11-14-2003

On the Development of an Automated Design Procedure to Design Optimal Robots

Edward William Mebarak
Florida International University, emebarak@hotmail.com

DOI: 10.25148/etd.FI08081533

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

ON THE DEVELOPMENT OF AN AUTOMATED DESIGN PROCEDURE TO DESIGN OPTIMAL ROBOTS

A thesis submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

in

MECHANICAL ENGINEERING

by

Edward Mebarak

2003

To: Dean Vish Prasad
College of Engineering

This thesis, written by Edward Mebarak, and entitled On the Development of an Automated Design Procedure to Design Optimal Robots, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this thesis and recommend that it be approved.

Ibrahim Tansel

Diana Rincon

Sabri Tosunoglu, Major Professor

Date of Defense: November 14, 2003

This thesis of Edward Mebarak is approved.

Dean Vish Prasad
College of Engineering

Dean Douglas Wartzok
University Graduate School

Florida International University, 2003

ACKNOWLEDGMENTS

For this work, groups of people have contributed directly, some others indirectly. This work has become possible due to opportunities that have come across in this undeniably wonderful country. These opportunities would not have been possible without the advice and concern of my major professor Dr. Sabri Tosunoglu, who advised me from the beginning to end when pursuing this degree. I would also like to thank my sister who brought me up here, my big family; most of whom are in my hometown and my girlfriend who has supported me and encouraged me to excel.

In some way I want to dedicate this thesis to Colombians. We are used to suffer in different senses and still give a warm smile to others and work hard to have thriving cities of peace. I cannot forget my essence, the place I come from, my nature or who I am.

I cannot forget my workmates, they were there for me, never said I'm busy or come back later, and every body who had contributed me to learn and gain knowledge.

Without any of these factors this work simply would not have been possible.

ABSTRACT OF THE THESIS

ON THE DEVELOPMENT OF AN AUTOMATED DESIGN PROCEDURE TO DESIGN OPTIMAL ROBOTS

by

Edward Mebarak

Florida international University, 2003

Miami, Florida

Professor Sabri Tosunoglu, Major Professor

The objective in this work is to build a rapid and automated numerical design method that makes optimal design of robots possible. In this work, two classes of optimal robot design problems were specifically addressed: (1) When the objective is to optimize a pre-designed robot, and (2) when the goal is to design an optimal robot from scratch.

In the first case, to reach the optimum design some of the critical dimensions or specific measures to optimize (design parameters) are varied within an established range. Then the stress is calculated as a function of the design parameter(s), the design parameter(s) that optimizes a pre-determined performance index provides the optimum design.

In the second case, this work focuses on the development of an automated procedure for the optimal design of robotic systems. For this purpose, Pro/Engineer© and MatLab© software packages are integrated to draw the robot parts, optimize them, and then re-draw the optimal system parts.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF FIGURES

FIGURE PAGE

1. INTRODUCTION

A robot is defined as “an automatic device that performs functions normally ascribed to humans or a machine in the form of a human” [1]. For many it is also how science fiction has become scientific fact.

1.1 History and Background

The idea of a robot is not new. For hundreds of years man has been imagining intelligent mechanized devices that perform human-like tasks.

In fact, the term robot was first used in 1920 in a play called Rossum’s Universal Robots (R.U.R) by the Czech writer Karel Capek. The plot was simple: man makes robot then robot kills man. Many movies that followed continued to show robots as harmful, menacing machines. In 1941, science fiction writer Isaac Asimov first used the word *robotics* to describe the technology of robots and predicted the rise of a powerful robot industry. His prediction has come true. Recently there has been explosive growth in the development and use of industrial robots to the extent that terms like robot revolution, *robot age*, and *robot era* are used. *Robotic*s is now an accepted word that describes all technologies associated with robots.

Five years later, George Devil and Joseph Engelberger formed the world’s first robot company [1].

Devol predicted that the industrial robot would “help the factory operator in a way that can be compared to business machines as an aid to the office worker.” A few years later, in 1961, the very first industrial robot was “employed” in a General Motors automobile factory in New Jersey. Since 1980, there has been an expansion of industrial robots into non-automotive industries. The main factor responsible for this growth has been the technical improvements in robots due to advancement in microelectronics (“ME”) and computers [2].

Today, fully functioning androids are many years away due to the many problems that must be solved. However, real, working, sophisticated robots are in use today and they are revolutionizing the workplace. These robots do not resemble the romantic android concept of robots. They are industrial manipulators and are really computer controlled arms and hands. Industrial robots are so different to the popular image that it would be easy for the average person not to recognize one.

Why robots needed to exist?

The present invention makes available for the first time a more or less general-purpose machine that has universal application to a vast diversity of applications where cyclic control is desired.

1.2 Benefits of Robots

Robots offer specific benefits to workers, industries and countries. If introduced correctly, industrial robots can improve the quality of life by freeing workers from dirty, boring, and dangerous and heavy labor. It is true that robots can cause unemployment by replacing human workers but robots also create jobs: robotic technicians, salesmen, engineers, programmers and supervisors.

The benefits of robots to industry include improved management control and productivity and consistently high-quality products. Industrial robots can work tirelessly night and day on an assembly line without a loss in performance. Consequently, they can greatly reduce the costs of manufactured goods. Summarizing we can say:

1) The robots proven to be more cost-effective than human labor in many industrial applications.

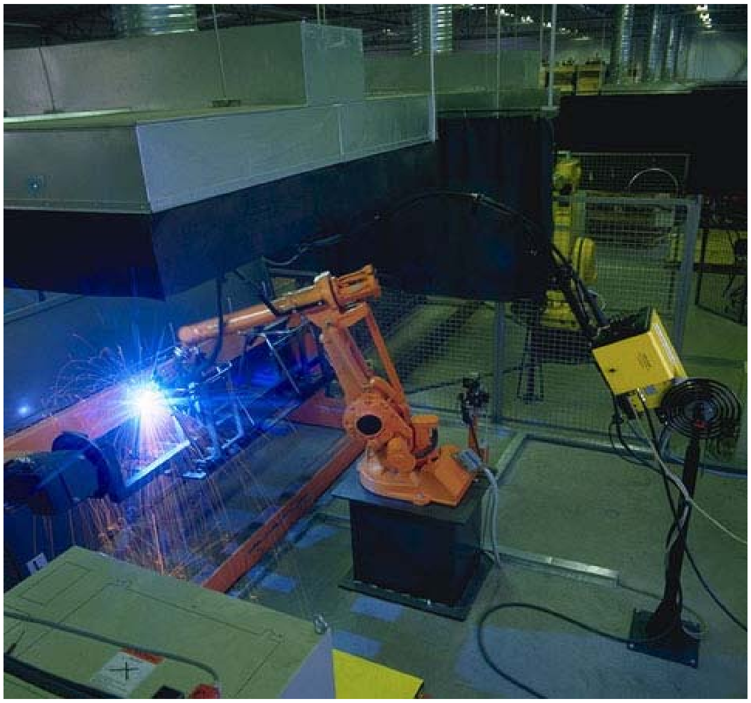


Figure 1 Automated robot

2) They are very fast when performing assembly operations.

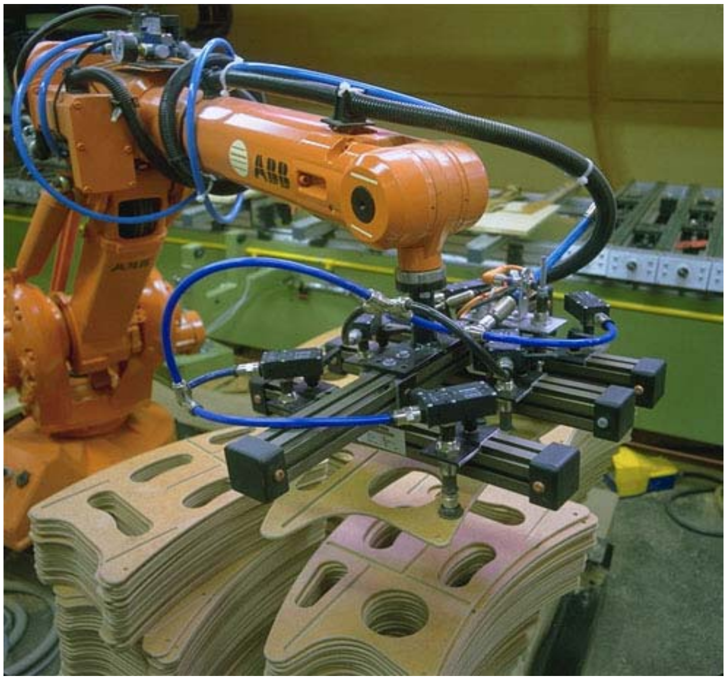


Figure 2 Efficiency and rapidness in cyclic processes

3) Robots can go where dangerous environments represent danger for humans or simply where the environment is not appropriate due to contamination or humans cannot withstand it.

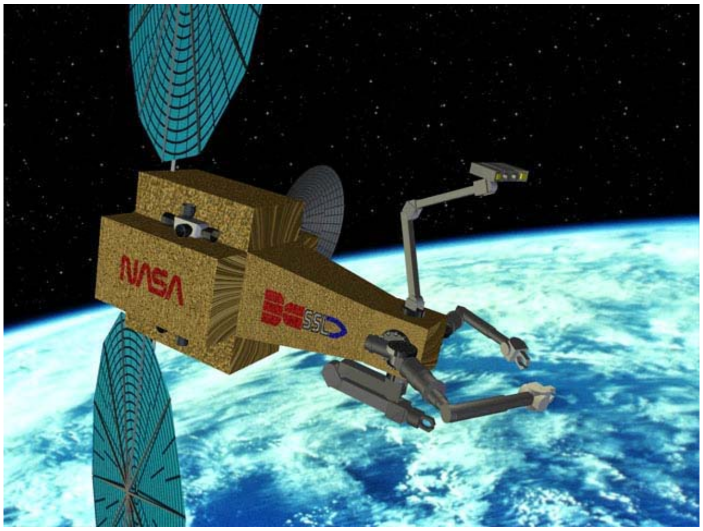


Figure 3 Out of reach applications for humans

1.3 Growing Importance of Robots

As technology develops, robots have become very important due to the automation of many of the industrial and manufacturing processes. Also, the need to explore or work in hostile regions or areas that are subject to hostile environmental conditions such as the lack of oxygen, or presence of radiation requires the use of robots. For these reasons many researchers around the world are spending time to analyze, develop and improve machines to perform these tasks [3]. The final goal of this thesis is to develop a method for the optimal design of a robot. One of the main points of designing a robust and versatile robot is to develop a solid geometry; as the two factors affecting robot geometry are the link shape and weight [4].

The need for more flexible and adaptable structures, improvement of the overall performance index as well as the reduction in cost of manufacturing, reduced gear and motor sizes, adaptability to small environments.

1.4 Robotic Components

In general, major components of a robot might be categorized in three areas.

(A) Electronic parts

(B) Mechanical system

(C) Main structure or frame

Another components classification for a robotic manipulator is set in Table 1.

