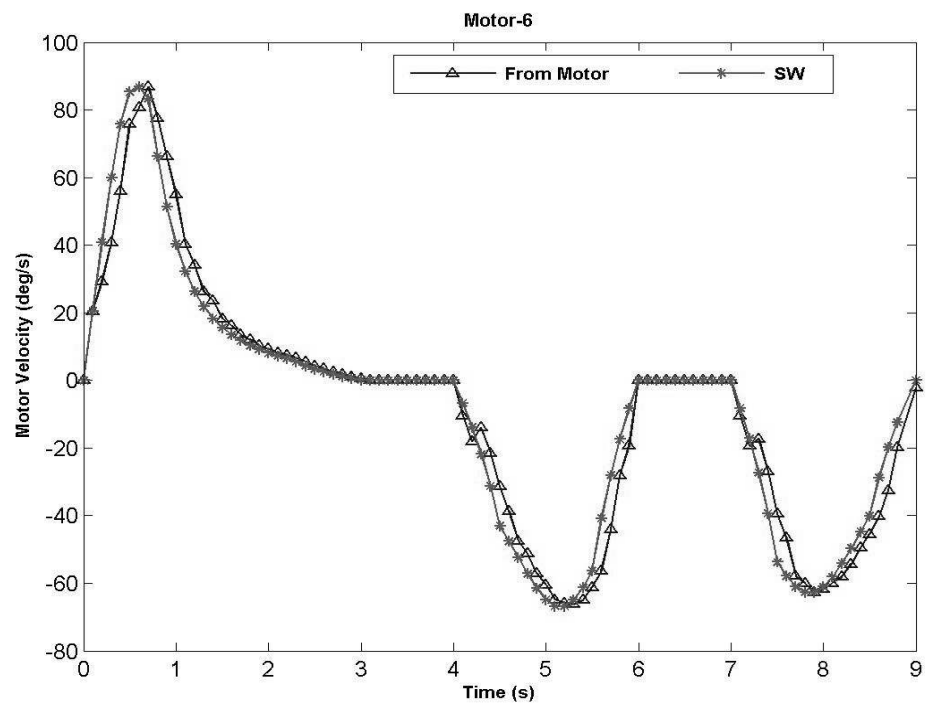


Figure 6.10 (f) Velocity profile for actual motors and motors in simulation (Motor-6)

The ratio of the maximum deviation between reference data (taken by SolidWorks) and receiving data (taken by actual motor) is %0.0279 (for motor-6).

Table 6.1 The ratio of the maximum deviation between reference data (taken by SolidWorks) and receiving data (taken by actual motor)

	The ratio of maximum deviation (%)
Motor-1	0.0134
Motor-2	0.1285
Motor-3	0.0639
Motor-4	0.0005
Motor-5	0.115
Motor-6	0.0279

CHAPTER SEVEN

CONCLUSION

Six axis serial robots are widely used in industry such as automotive, aerospace, ship building etc. for welding, painting, mounting, drilling and pick and place. Robots have higher capacity to perform all kinds of jobs, but the most important is to program these robots. Evaluation of robot strength in every task and planning trajectory to perform a job is a very complex procedure that plays an important role in optimizing of application of robots. Over 30 robot programming manufacturers are available and there are also 30 different programming languages are available. These programs are complicated to understand their language and restricted in only kinematic analysis, therefore integrated software is required which includes kinematic and kinetic analysis and robot programming to move the robot with defining path. The research objective of this thesis is to design, manufacture and develop a six axis serial robot in BATUL laboratory, to evaluate the kinematic and kinetic workspace of this robot and to define an optimal trajectory for given tasks in view of life span of robot and less energy consumption.

In this study, a six axis serial robot is designed and produced in BATUL laboratory. A parametric model is generated for the base part of the robot by using VisualBasic to determine the sizes of the robot. Parametric model program can be applied to all robot parts. Rigidity of six axis serial robot is evaluated by dynamic analysis. SolidWorks + CosmosMotion + Abaqus program packages are used for the dynamic and rigidity analysis. An integrated program is developed in VisualBasic to analyze dynamics and rigidity parameters of the robot, to define an optimal trajectory and to move the actual robot and to define an optimal trajectory. The programming application interface (API) capabilities of the package are used. The program uses the package for the rigid body dynamic analyses necessary to test kinematic and kinetic limits and F.E analyses for rigidity work space. The integrated software processes defining motor velocity profile data by SolidWorks and sends commands to the actual robot to move the desired path.

The following parameters are considered in the analyses:

1. Robot workspace
2. Motor angular velocity limits
3. Reaction forces in bearings
4. Motor torques
5. Maximum displacement during the path
6. Von Mises stress
7. Lowest natural frequency

An obstacle is assumed which is placed between the starting point and ending point of the desired path. 3 different cases for the paths are considered to define the optimal trajectory, which are passed over the obstacle, under the obstacle and behind the obstacle. The starting point and ending point for 3 paths are in the same locations. The simulation results show that the motor torques in case 2 are the highest between cases. The motor velocities in case 3 are the highest. Robot is more rigid in case 1 and the motor torques and motor velocities are in acceptable ranges. The path for case 1 can be selected for the optimal trajectory. Due to having lowest motor velocities, lowest motor torques and higher rigidity of the path for case 1, robot is more durable and consumes less energy to perform the same job.