Table 1. Components and subcomponents of a robot

Component	Description	Example
Central Processor	The central processing unit is the brains of the operations. It is responsible for interpreting the commands given to the robot and is responsible for controlling all the other functions	Human Brain Computer chip with memory and program Computer system
Transportation	This system is responsible for the movement of the robotic system. Some robots are fixed like those on the floor of manufacturing facilities; others have wheels or legs and feet to move them around.	Wheels Feet Pogo Stick Snowboard Wings Propellers Rockets
Sensors	These are special systems that allow the robot to gather information about the world around it. Touch, heat, light, speed, sound sensing systems. Can all be part of a robotic system. A special version of this kind of sensor is a vision system	Touch sensors in the Skin Nose Taste buds Ears
Remote link or communications system	This system is responsible for the communications to an operator or controller usually at some distance from the robot.	Telephone Fax Internet
End effectors	End effector's systems are systems that allow the robot to do things other than move. These may include hand-like objects or clamps, drills or latches. The end effectors may be tools themselves, such as the Canada arm end effector's or it may be a vehicle for other tools such as the Special Purpose Dexterous Manipulator of the International Space Station Canada arm.	Arms Legs Poles Hands Screw Drivers Drills Hammers Brooms
Vision Systems	This system is responsible for the gathering of visual signals similar to the human eyes. This system can "see" in other parts of the light spectrum such as radar or infrared.	Eyes TV camera

This thesis focuses on the structure or the frame of the robot; the structure is the one that must withstand the external conditions and payloads.

1.5 Optimization of Robots

The aim of an inventor is commonly to come up with the specification of a innovative method or tool that satisfies a need. For the *mechanical* inventor the ambition is a machine that coordinates movements and applies forces to accomplish a task. While computer-aided design tools are available to generate, analyze and even manufacture new parts for existing machines, software tools that attempt to identify a new device matching to a user-specified task are limited to very specific examples. This thesis describes an integrated theory for a computer aided design environment to support the invention of articulated machines or robots. The focus on spatial linkages seeks to bring the broad base of knowledge on the analysis and synthesis of geometric constraints in kinematics synthesis together with the modern insights and capabilities of robotics. While the one-degree-of-freedom planar devices and six-degree-of-freedom spatial devices have been the focus of research for the past decades, the broad range of devices between these extremes have received little attention. The challenges of creating in one's mind and analyzing spatial linkages pose a fundamental obstacle to the inventive use of these devices [19].

The need to satisfy an optimal design of a robot is the aim of the research; optimal design is defined in terms of weight and stress. In other words the question posed is as follows: Is the robot capable of withstanding the external and internal loads and have the lightest structure possible?

In reality two different case scenarios where it is feasible to conduct an optimization of a robot are given: First, when we have a pre-designed robot and the second and most important one when it is wanted a robot designed from scratch.

1.5.1 Optimization of Pre-Designed Robots

In order to describe this approach an existing robot is reproduced and modeled in a regular drawing package. (Pro/Engineer is used to generate all parts assembly and drawing). Design parameters that define the robot's geometry are selected. The objective is to choose the optimal values that satisfy the objective function, as it was mentioned earlier focusing on weight and stress. Energy and efficiency are directly related to the weight of the robot. Therefore, the goal was set to finally optimize the robot's structure by means of an FEA (Finite Elements Analysis) package.

A previous study carried out at Massachusetts Institute of Technology (MIT) uses Pro/Mechanica to minimize the stress within a flexible body at a given configuration. Pro/Mechanica optimization module employs the FEM for computing, approximate numerical solutions to the deflection equations that predict the response of physical systems subjected to external influences. Unfortunately, the study shows no viability to a particular model selection and it is evaluated on one design parameter [5].

1.5.2 Design of a Robot from Scratch

Optimal design of robot manipulators is important in the design stage as it influences the system performance such as the cost of manufacturing, accuracy, related deflection, and so on. In general, the overall performance index could be improved at this stage. These characteristics, besides directly influencing the robot's functionality, also relate to the overall cost, gears and motors size, power consumption, and reliability. All these variables influence the weight of a robot to a certain degree. It is known that the weight is defined by the geometry and parameters such as link lengths, thickness, etc. in addition to the material selected.

To avoid going back and forth in the design stage, to handle with someone else's design that implies sometimes to understand concepts functionality and background. An approach is given to entirely design a robot from scratch, where the unknown variable is the physical configuration (geometry of the robot). The thesis' particular goal is to make a contribution and make an effort to bring these concepts together (Design and Automation) and develop an automated procedure for the optimal design of robot manipulators with the aim to design the links and final assembly of the robot with a few clicks of the mouse.

1.6 Proper Design of Robots

Robot design is hampered by the lack of established, well-known design rules, and designers cannot easily grasp the space of possible designs and the impact of all design variables on a robot's performance. Realistically, a human can only design and evaluate several candidate configurations, though there may be thousands of competitive designs that should be investigated. In contrast, an automated approach to configuration synthesis can create tens of thousands of designs and measure the performance of each one without relying on previous experience or design rules. This thesis creates an extensible, automated system for robot configuration synthesis. This research focuses on the development of synthesis capabilities required for many robot design problems: A flexible and effective synthesis algorithm, useful simulation capabilities, appropriate representation of robots and their properties, and the ability to accommodate application-specific synthesis needs. It can synthesize and optimize kinematics, structural geometry, and task and control parameters for a wide range of robots.

A similar and extended work titled evolutionary design for robots was carried out by Chris Leger in which a full integration is developed between dynamics, kinematics and structure of an optimal design of a robot [6].

When designing a robot for a task with many new characteristics, relevant experience in the design team may be limited and may restrict the range of designs that are explored. Frequently, a person or team investigates a small number of concepts based on previous design experiences and selects a few that look promising. This initial brainstorming often consists of qualitative thought experiments and back-of-the-envelope calculations to predict how well each design meets the major requirements of the task: Can each robot perform the basic motions required? Will the robot's kinematics necessitate large actuation forces or be prone to collisions and link interference? Based on the answers to this question one of the candidate configurations is selected for simulation and further design. Only after detailed simulation of the robot some problems become apparent.

Significant effort has now been invested in the design, such as deriving inverse kinematics for the robot, devising an appropriate controller, and searching through catalogs for motors and gearboxes. Much of this effort is robot-specific and is lost if a different design is chosen.

2. OPTIMIZATION CONCEPT

What is Optimization?

Optimization is the task of finding a best compromise of all admissible conditions to achieve an objective, formulated in mathematical terms, in any type of engineering problem to make a decision (The ultimate goal of this method is either to minimize the effort or to maximize the benefits).

In this thesis, we are to focus on minimization problem, to be precise on the minimization of stress or weight (or a combination of them) of a robotic structure.

Different methods of optimization have been developed to deal with diverse types of problems.

2.1 Optimization Problems

According to the magnitude of the problem, a classification can be given with respect to the nature of the solution [7]:

- ✓ Multi-objective variable: It is defined as a problem where more than one set of solutions is satisfied (relative absolute).
- ✓ Single-objective variable: It is defined as a problem where one exclusive set of solution is found (total absolute).

Another classification may be given relative to the dimensionality of the optimization problem:

Single-variable minimization: This is a recognized type of problem where the function to be minimized depends only on a single variable, this means that the function to be optimized possesses only a single variable that totally fulfills and satisfies the conditions to be achieved.

The solution for this type of problem can be found by using several well-known methods; the most common approach is the one that takes the derivative with respect to the unknown design variable and then equates to zero and then solve for the variable.

Multi-variable minimization: In this type of problems, more than one set of variables are involved in the function.

Regardless of the dimensionality of the optimization problem, relative and/or absolute optimum can be encountered.

2.2 Optimization Techniques

To widen the knowledge, a study of the basics of each optimization definitions constrained and unconstrained approaches are given below.

Another characteristic that can be used to classify an optimization problem is its nature: The unconstrained and constrained nature of the optimization problem, which is also explained below.

2.2.1 Unconstrained Optimization Problems

It is the process where it is needed to find along a function a set of points that satisfies certain conditions, exactly to find all critical points and classify each as a local maximum, a local minimum or a saddle point. Example of the common behavior of this type of functions is shown below. Figure 4 represents the function *$f(x,y)=x^2y-3x$,* where x and y $\in$ c (complex numbers).

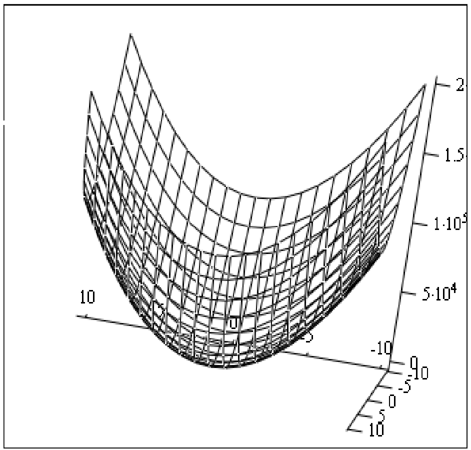


Figure 4 Unconstrained function in space

A problem of this type is not usually found in engineering related subjects because an engineering problem is based on its physical behavior that may cause the design parameters to be bound or the solution may converge to negative or unacceptable values where it may possibly not make sense.

Most commonly it is desirable to have a set of positive numbers (for physical dimensions) that makes sense and give a physical meaning of the design parameters. Among the most used methods are:

Nonlinear Systems of Equations, Nonlinear Least Squares, Global Optimization and Non-differentiable functions [10].

2.2.2 Constrained Optimization Problems

In this type of optimization problems, the performance index is minimized (or maximized) subject to constrained equations. In this type of optimization the solution is basically restricted by how the functions behaves in the space. In the next graph shown below it is plotted the bending stress function when a load is applied in the extreme of a cantilever beam, as defined by (Eq 2.1), the constraint in this case is the yield stress of the material.

It is noticed that more than one unique solution is possible to obtain (Fig 5). This example is detailed in the next lines, according to the conditions of the Figure 30. (Where x1 is represented as the link length of the beam, x2 is the base length and x3 is the height) Let's fix F (force) and x1 (link length) in order obtain a function that will depend on only two variables with the aim to plot it in 3-D space. Here the function's behavior is visually presented in Figure 5 to illustrate this sample optimization problem.

$$\sigma = \frac{Mc}{I} \quad \text{and} \quad \begin{matrix} M = F * x1 \\ I = {}^{1}\!/_{12}(x2 * x3^{3}) \end{matrix} \longrightarrow \sigma = \frac{6 * F * x1}{x2 * x3^{2}} \quad \text{Subject to:} \quad \begin{matrix} x1 = k, \\ x2 \geq 0, \\ x3 \geq x2 \end{matrix} \qquad (2.1)$$

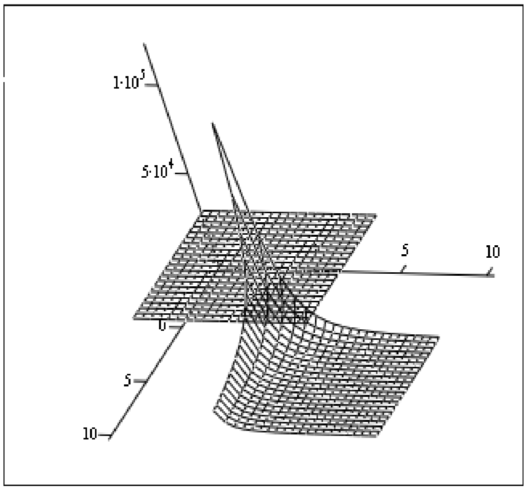


Figure 5 Constrained optimization

This type of a problem definition is more likely to be found in engineering problems because normally an objective function is given and some parameters are restricted or limited by other relations or are simply constrained to have positive values, material properties and have limited capacities. These are to be considered in order to provide optimal solutions with a complete and meaningful sense.

Among the most used of this category of methods are [8,10]:

- ✓ Linear Programming
- ✓ Semi-Definite Programming
- ✓ Nonlinearly Constrained
- ✓ Bound Constrained
- ✓ Quadratic Programming
- ✓ Network Programming
- ✓ Stochastic Programming Methods.

Now we will proceed with the evaluation of the nature of the problem and then continue to revise the field of interest and examine the constrained-multivariable optimization more in-depth.

The best approach to a minimization problem is the one that copes with the problem characteristics and takes into account the nature of the problem, including the type of constraints. Experience is highly desirable when solving this type of problems.

2.3 Optimization of a Constrained Problem

Due to the nature of the problems dealt with in engineering, it is possible to assume that the objective function and the constraint functions are differentiable along the considered domain. The main idea for solving the problem is to derive a set of necessary conditions for the optimality of the problem [8]. We may use these conditions to determine an optimal solution.

Some of the most important and effective optimization methods are briefly reviewed below. These methods presented below are listed as [10]:

- ✓ Weight sum
- ✓ Compromise approach
- ✓ Paretto Approach
- ✓ Newton Method
- ✓ Sequential Quadratic Programming (SQP)
- ✓ Lagrangian Approach

2.3.1 Weight sum

Zadeh first proposed the basic idea of assigning weights to each objective function and combining them into a single-objective function. The weighted-sum method can be represented as follows [9]:

Max or min z(x)= $\sum_{k=1}^{q} w_k f_k(x)$ where x $\in S$ (2.2)

Where W_k, the weight can be interpreted as relative emphasis or worth of that objective compared to all other objectives. In other words, the weight can be interpreted as representing our preference over objectives. Therefore, an optimal solution to this problem relates to a particular preference structure. Moreover, the optimal solution to the problem is a non-dominated solution provided that all the weights are positive.

For two given points x^1 and x^2 in the decision space S, $z(x^1)> z(x^2)$ if and only if $x^1>x^2$, and , $z(x^1) = z(x^2)$ if and only if $x^1>x^2$, then we have :

$\{>\}=\{(x^1, x^2)\in S \text{ x } S/ z(x^1) > z(x^2)\}$

$\{\sim\}=\{(x^1, x^2)\in S \text{ x } S/ z(x^1) = z(x^2)\}$

{?}=0

Because of numerical ordering in the weighted-sum function, there is no ambiguity in preference comparison. For any two points, either is better, worse or equivalent to the other. Only one of the three cases must ensure. There is no such thing as an indefinite set in preference structure.

2.3.2 Compromise Approach

The compromise approach can be regarded as the kind of mathematical formulation of the goal-seeking behavior in terms of distance function. Because it is simple to understand and easy to compute the concept has a much general appeal. The compromise approach identifies solutions close to the ideal point [9,10].

2.3.3 Paretto Approach

The Paretto approach assumes that no information on the preference among objectives is available and that all we know is that for each component z_j, the greater value is preferred. The Paretto preference is defined as follows: For any two points z^1 and z^2 in the criterion space Z, point z^1 is preferred to point z^2 if and only $z^1 \geq z^2$ and that there is at least one component, say the r^{th} component for which $z_{r^1} > z_{r^2}$ and that for all others $z_{k^1} > z_{k^2}$ k=1,2..q, $k \neq r$, then we have [10].

$\{>\}=\{(z^1, z^2) \in Z \text{ x } Z/\ z^1 > z^2)\}$

$\{\sim\}=\{(z,z) \in Z \text{ x } Z/\ z \in Z)\}$

$\{?\}=\{(z^1, z^2) \in Z \text{ x } S/ \text{ neither } z^1 > z \text{ nor } z^1 < z^2)\}$

Compared with other methods, once the proper regret function, or value function, or weight coefficients are determined, the set {?} is empty and the problem becomes a dimensional comparison of a mathematical programming problem. With the Paretto approach we tackle problems with the set {?} not empty.

2.3.4 Newton's Method

Newton's name is associated with that large number of the root finding methods that employ derivatives or approximations to them. One of the most widely used of these is usually known as the Newton-Raphson methods. One variant that we will look at briefly is the modified quasi-Newton's method [9]. In all cases it is necessary to evaluate both the function and its derivatives (or else to approximate the derivative) at various values of x. In general the solution is stated as follows:

$$f(x+dx) = f(x) + f'(x)\,dx \qquad (2.3)$$

2.3.5 Lagrangian Approach

This method provides a general solution to the optimization problem [11]. The objective function f(x) should be minimized subject to m constraint equations: $g_i\ (x) \geq 0$ and/or $h_j\ (x)=0$, where $g_i\ (x)$, $i = 1, 2...\ m$ and $h_j\ (x)$, $j = 1, 2...\ n$ are differentiable inequality constraints.

By defining the slack variables $\theta_i^2 = -g_i(x)$, $i = 1, 2...\ m$, the problem is restated as follows: Minimize f(x) subject to $g_i\,(x) + \theta_i^2 = 0$, $i = 1, 2...\ m$. This approach converts the constrained optimization problem into an unconstrained problem by defining the Lagrangian function [11]: $L(x,\theta,\lambda) = f(x) + \sum_{i=1}^{n} \lambda_i (g_i(x) + \theta_i^2)$ (2.4)

Then, to find a solution, we must take partial derivatives of *L (x,θ,λ)* with respect to every variable involved: [*x*1, *x*2…*x*n],[λ1, λ2.‥λn],[θ1, θ2.‥θn], equate each of them to zero, and then solve the system of resulting set of equations (3n+2m) for x, λ and θ. For instance, let us set up an example and consider minimizing the geometric dimensions of a cantilever beam (with a square cross section) subject to a payload *f* (see Figure 30) at its extreme end. Let's assume that x1=beam length, x2=base, x3=height.

$$\sigma = \frac{M\cdot c}{I} \quad \text{and} \quad \begin{matrix} M = f\cdot x1 \\ I = \frac{1}{12}(x2\cdot x3^{3}) \end{matrix} \longrightarrow \sigma = \frac{6\cdot f\cdot x1}{x2\cdot x3^{2}} \quad \text{Subject to:} \quad \begin{matrix} x1 = k, \\ x2 \geq 0, \\ x3 \geq x2 \end{matrix} \tag{2.5}$$

Where; σ=bending stress on the beam, f=external payload applied on the beam and I=beam inertia, k=constant value (fixed constraint).

Forming the Lagrangian function:

First of all we need to transform inequality constraints to equality ones when all constraints are expressed in to equalities we are ready to transform the constraints into Lagrangian parameters.

Solving for the unknown x1-k=0, then adding the Lagrangian term, $\lambda 1\cdot(x1-k)=0$

Then to transform the inequality constraint to equality constraint it is necessary to add a factor of restitution, Then we obtain $-x2+t1^{2}=0$, which contributes to the Lagrangian function as a term $\lambda 2\cdot(-x2+t1^{2})=0$. Then, repeat the same process over the other constraints and form the Lagrangian function as follows, just needed to add them up, to acquire the big expression:

$$\sigma(x1,x2,x3,\lambda1,\lambda2,\lambda3,t1,t2)=\left(6\cdot f\cdot\frac{x1}{x2\cdot x3^{2}}\right)+\lambda1\cdot(x1-k)+\lambda2\cdot\left(-x2+t1^{2}\right)+\lambda3\cdot\left(-x3+x2+t2^{2}\right) \tag{2.6}$$

where λ=[λ1 λ2 λ3]^T is the vector of Lagrange variables and t[t1 t2]^T and x=[x1 x2 x3]^T the slack variable vector that converts inequality constraints into equality equations. The above function can now be minimized as an unconstrained function *L(x, λ,* t *)* by taking partial derivatives with respect to the elements of x, λ and t variable vectors:

$$\begin{array}{ll}
1 & \frac{d}{d\,x1}\sigma(f,x1,x2,x3,t1,t_{2},\lambda1,\lambda2,\lambda3)\rightarrow 6\cdot\frac{f}{\left(x2\cdot x3^{2}\right)}+\lambda1 \\
2 & \frac{d}{d\,x2}\sigma(f,x1,x2,x3,t1,t_{2},\lambda1,\lambda2,\lambda3)\rightarrow -6\cdot f\cdot\frac{x1}{\left(x2^{2}\cdot x3^{2}\right)}-\lambda2+\lambda3 \\
3 & \frac{d}{d\,x3}\sigma(f,x1,x2,x3,t1,t2,\lambda1,\lambda2,\lambda3)\rightarrow -12\cdot f\cdot\frac{x1}{\left(x2\cdot x3^{3}\right)}-\lambda3 \\
4 & \frac{d}{d\,\lambda1}\sigma(f,x1,x2,x3,t1,t2,\lambda1,\lambda2,\lambda3)\rightarrow x1-k \\
5 & \frac{d}{d\,\lambda2}\sigma(f,x1,x2,x3,t1,t2,\lambda1,\lambda2,\lambda3)\rightarrow -x2+t1^{2} \\
6 & \frac{d}{d\,\lambda3}\sigma(f,x1,x2,x3,t1,t2,\lambda1,\lambda2,\lambda3)\rightarrow -x3+x2+t2^{2} \\
7 & \frac{d}{d\,t1}\sigma(f,x1,x2,x3\ t1,t2,\lambda1,\lambda2,\lambda3)\rightarrow 2\cdot\lambda2\cdot t1 \\
8 & \frac{d}{d\,t2}\sigma(f,x1,x2,x3,t1,t2,\lambda1,\lambda2,\lambda3)\rightarrow 2\cdot\lambda3\cdot t2
\end{array} \tag{2.7}$$

Equating each of the derivatives to zero and the solution is set-up; we need to solve a set of nonlinear algebraic equations to obtain the final solution.

2.3.6 Sequential Quadratic Programming (SQP)

This method is developed for those systems that fit the following description [10, 20]:

$$f(x)=C^{T}*x+\frac{1}{2}*x^{T}*D*x \qquad \text{Subject to:}\quad A*x\le B, x\ge 0 \tag{2.8}$$

where f(x) is an objective function, x is an n-dimensional design vector, and the rest of the coefficient matrices and vectors are defined accordingly.

			A= [Matrix of constraints]
x=$\begin{bmatrix} x1 \\ x2 \\ x3 \\ . \\ . \\ xn \end{bmatrix}$	C=$\begin{bmatrix} c1 \\ c2 \\ c3 \\ . \\ . \\ cn \end{bmatrix}$	D=$\begin{bmatrix} D1 \\ D2 \\ D3 \\ . \\ . \\ Dn \end{bmatrix}$	B=[Matrix of vectors]
			C=[linear-terms representation Matrix of the function f]
			D=[non-linear-terms representation Matrix of the function f]

To express the inequality constraints as equality ones, it is needed to transform them as follows:

Introduce the slack variables S_i^2, i=1, 2,…m associated to $A \cdot x \leq B$

Introduce the slack variables T_j^2, j = 1, 2,…n associated with the n-dimensional vector x

Then the problem becomes:

$$f(x)= C^T*x + \frac{1}{2}*x^T*D*x \qquad \text{Subject to} \quad A_i \cdot x+S_i^2=b_i \ ; \ -x_{j+}T_j^2=0 \qquad (2.9)$$

Now we can transform the original the original optimization problem with constraints to a new optimization problem *without* any constraints. This is accomplished via the addition or definition of Lagrange variables:

$$L(x,S,T,\lambda,\theta) = C^T+\frac{1}{2}*x*D*x + \sum_{j=1}^{n} (\theta_j*(-x_j+T_j^2)) + \sum_{i=1}^{m} (\lambda_i*(A_i^T+S_i^2-b_i) \qquad (2.10)$$

Where the function L(x,S,T,λ,θ) is known as the Lagragian function

Now the problem has become unconstrained and ready to be solved by following these steps:

Take partial derivatives of the Lagrangian function with respect to every variable that the function is dependant of:

$$\frac{\partial L}{\partial x1}=0;\ \frac{\partial L}{\partial x2}=0\ldots\frac{\partial L}{\partial xn}=0$$

$$\frac{\partial L}{\partial S1}=0;\ \frac{\partial L}{\partial S2}=0\ldots\frac{\partial L}{\partial Sn}=0$$

$$\frac{\partial L}{\partial T1}=0;\ \frac{\partial L}{\partial T2}=0\ldots\frac{\partial L}{\partial Tn}=0 \tag{2.11}$$

$$\frac{\partial L}{\partial \lambda 1}=0;\ \frac{\partial L}{\partial \lambda 2}=0\ldots\frac{\partial L}{\partial \lambda n}=0$$

$$\frac{\partial L}{\partial \theta 1}=0;\ \frac{\partial L}{\partial \theta 2}=0\ldots\frac{\partial L}{\partial \theta n}=0$$

Equate each derivative to zero and solve the resulting system of non-linear equations.