The path for case 1 is selected and the motor velocities profile data for all six motors are taken from SolidWorks. SolidWorks gives the velocity data in rad/sec if VisualBasic is used for this work and the velocity data in pulse/sec are needed to send to the actual motors, therefore an algorithm is developed in VisualBasic to convert the received data from SolidWorks to sending data to the actual motors. Integration software included one more module to communicate with the ADLINK motion control card (PCI 8366) to move the robot. An algorithm is developed in VisualBasic for the desired trajectory. This algorithm should send velocity data to all six motor simultaneously, because robot must follow the same path given by SolidWorks. “tv_move” and “tv_change” commands are used to move the robot in velocity mode. The time based feedback velocity data are taken from six motors during the motion and compared with the velocity profile data which are taken by

SolidWorks. Both data are plotted in the same graphic for each motor. The maximum deviation for each motor is determined which is between the sending velocity profile data and the receiving velocity profile data. It is seemed that the deviations for six motors are in acceptable ranges.

Robotic systems are needed to achieve certain tasks. Engineers are forced to use computer programs that are difficult to understand. Hence, user-friendly integrated software are needed to achieve integrated analysis after defining tasks, to define an optimum trajectory for a desired job and to generate motion algorithm to control the actual system. The next goal may be to operate the integrated software more user-friendly. Engineers should only define tasks and choose the type of mechanical structure. Then, integrated computer programs should make dynamic analysis, choose an ideal robot type and sizes, offer an optimum trajectory, show simulation results and control the actual system.

REFERENCES

- Articulate robot*, (2011). Retrieved 20 May 2012, from <http://www.robotmatrix.org>
- Akdağ, M., Karagülle, H., Malgaca, L. (2011). An integrated approach for simulation of mechatronic systems applied to a hexapod robot. *Mathematics and Computers in Simulation*, 82(5), 818-835.
- Angeles, J., Ma O. (1989). An algorithm for the inverse dynamics of n -axis general manipulators using Kane's equations, *Computers & Mathematics with Applications*, 17(12), 1545–1561.
- Angeles, J., Ma, O. (1988). Dynamic simulation of n -axis serial robotic manipulators using a natural orthogonal complement, *The International Journal of Robotics Research*, 7(5), 32-47.
- Arakelian, V., Jean-Paul, Le B., Mottu, P. (2011). Torque minimisation of the 2-DOF serial manipulators based on minimum energy consideration and optimum mass redistribution, *Mechatronics*, 21, 310-314.
- Borrow, J.E. (1988). Optimal Robot path Planning Using the Minimum-Time Criterion. *IEEE Journal of Robotics and Automation*, 4, 443-450.
- Hasan, A. T., Hamouda, A.M.S., Ismail, N., Al-Assadi, H.M.A.A. (2006). An adaptive-learning algorithm to solve the inverse kinematics problem of a 6 D.O.F serial robot manipulator, *Advances in Engineering Software*, 37, 432–438.
- Industrial robot*, (2012). Retrieved 20 May 2012, from <http://en.wikipedia.org>

- Lotz M., & Bruhm H., & Czinki, A., & Zalewski, M. (2011). A real-time motion Control strategy for redundant robots improving dynamics and accuracy, *Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2011 3rd International Congress on*, 1-6
- Nearchou, A. C. (1998). Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm, *Mech. Mach. Theory*, 33(3), 273-292.
- Nguyen, C.C., Pooran, F.J. (1989). Dynamic analysis of a 6 dof ckcm robot end effector for dual-arm telerobot systems, *Robotics and Autonomous Systems*, 5(4), 377–394.
- Nguyen, C.C., & Pooran, F.J. (2003). Kinematic analysis and workspace determination of a 6 dof ckcm robot end-effector, *Journal of Mechanical Working Technology*, 20, 283–294.
- SolidWorks Premium Package-2007 (SolidWorks+ CosmosMotion + CosmosWorks)*, (December 05, 2008). Retrived 18 February 2012 from www.solidworks.com
- Song, Y.D., & Anderson, J.N. (2003). Adaptive path tracking control of robotic manipulators with unknown payload Dynamics, *Systems & Control Letters*, 17(1), 59–70.
- The Language of technical computing (MATLAB)*, (n.d). Retrieved 10 April 2012, from www.mathworks.com/products/matlab/
- Witney, D. (1969). Resolved motion rate control of manipulators and human prostheses. *Man and cybernetics*, 10, 47-53.
- Yun, W. M., & Xi, Y. G.(1991). Optimum motion planning in joint space for robots using genetic algorithm. *IEEE on Robotics and Automation System*, 18, 373-393.