The total number of resulting equations is 3n+ 2m (where n is the number of initial unknown variables and m the number of the slack variables).

Despite being a very reliable, simple and fast method, the difficulty for the SQP approach consists of expressing the nonlinear terms in the objective function in the form of coefficient D as stated above, and it is almost impossible to have dividing terms in the objective function due to the same reason. This also means that no higher-powered terms are allowed in f(x) .

Now that we have a better understanding of the different types of optimization methods and how can we transform inequality constraints to equality ones and knowing that the physical meaning is crucial in finding a reasonable solution, it is proceed in the next chapter with the design and optimization of a robotic structure.

2.4 Optimization of a Robotic Structure

This solution of this type of problem corresponds to a constrained-multivariable optimization problem that is reviewed in this chapter.

2.4.1 Problem Definition

When a robot of N degrees of freedom and M number of links is subject to an external payload, it develops a stress condition that depends both intrinsic (Internal payloads, see Figure 55) and extrinsic parameters (materials and external payloads, see Figure 12).

In general the objective function that will be minimized has the following form:

$$f = f(\sigma) \qquad \text{where} \qquad \sigma = \sigma(P) + \sigma(W) \tag{2.12}$$

And also *W=W(GV),* where P=external payload, $\sigma =$ stress level, W=weight of the robot and GV = constant robot parameters such as link lengths, thickness', etc.

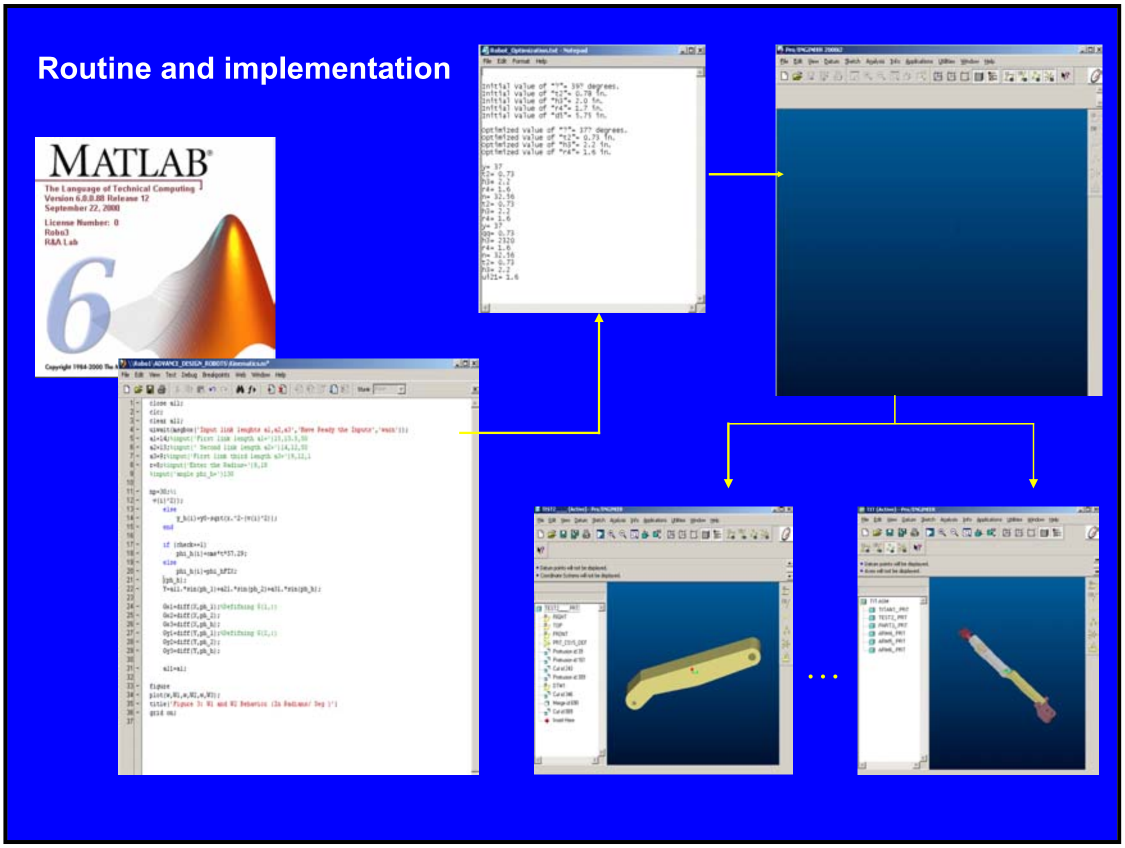
2.4.2 The Optimization Toolbox in Matlab

Matlab is a programming package that has different optimization functions that are used in this work. The function "fmincon" finds the constrained minimum of a scalar function of several variables. The minimization starts at an initial estimate. This method is generally referred to as constrained nonlinear optimization method [12]. The fmincon function bases its approach on the Sequential Quadratic Programming (SQP) for medium scale optimization and on the method of precondition conjugate gradients for large-scale optimization [10,11].

The optimization part is carried out and written in Matlab software where this program handles as inputs link lengths, cross-area definition, kinematics of the robot (path’s validation, joint velocities and accelerations), payloads, materials of manufacturing and stresses.

2.4.3 Matlab’s Role in the Optimization

Figure 6 shows how the optimization in Matlab and design in Pro/E are mapped in order to automate the optimal design process.



Routine and implementation

MATLAB
The Language of Technical Computing
Version 6.0.0.88 Release 12
September 22, 2000
License Number: 0
Robu3
R&A Lab
6
Robot_Optimization.txt - Notepad
File Edit Format Help
. . .

```
Initial value of "?"= 39? degrees.
Initial value of "t2"= 0.78 in.
Initial value of "h3"= 2.0 in.
Initial value of "r4"= 1.7 in.
Initial value of "d1"= 5.75 in.

Optimized value of "?"= 37? degrees.
Optimized value of "t2"= 0.73 in.
Optimized value of "h3"= 2.2 in.
Optimized value of "r4"= 1.6 in.

y= 37
t2= 0.73
h3= 2.2
r4= 1.6
h= 32.56
t2= 0.73
h3= 2.2
r4= 1.6
y= 37
gg= 0.73
h3= 2320
r4= 1.6
h= 32.56
t2= 0.73
h3= 2.2
u121= 1.6
```

```
close all;
clc;
clear all;
uiwait(msgbox('Input link lenghts a1,a2,a3','Have Ready the Inputs','warn'));
a1=14;%input('First link length a1=')13,13.5,50
a2=13;%input(' Second link length a2=')14,12,50
a3=8;%input('First link third length a3=')9,12,1
r=8;%input('Enter the Radius=')8,18
%input('angle phi_b=')130

np=30;%i
 +(i)^2));
    else
        y_b(i)=y0-sqrt(r.^2-(x(i)^2));
    end

    if (check==1)
        phi_b(i)=ma*t*57.29;
    else
        phi_b(i)=phi_bPI2;
    ]ph_b);
    Y=a11.*sin(ph_1)+a21.*sin(ph_2)+a31.*sin(ph_b);

    Gx1=diff(X,ph_1);%Defining G(1,1)
    Gx2=diff(X,ph_2);
    Gx3=diff(X,ph_b);
    Gy1=diff(Y,ph_1);%Defining G(2,1)
    Gy2=diff(Y,ph_2);
    Gy3=diff(Y,ph_b);

    a11=a1;

figure
plot(w,W1,w,W2,w,W3);
title('Figure 3: W1 and W2 Behavior (In Radians/ Deg )')
grid on;
```

. . .

Figure 6 General schematic view of the process

Matlab bases its optimization routines in two approaches: The sequential quadratic programming and the conjugate method based on the calculation of Lagrangian, as an example was given earlier.

2.5 Summary

Since the purpose of this thesis is to optimize the design of robots we must get to know how optimization works. In this chapter we have covered the most important and useful methods in optimization. The two basic types of optimization problems encountered are: Constrained and unconstrained optimization problems. Also, it is realized that the design and optimization of robots fall in the constrained-type problem description; a given example of a cantilever beam subjected to a payload solves the optimal geometry by using the Lagrangian method. By depicting this problem, the reader gets a better understanding of how an optimization problem is defined, delimited and solved.

3. FINITE ELEMENT ANALYSIS AND PRO/MECHANICA

3.1 Definition

Finite Element Analysis (FEA) is a computer-based numerical technique for calculating the deflection, strain, stresses and general and behavior of engineering structures under loading. It is used to calculate deflection, stress, vibration, buckling behavior, temperature distribution, and many other phenomena. It can be used to analyze either small or large-scale deflection under loading or applied displacement. It can analyze elastic deformation, or *permanently bent out of shape* plastic deformation. The computer is required because of the astronomical number of calculations needed to analyze a densely meshed structure. The power and low cost of modern computers has made Finite Element Analysis available to many disciplines and companies.

In FEA, a structure is broken down into many small simple blocks or elements. The behavior of an individual element is described with a relatively simple set of equations to model the whole structure, the equations describing the behaviors of the individual elements are coupled into an extremely large set of equations that describe the behavior of the whole structure. The computer can solve this large set of simultaneous equations. From the solution, the computer extracts the behavior of the individual elements. From this, it can get the stress and deflection of all the parts of the structure [21].

FEA also offers a means to deal with structures that are more complex than can be dealt with analytically using partial differential equations. FEA methods can deal business with complex boundaries better than finite difference equations will, and gives answers to real world structural problems. It has been substantially extended in scope during the roughly 40 years of its use.

3.2 Modeling Structures

Finite Element Analysis permits to evaluate a detailed and complex structure in a computer, during the planning of the structure. (The demonstration in the computer of the adequate strength of the structure and the possibility of improving the design during planning can justify the cost of this analysis work). FEA has also been known to increase the rating of structures that were significantly over designed and built many decades ago.

In the absence of Finite Element Analysis (or other numerical analysis), development of structures must be based on hand calculations only. For complex structures, the simplifying assumptions required to make any calculations possible can lead to a conservative and heavy design.

With Finite Element Analysis, the weight of a design can be minimized while the resistance is maintained. Field-testing will be used to establish loading on structures, which can be used to carry out future design improvements via Finite Element Analysis. This technique that is the one used to minimize the weight of Pre-Designed robots in the present work.

In FEA there are different type of concepts that we must get to know, as the material's properties that in other words defines how the part to be analyzed behaves when a force or thermal stress is applied on it. In engineering we find elastic and plastic properties.

By plastic material properties, we mean that the structure is deformed beyond the yield point of the material, and the structure will not return to its initial shape when the applied loads are removed. The amount of permanent deformation may be slight and inconsequential, or substantial and disastrous. In metal forming, deformation is substantial and intentional (consider the shaping of a fender for an automobile). In some structures, *shakedown* producing residual stress due to local permanent deformation that may in some circumstances reduce fatigue problems in zones that will remain in compressive stress as a consequence. An example is the hydrostatic pressure test on a new, post weld heat-treated, steel pressure vessel (opinions on this may vary). In this test the pressure may be taken to 1.5 times the design pressure. Local yielding means that some zones will usually be in compressive stress during conventional use of the pressure vessel, and may be less prone to fatigue crack development.

By large deflection, it is meant that the shape of the structure has changed enough that the relationship between applied load and deflection is no longer a simple linear relationship. This means that doubling the loading will not double the deflection. The material properties can still be elastic.

In addition to analyzing structures for their stress and deflection, other typical analyses are an evaluation of the natural frequency of vibration, and calculation of buckling loads. Steady state, transient, and random vibrations behavior can also be analyzed.

Another aspect to consider is the loads on structures, they can be represented by using forces on the mass of the structure, or by applying distributed pressure over surfaces of the structure, or by applying forces directly to positions in the structure. Also indicating the axis for the motion, and the rate of rotation can enter centrifugal load. Displacements of the structure can be specified at positions in the structure. This can include boundary conditions that imply symmetric structures where only a portion of the structure is modeled. Other boundary conditions will indicate where the structure is supported against movement, by the outside world. Temperature distribution that causes thermal expansion and stress can be applied directly to nodes or to elements with appropriate commands. Uniform temperatures and reference temperatures can also be applied to full models.

3.3 Computer Optimization and FEA

Classical optimization is done manually with algebra, calculus, and the calculus of variations. Problems with a variety of constraints may be handled symbolically using Lagrangian multipliers. Many modern design problems are too complex to be handled with purely algebraic symbolic methods. Computers are used for numerical assessment of variations in a design.

Computer codes to optimize design have been developed ever since the inception of modern digital computers. Today, codes for optimization can be acquired for free, or purchased as part of mathematical subroutine libraries. Some coding of the problem to be solved is required, as are calls to the optimization subroutines to be employed.

A faster method suitable for many optimization problems is to use the optimization engine bundled into spreadsheet programs, such as Microsoft Excel. (This is an option but it must be intentionally installed.) Then the only significant work required is to transform the problem into spreadsheet form. If the problem is difficult to program excel spreadsheet cells can reflect the results of code written in Visual Basic. The spreadsheet program then does most of the work, and the user interface is relatively easy to construct.

In Finite Element Analysis when coupled with optimization, becomes a more demanding problem, because each variation in the design takes a significant amount of time to evaluate its influence. This can make brute-force iterative optimization techniques excessively time consuming. The designer will usually attempt nonlinear optimization under both equality and inequality constraints, when optimization is used with FEA. One approach is to run a small set of variations on the design, then fit curves to the relationship between degrees of freedom, and the properties of the optimization function and properties to be constrained. Software can be used to search this design space, and suggest good starting points for the next set of design checks.

Optimization, as the term is above, implies changing the setting of independent variables in a continuous manner, to get best possible structural properties. An approach like this will look at variations in an existing configuration, but not invent significantly new configurations for a structure. When trying to improve a structure, or to respond to a defined need to support a set of loads with a newly created structure or geometry, the problem becomes far more involved than the rather narrow definition of optimization that have been used. The designer should keep in mind that creativity in finding a structural configuration should not be sidetracked by a narrow approach to optimizing an existing shape.

3.4 FEA Optimization and Implementation in Pro/Mechanica

Engineering re-design is an improvement of a proposed design that results in the best properties for minimum cost, minimum weight or another objective. One of the simplest examples is determining the shape of a fence that will enclose the most area. If the fence can be any shape, but only a certain amount of fencing is available, then a circle will enclose the most area with the given amount of fencing. In order to minimize the amount of steel used in manufacturing a cylindrical tin can, a certain relationship between the diameter of the can and the height of the can is found. This will enclose a volume with the least amount of steel used for the surface area.

In each of these simple optimization examples, there have been two criteria - one was a criterion to be made best. In the fence, it was the enclosed area. In the tin can, it was the amount of steel in the body. The other criterion was a constraint on the design. In the fence, it was the amount of available fencing material. In the can, it was the specified volume to be enclosed.

In more elaborate problems encountered in engineering, there will be a property to be made best (optimized) such as weight or cost of a structure. Then there will be constraints, such as the load to be handled, and the strength of the steel that is available.

Constraints on the design are of two types. One is *Equality Constraints*. An equality constraint specifies a property of the design that must satisfy a specified value.

In our fence example, the length of the fencing (perimeter of the enclosure) was a certain number. This is an equality constraint. In a structure, the steel throughout the structure will need to be kept below the yield strength (localized stress concentration regions excepted). In many parts of the structure, the stress will be below yield stress. Consequently, there is an *Inequality Constraint*. In an inequality (or one-sided) constraint, a property of the design will be required to be kept above or below some limiting value.

Once a preliminary design has been developed, variations in some of the dimensions of the design can be evaluated. The particular dimensions will be permitted to be changed are; the degrees of freedom, known simply as variables. Some of the resulting properties of the design will be required not to exceed certain boundary values, or constraints. There may be constraints on the degrees of freedom, as well as on derived properties, such as the stress in a structure.

If the initial design were feasible, it would mean that the current solution did not violate any constraints. Variations on the design may result in properties that are an improvement. When the degrees of freedom have been set to values that give the best possible properties for the design, the design is said to be optimum. In the case of the fence example above, if we started out by trying a rectangular shape, and eventually arrived at the circle, we would have optimized the design. This would require that there was no constraint on the permitted shapes, such as requiring that the fence be rectangular.

3.5 Recommendations on Using FEA

- ✓ Check for incorrect application of loads and boundary conditions.
- ✓ Rotations can be prevented either by having constraints on translations at enough distinct nodes in space.
- ✓ A common check on results is to see whether the sums of the reaction forces at the constrained nodes equal the sums of the applied forces and gravity loads.

3.6 Summary

Besides giving the definition of finite element analysis, , in the past chapter we also defined and explained how FEA works, its capabilities when facing an optimization problem , its potential to analyze the behavior, deflection, stress levels, etc. In this chapter, we also covered some recommendations that should be followed in order to avoid misunderstandings at the moment of doing an FEA on robots, structures or mechanical devices.

4. FINITE ELEMENT ANALYSIS TO OPTIMIZE PRE-DESIGNED ROBOTS

As technology develops, robots have become very important due to the automation of many of the industrial and manufacturing processes. Also, the need to explore or work in hostile regions or areas that are subject to hostile environmental conditions such as the lack of oxygen, or presence of radiation requires the use of robots. For these reasons many researchers around the world are spending time to analyze, develop and improve machines to perform these dangerous and demanding tasks [3]. The final goal of this work is to develop a method for improving the design of a robot for high performance tracking and force control applications. One of the main points of designing a robust and versatile robot is to develop a solid geometry; as the two factors affecting robot geometry are the link shape and weight [4]. Robotic systems implement the concept *fighting of gravity* that means each component of the robot must be designed as light as possible in order for the arm to have less mass attracted by the earth, and also a larger payload for a given set of actuators.

Control, consumption of energy and efficiency are directly related to the weight of the robot. Therefore, this papeσr focuses on optimizing a robot structure by minimizing its weight through the use of an FEA package. When reducing the mass of a robot, the reduced inertial loads produced by lighter arms make it easier to control. The attention of this analysis will be centered in the first three links of the robot because they support the weight of practically the whole structure.

A robot design consists of many components all of which belong to one of the following:

a) Electronic parts: microchips, microprocessors and information component processors (controllers), etc.

b) Mechanical system: transmission systems such as, gears, belts, hydraulic cylinders, power systems like motors or pneumatic devices, etc.

c) Main structure or frame: It consists of the shell, housing, supports and protects all other components such as gears, chips, etc. It provides the structural rigidity to the robot. .

It is realized that the electronic component, chips and other mechanical devices are not likely to fail due to the payload or that the performance of the robot is going to be affected by any other external variable such as current flow, electric shock, etc.

With respect to the gear trains or any other transmission devices, it is assumed no failure will occur and that other components such as bearings, belts, chains and drivers are selected correctly.

In the FEA, a structure is broken down into many small simple blocks or elements. The behavior of an individual element can be described with a relatively simple set of equations. Just as the set of elements would be joined together to build the whole structure, the equations describing the behaviors of the individual elements are joined into an extremely large set of equations that describe the behavior of the whole structure. The computer then solves this large set of simultaneous equations. From the solution, the computer extracts the behavior of the individual elements. Thus, the deflection, strains and stress of the arms in the structure are defined. The stress levels will be compared to the allowable stress for the materials to be used in order to see if the structure is strong enough.

Pro/Mechanica is a way to deal with structures that are too complex to be dealt-with analytically. Pro/Mechanica deals with complex boundaries better than finite difference equations and gives answers to *real world* structural problems. The scope of finite element methods has been substantially extended during the roughly last forty years of its use.

Optimization packages such as Abaqus, Algor, Pro/Mechanica among others have been used for analyzing and improving geometries (to reduce weight) or checking when a part or even the entire robotic system is under failure mode. To prevent this to happen certain techniques such as Sensitivity Analysis are developed and implemented for this programs.

Sensitivity Analysis (SA) is the study of how the variation in the output of a model (numerical or otherwise) can be apportioned, qualitatively or quantitatively, to different sources of variation.

Originally, SA was created to deal simply with uncertainties in the input variables and model parameters. Over the course of time the ideas have been extended to incorporate model conceptual uncertainty, i.e. uncertainty in model structures, assumptions and specifications. As a whole, SA is used to increase the confidence in the model and its predictions, by providing an understanding of how the model response variables respond to changes in the inputs, be they data used to calibrate it, model structures, or factors, i.e. the model independent variables. SA is thus closely linked to uncertainty analysis (UA), which aims to quantify the overall uncertainty associated with the response as a result of uncertainties in the model input.

Abaqus has specialized add-on software that provides sensitivity analysis. This capability is useful for understanding the behavior of a design and/or predicting the effect of design changes with respect to design parameters. Abaqus design can compute both shape and material property sensitivities for general, static analysis of three-dimensional and ax symmetric continuum structures [13].

Algor is another important package that has FEA-based features for Mechanical Event Simulation software. Algor reduces the need for physical prototyping and eliminates the input dynamic loads by determining the motion, flexing and resulting stresses of a part or assembly at each instant of an event. Algor's Mechanical Event Simulations option makes performance and assemblies practical and produces realistic and accurate results [14].

A previous study carried out at MIT uses Pro/Mechanica to minimize the stress within a flexible body at a given displacement. Pro/Mechanica optimization module employs the FEM for computing, approximate numerical solutions to the deflection equations that predict the response of physical systems subjected to external influences. Unfortunately, the study shows no viability to a particular model selection and it is evaluated on one design parameter [5].

The objective is to evaluate how the stress varies as the selected design parameter varies. It would be desirable to reduce any of its dimensions in order to make the part lighter.

This thesis is concerned with the structure itself; how to improve robot's design by reconsidering some of the dimensions and checking iteratively against a possible mode of failure. Failure will be evaluated using the von Misses Failure Criteria when compared against the yield stress of the material of the arm in order to know when a plastic deformation begins.

The von Misses failure criterion is a theory based on the distortion energy in a given material; it is the energy associated with changes in the shape of the material.

A given component is safe as long as the maximum value of the distortion energy per unit of volume in the assigned material remains smaller that the distortion energy per unit volume required causing yield in a tensile-test specimen of the same material [15].

4.1 Problem Definition

When a robot is assigned to carry a payload, certain stress distribution develops along the arm. The characteristics of this stress depend on the material, the geometrical design of the robot arm and other external factors such as the interaction of the robot with the environment. It is necessary to know what the most critical configuration is in order to optimize the robot design. Although it depends on the overall kinematics' design of each robot in general this position corresponds to the configuration where the arm is fully extended so that the moment arm is maximized.

4.2 Approach

Reducing the weight or changing the shape of a robot is not an easy task to accomplish; there are several factors involved in this, such as the type of external load that the manipulator is subjected to, material used (links of the robot) and the most complex variable to handle is its shape (geometry). The complexity introduced by these factors make it awkward to calculate the stress levels by hand. For this reason, an FEA package is needed; in this particular case Pro/Mechanica is used. This software will calculate how the von Misses stress is distributed along the links (results obtained by bending, shear and torsion stresses). Results are compared to the to the permissible or yield stresses, which make it possible to know if any arm is under failure mode.

$$\sigma\text{' (Von Misses Stress)} \leq S_y \text{ (Yield Strength)} \tag{4.1}$$

To reach the optimum design some of the critical dimensions or specific measures to optimize (called design parameters) are varied within an established range. Then the stress is calculated as a function of the design parameter(s). Once the function is obtained and analyzed, the design parameter that optimizes a pre-determined performance index provides the optimum design. In this study, weight of the robot is selected as performance index. For a selected material, the problem becomes equivalent to finding the minimum volume. A design parameter, which produces the smallest volume for a link and simultaneously yields the lowest stress, is selected as optimal. This technique is applied to several parameters to reduce the weight of an industrial robot.

All of this will be possible by using an FEA. It can be used to calculate deflection, strain, stress, vibration, buckling behavior and many other phenomena. It can be used to analyze either small or large-scale deflections under a load or applied displacement. It can analyze elastic deformation, or "permanently bent out of shape" plastic deformation. A computer is required because of the large number of calculations are needed to analyze a complex structure [16].

This paper makes it possible to evaluate a detailed and complex structure on a computer during the design of the structure before it is actually built. The computer demonstration of the adequate strength of the structure and the possibility of improving the design during its planning stages can justify the cost of such analysis.

4.3 Optimization Using FEA

The way that Pro/Mechanica deals with optimization as the robot's design appearance is developed in Pro/Engineer, and deflection analysis in Pro/Mechanica, the FEA consists of the following steps:

4.3.1 Inside Pro/Engineer

- ✓ Define the model (robot): draw each component and create the parts that make the robotic system.
- ✓ The geometry needs to be well defined. So the design parameters could be selected

4.3.2 Inside Pro/Mechanica (FEA Package)

In an average engineering problem the redesign variables are:

- ✓ Characterize type of loading, materials, constraints, and type of connections between the arms.
- ✓ Identify the Design Parameter(s) (what variable(s) to optimize to minimize (or maximize) a selected performance index; i.e., weight of the robot).

To accomplish the goal of reducing the weight of the structure; therefore, improving its performance and the payload capacity, the weight distribution of the whole structure should be revised. Every link needs to be taken into account in order to avoid high inertial loads and an unstable robot design. A robotic design should follow the rule that the first link should be the most robust and the outermost as light as possible (The first link is going to hold the weight of the whole structure plus the payload) [16].

An optimization design process implies a search for a specific relational point where the robot still accomplishes a task and keeps its strength and maneuverability while making it possible to reduce manufacturing cost, materials and power supplied.

This optimization process will include and evaluate the proper dimension that an arm should have. The resulting design will be lighter but strong enough to avoid failure or excessive deflection that could cause improper performance of the whole structure.

It is very important to analyze the geometry of the parts; since the more complex the structure is the more difficult it is for the software to perform the task (mesh creating). It is also important to have the proper meshing size because otherwise the calculations done by Pro/Mechanica will not simulate the real behavior of the continuous system when checking for numerical convergence of the solution.

Once it is proceed with the selection of the design parameters, a FEA is desired to run in aim to compute the Von Misses Stresses' values on the any particular robot. Conceptually, the design of a robotic system is improved by implementing a good design strategy starting with the geometry and dimensioning of the parts.

4.4 Formulation of the Optimization Problem

In Pro/Mechanica (FEA Software) a sensitivity study will be developed to find the optimum values of some of the design parameters. Specific design goals for each design parameter will be to minimize mass, reduce a thickness and make sure that the maximum von Misses stress will not exceed the yield strength of the material used.

Figure 7 Schilling robot,
model titan 3.

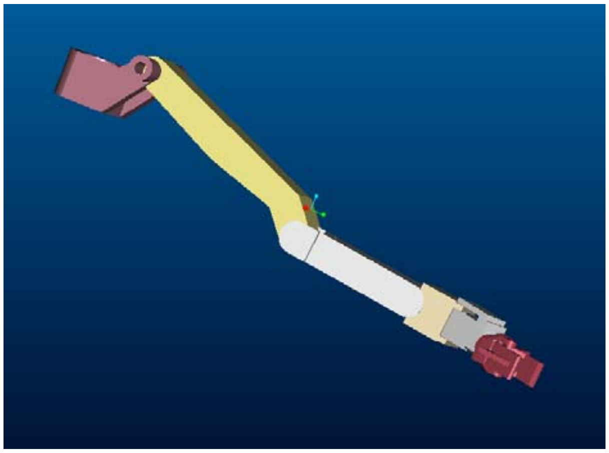


Figure 8 Model of the schilling robot created in pro/engineer.

It is recommended to use as few design parameters as possible. Sensitivity studies are helpful in deciding which design variables are important to evaluate. In particular, local sensitivity studies help to identify design parameters that do not affect the design significantly [16].

The next step is to select one of the following: which design parameter to pick or, which variable should be handled first? In order to perform a worthy analysis and obtaining no meaningful results, selection of design parameters must be made carefully. Since the type of stress that influences the most in the critical components of the robot is usually pure bending, we will take the following two aspects into consideration:

a) Geometrical Aspects:

When designing the basic shape of the robot, reducing the cross-sectional area translates into a lighter the arm (robot). The point here is to re-dimension the parts that are unlikely to fail. In other words, it is more important to take care of the parts that are subjected to a medium or low stress levels than the stressed ones. Usually, the critical parts when designing a robot are those that are positioned closer to the base of the robot.

Be aware that when reducing the thickness, second moment of area of the link's cross section in Equation (2) will decrease and the stress in Equation (1) will increase.

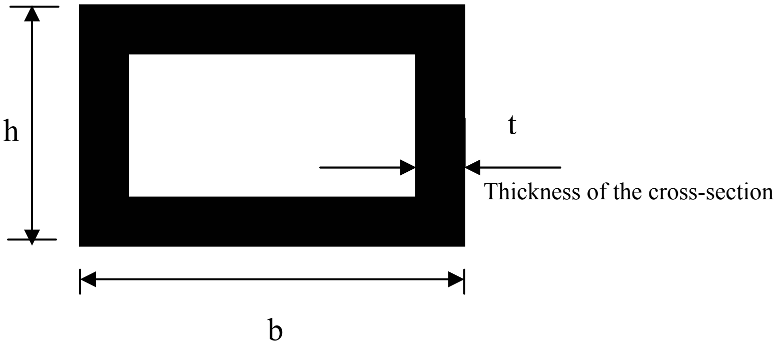


Figure 9 Cross-section of a beam.

$$\sigma = \frac{M \cdot c}{I} \tag{4.2}$$

$$I = \frac{1}{12}[b \cdot h^3 - (b-t) \cdot (h-t)^3] \tag{4.3}$$

Where σ is the bending stress and I is the second moment of the cross sectional area, t is the thickness of the cross-section, b is the base of the beam and h is the height.

The von Misses stress is greatly influenced by the second moment of area; therefore, it is very important to handle it correctly. All of the above arguments imply that: if the second moment of area is reduced, σ is expected to increase, and if the safety factor is decreased, this dimensions; therefore results in a heavier robot.

b) Second moment of inertia

This is also referred to inertia of the cross-sectional area, second moment of area varies drastically as h varies so that the first design parameters to consider must be the dimension h, no matter whether the link has a hollow cross section or not. (a hollow arm has an even more reduced second moment of inertia).

4.5 Summary

This chapter described a procedure to reduce the weight of a robot by using a FEA package, in this case Pro/Mechanica was used. The methodology used was to plot how the Von Misses stress and the mass of the arm change as every design parameter varies, then the optimal design is obtained by selecting the design parameter that provides the minimum weight and the lowest von misses stress level.

5. AUTOMATION OF ROBOT DESIGN STARTING FROM SCRATCH

Reduction of the weight or size of a robot through optimization produces more effective and capable robots, which also use less power and are more efficient. These attractive reasons and the lack of automated design procedures have motivated the current work.

An interesting work is carried out by Suh and Radcliffe in which linkage design theory is pointed out as a useful source of automated design. The organization of the “inventor's environment” generalizes existing linkage design practice. The task of the device is prescribed in terms of a desired workspace and the geometric constraints imposed by the device architecture are resolved to determine its physical dimensions and optimization strategy that balances compliance with the task specification against performance metrics [17].

5.1 Task Specification

The concept of robot motion planning is given below. The *configuration space* of a robot is the subset in which it can position its end-effector, often termed its combined reachable and dexterous workspace. Obstacles within the workspace of the robot create forbidden regions in its configuration space. In the free space, key frames are identified and interpolations between these frames define a desired end-effector’s motion. Any optimization procedure that minimizes a measure of distance between the task manifold and the workspace of a linkage primitive must necessarily depend on the choice of coordinate frames. This means that an optimum design computed in one coordinate frame will, in general, differ from that computed in another frame [17].

5.2 Weakness of Robots Design

Robots design configuration is hampered by the lack of established, well-known design rules, and designers cannot easily grasp the space can only design and evaluate several candidate configurations, though there may be thousands of competitive designs that should be investigated. In contrast, an automated approach to configuration synthesis can create tens of thousands of designs and measure the performance of each one without relying on previous experience or design rules [18].

5.3 Problem Definition and Approach

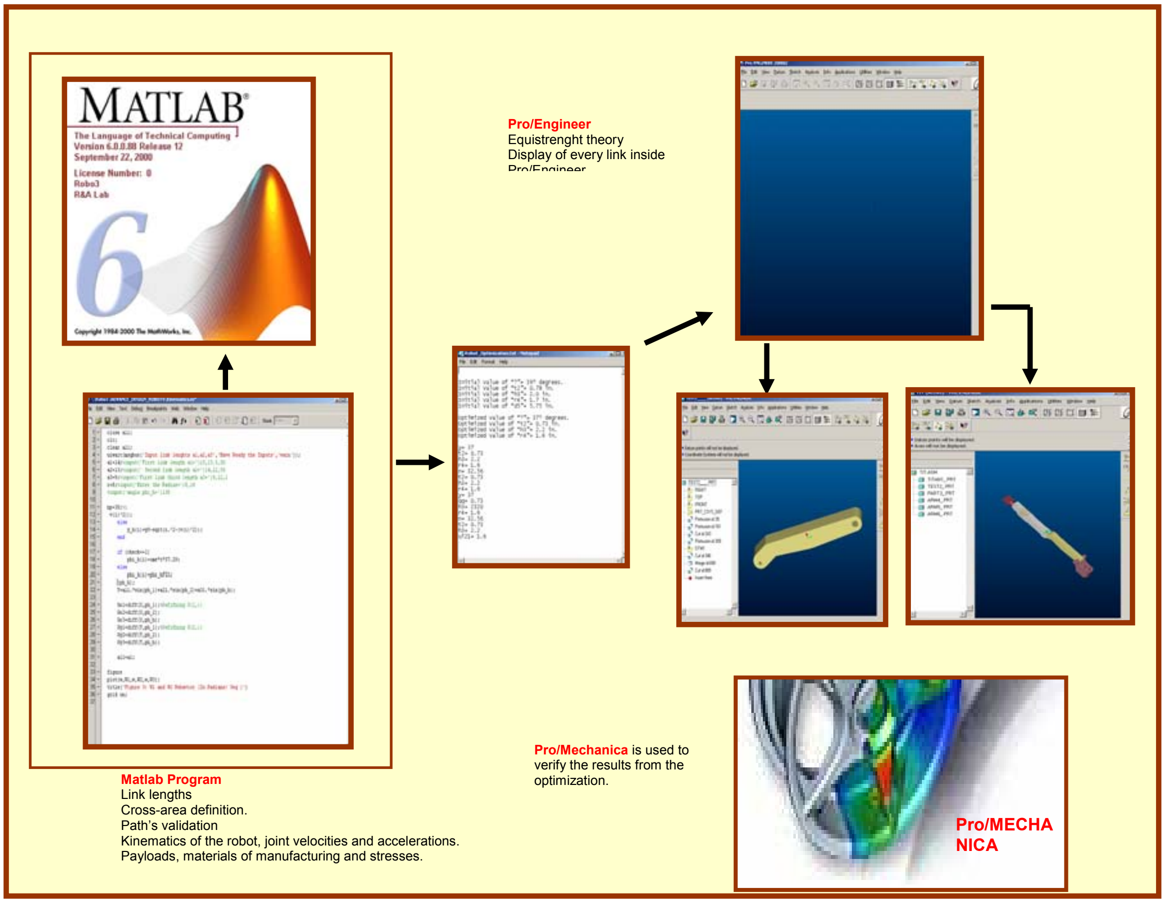
The main objective of this research is to develop an interface that will enable designers to get simplified and automated mechanisms to design robots. Below every single stage of the process is explained and detailed below.

It is composed by the two basic stages as indicated in Figure 10, one addressed in Matlab and another in Pro/Engineer.

The problem is described on terms of in what is given, and what it is needed to find.

Given: The inputs are material to manufacture the robot, safety factor of design, initial thickness of the hollowed links, payloads applied at the end-effector's position. And the constrained geometric definitions such as to have a larger link length in link 1 than 2 that will lead to a reasonable design.

Find: Al parameters that define the geometry are to be determined. This includes parameters such as link lengths, final or ideal thickness, and other fixed parameters that define link geometry.


MATLAB
The Language of Technical Computing
Version 6.0.0.88 Release 12
September 22, 2000
License Number: 0
Robo3
R&A Lab
6
Copyright 1984-2000 The MathWorks, Inc.
Pro/MECHA
NICA

Pro/Engineer
Equistrenght theory
Display of every link inside
Pro/Engineer

Pro/Mechanica is used to verify the results from the optimization.

Matlab Program
Link lengths
Cross-area definition.
Path's validation
Kinematics of the robot, joint velocities and accelerations.
Payloads, materials of manufacturing and stresses.

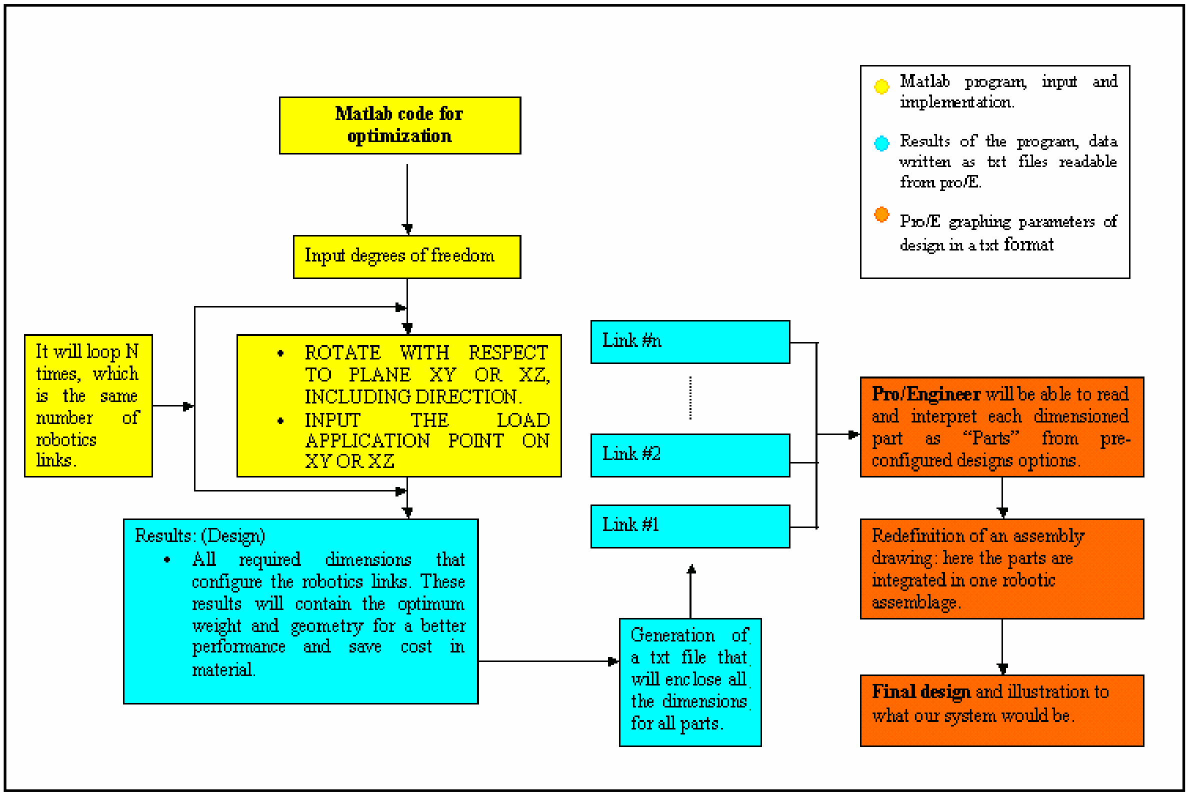
Figure 10 The two Stages of the automatic design process

The program code is entirely written in Matlab and it determines the optimal dimensions that the robotic manipulator must have.

By optimal dimension, we imply that the robot has the lightest arms that are able to withstand the external conditions and payloads.

This final configuration shape represents the optimum weight of the robot; the joint accelerations, velocities and the display of the robot could be taken as outputs of this program.

The problem focuses on finding the lightest possible structure that will meet the requirements. (Robotic System).



Matlab code for optimization
Input degrees of freedom
It will loop N times, which is the same number of robotics links.
• ROTATE WITH RESPECT TO PLANE XY OR XZ, INCLUDING DIRECTION.
• INPUT THE LOAD APPLICATION POINT ON XY OR XZ
Results: (Design)
• All required dimensions that configure the robotics links. These results will contain the optimum weight and geometry for a better performance and save cost in material.
Generation of a txt file that will enclose all the dimensions for all parts.
Link #n
Link #2
Link #1
● Matlab program, input and implementation.
● Results of the program, data written as txt files readable from pro/E.
● Pro/E graphing parameters of design in a txt format
Pro/Engineer will be able to read and interpret each dimensioned part as "Parts" from pre-configured designs options.
Redefinition of an assembly drawing: here the parts are integrated in one robotic assemblage.
Final design and illustration to what our system would be.

Figure 11 Program flux diagram

The optimum design of each link is defined by the objective function *f(x)* that depends on the stress (bending, torsion, and shear) and this is also affected by the weight of each arm which is described by the unknown design parameters. (See Appendix 2)

Mathematically, The design goals is *f(x)* : $f(x) = \sigma(x) + w(x)$ (4.4)

Where $\sigma(x)$ =bending stress (4.5)

And; $w(x)$=weight of the robot as a function of the link lengths. (4.6)

Therefore $\sigma(x) = f(P, x1, x2,..x12)$ and $w(x) = f(P, x1, x2,..x12)$ (4.7)

Being P=external payload at the end-effector’s position, x1=distance from the centroid of the first link to the ground, x1=distance from the centroid of the second link to the ground, x1=distance from the centroid of the third link to the ground.

In the case study that will be shown in chapter 8 the only type of stress is due to bending (consider that the robot moves in the plane) which is billed by the external (payload) and the internal loads (weight of the links themselves). The torsion and shear stress formulas are included to let the user to let the reader think headed about our 3-d model.

$$\sigma = \frac{\mathrm{M} \cdot \mathrm{c}}{\mathrm{I}}; \tag{4.8}$$

$$\tau(torsion) = \frac{\mathrm{T} \cdot \mathrm{r}}{\mathrm{J}} \tag{4.9}$$

$$\tau(torsion) = \frac{2 \cdot V \cdot q}{I} \tag{4.10}$$

The designer must keep in mind that the weight affetcs the bending stress and the bending stress is affected by the possible values that the design parameters could acquire, The design parameters are introduced into the optimization routine as an X vector (the unknown vector). According to this definition a maximum of 12 design parameters are to be solved in the optimization process. (4 per each of the links as shown in Figure 12). The links will be designed as hollow beams that changes in height as illutraded below:

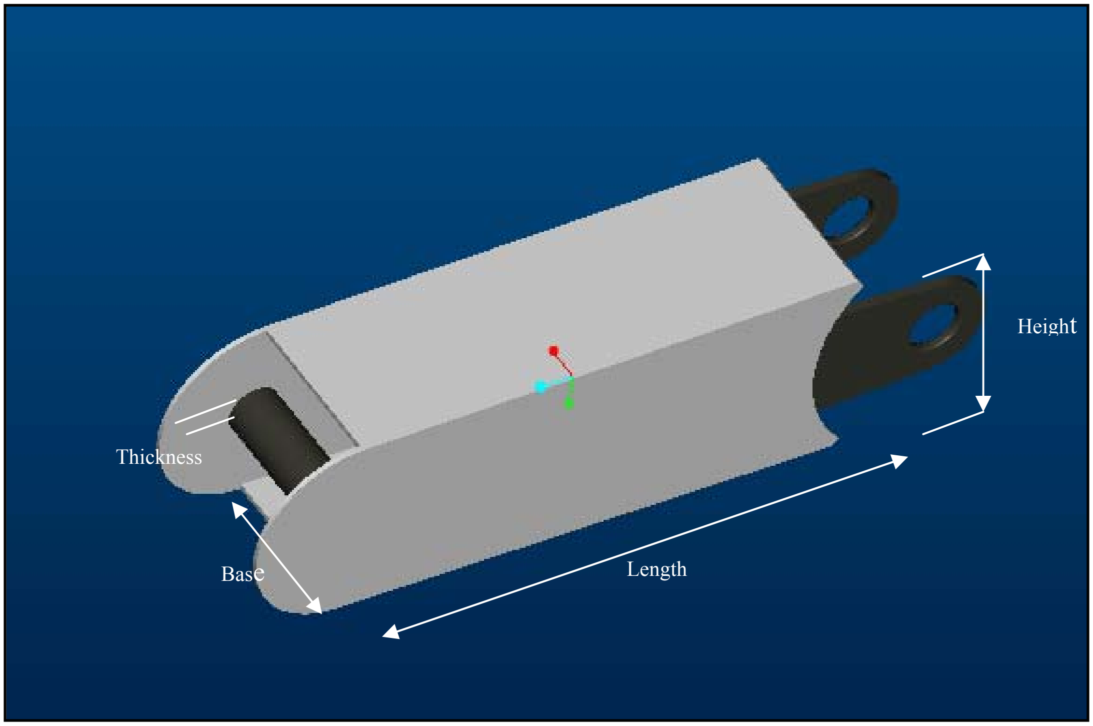


Figure 12 Pre-Configured design of each arm

5.5 Design Theory

The types of stresses applied on the robot are bending (neither shear nor torsion). So a good criteria evaluator (for general and particular cases) that compiles them is the Von Misses theory. The Von Mises failure criterion is a theory based on the distortion energy in a given material; it is the energy associated with changes in the shape of the material. A given component is safe as long as the maximum value of the distortion energy per unit of volume in the assigned material remains smaller that the distortion energy per unit volume required causing yield in a tensile-test specimen of the same material. This theory later will be used to secure that no failure will occur in any of the arms (links).

The program code is tailored to limit or define some of the parameters. It can be given three different approaches to run the optimization, solving for the design parameters according to the need. These three options are:

a) The thickness of the link #1,#2 and #3 are preset to have a specified value as t1, t2 and t3.

b) The bases of link #1,#2 and #3, have the same magnitude.

c) All twelve variables are left as design parameters.(we have 4 per arm).

Due to optimization feasability the second option is the one will be used in the case study (chapter 8).

5.6 Computer Interface

5.6.1 Matlab

The command used to achieve the optimization is the fmincom, and it is implemented as follows. x=fmincon('fc',x0,A,b,Aeq,beq,lb,ub) , where x is the unknown vector of design parametres and the other variables involved will be explained below.

The program’s optimization is defined and constrained by the next five definitions or constraints :

$$\begin{aligned} &1)c(x) \le 0 \\ &2)ceq\ (x) = 0 \\ &3)A * x \le b \\ &4)Aeq\ (x) = beq \\ &5)lb \le x \le ub \end{aligned} \tag{4.11}$$

Where $c(x)$ is a matrix coefficient that contains the non-linear inequality expressions, $ceq(x)$ is another matrix that contains the non-linear-equality terms, A is the matrix that contains the linear–inequality terms, b is a vector that contains the values for A, Aeq is the matrix that contains the linear–equality terms and beq is the matrix that defines assigns the values for Aeq. Ub (Upper boundary) and lb (a lower boundary) are vectors that restrict or limit the value for each design parameter. (In the case study chapter the dimension of these coefficients and matrixes will be given)

This problem was delimited by enforcing some measures to have certain value with respect to others, Once the X vector (values) are values are given by the optimization the next step in automating the design of a robot manipulator is to solve the kinematics inverse kinematics problem, This module (program, see apendix 1) solves for the joint angles at a given position and orientation in Cartesian space. This is a more complex problem than forward kinematics. Forward kinematics means to know the location or the specific movements of several articulated objects such as joints and links, in which the movement of one part affects all of the links and joints along the chain between it and the end effectors(the free end of the last link), the end effector's motion is the accumulation of all the motions of the upstream links. Inverse kinematics is exactly the opposite, it means that end-effector’s position is known, then the search for the movement of the links, how they must interact, link lenght is the task to accomplish. The complexity of this problem arises from the nature of the transformation equations, which are nonlinear. There are two issues in solving these equations: *existence of solutions* and *multiple solutions*. A solution can exist only if the given position and orientation lies within the workspace of the manipulator's end-effector. By workspace, we mean all points in space that can be reached by the manipulator's end-effector and the consistent equations that establishes the robot’s feasibility in order to fulfill every single point through the path and meet all criteria(be able to reach all path points, link one greater thank link two and so on). The results of an algorithm that checks the inverse kinematics when the robot follows a squared, round or any path function are shown:

A graphical result of the motion planning when the end effector’s position of the robot is depicted below (see appendix 1).

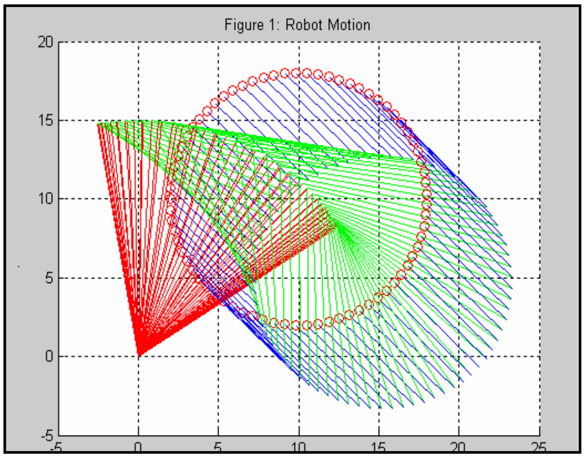


Figure 13 End-effector’s follows a circular path

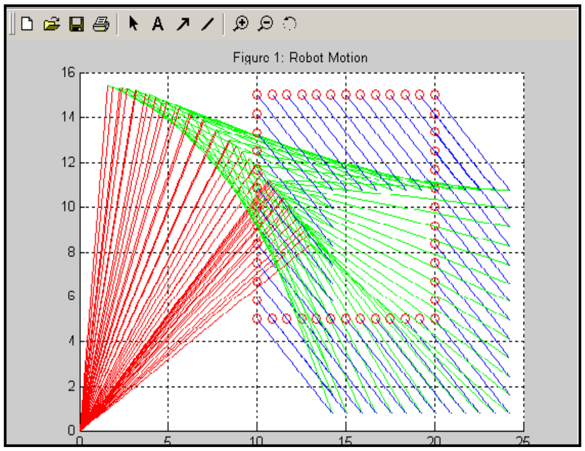


Figure 14 End-effector’s follows a rectangular path

5.6.2 Pro/Engineer

Once the optimal values are found for the optimization, a text file that contains all geometrical information of every link is written as a separate file and then extracted into Pro/Engineer that has.

Then to have a pre-configured model. This configuration shape will be able to regenerate it self and then adjusting their design parameters to the new calculated optimal values. The procedure is performed for every link and then finally for the assembly model.

Steps to perform are listed at follows:

1) Pre-configure the part in Pro/Engineer and define the design parameters that are design to be regenerated in the link of the robot.

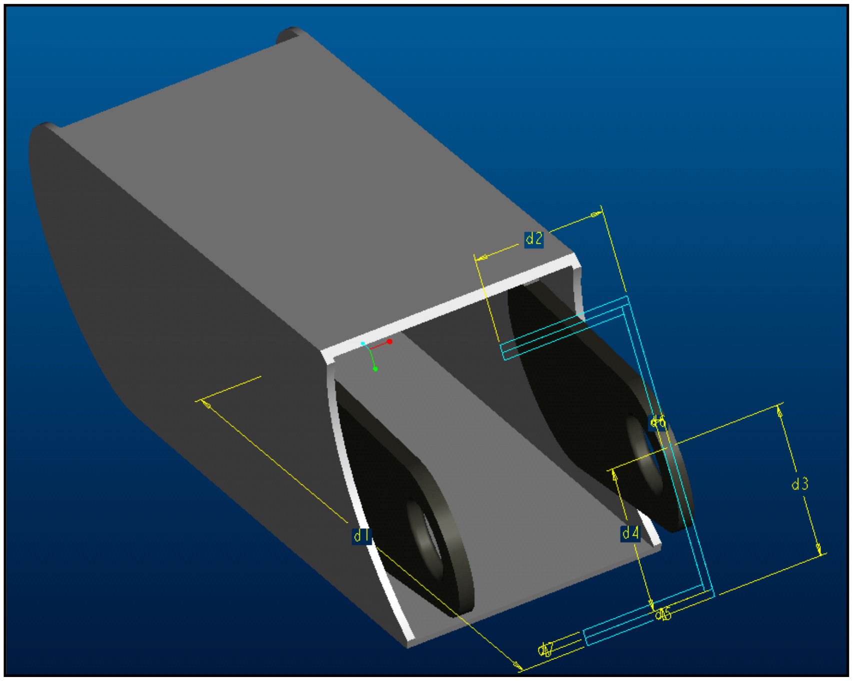
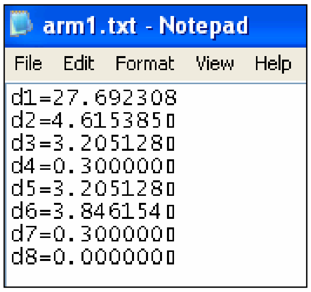


Figure 15 Design parameters selected along the arm.

2) Open the part (text file) relevant to the link to be designed.



```
d1=27.692308
d2=4.615385□
d3=3.205128□
d4=0.300000□
d5=3.205128□
d6=3.846154□
d7=0.300000□
d8=0.000000□
```

Figure 16 Outputs of the optimization program-

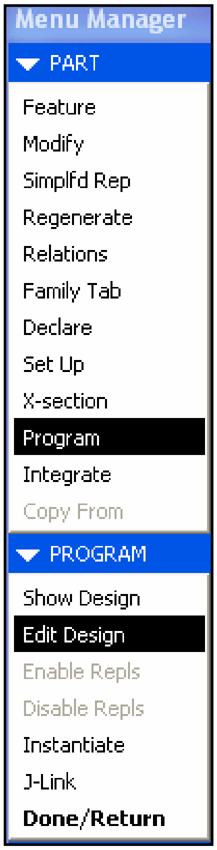
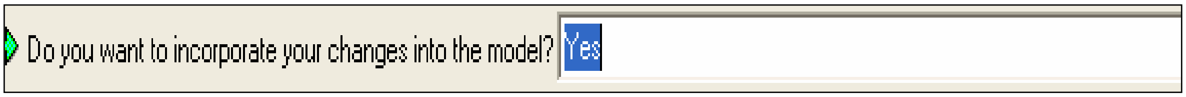


Figure 17 keying in the values for the design parameters to automate the regeneration



Do you want to incorporate your changes into the model? Yes

Figure 18 Prompted to make the changes into pro/engineer

3) In the Pro/Engineer window execute or open the file above by following this procedure (see Figure 18).

Menu > Manager > Program > edit > from file. > arm1.txt.. > press yes, when prompted to incorporate the changes.

4) Once this procedure is complete with every separate-designed part (repeat steps 2 and 3 and then save each link regenerated), the user will be ready to visualize the final propose (3-D assembly)

The total geometric information of the part is contained in this single script file that can be modified at any time, and later this part be regenerated (See Appendix 3).

The Pro/Engineer Assembly information is fully contained in this single script file that can re-assemble the parts at the user's convenience (See Appendix 4).

5.7 Shortcomings And Limitations Of The Constrained-Optimization Solution

Due to the very demanding nature of Multi-variable, non-linear optimization process we cannot guarantee that this is an absolute solution. It is possible to converge to a local minimum that may be ok as there are infinitely many possible solutions and we are merely selection one that attempts to improve the overall system characteristic by reducing the weight.

The solution will also somehow depend on the primary values input as initial ones in order for the program start iterating, there fore it is recommended to assign reasonable values to thickness, link lengths, etc. (i.e. the thickness is about 30 times smaller than the arm length), so it happens with the width of the robotic-arm. The program strictly keeps the formulation that relates these variables.

The optimization algorithm solves the 3-D geometry for the robot under the stress conditions that are developed in the plane. In the future work it is consider and advice what needs to be improved in order to extend this formulation to a full 3-D space characterization (see chapter 6, Table 1).

5.8 Advantages of Using Pro/Engineer

Many of the advantages of Pro/Engineer are provided the appendix 5; and recommendations for the regeneration process are given for the potential user of this program.

5.9 Summary

This chapter has described the approach taken in this thesis to optimize and design a robot from scratch. The procedure has two basic stages. The first stage is to implement the optimization program in Matlab in order to obtain the optimal-numerical values of the robot's geometry. Then the full display of the robot with the optimal design is carried out in Pro/Engineer.

6. PROBLEM DEFINITION

6.1 The 2-D Robot Motion and Stress

The problem was to design a multi-DOF revolute-joint robotic arm that is capable to trace a specific circular path. The robotic arm is defined as a 3-DOF robot, whose link's lengths should be appropriately selected so the end-effector's position can follow the prescribed path. In addition we had to evaluate the positions and velocities of the joints.

Input: Xo=Center of the Circle in the X axis, Yo=Center of the Circle in the X axis, and R=radius.

The program is also equipped to solve the links position, angles, velocities, etc. when the end effector's position follows a squared path (see Figure 19).

Depending on the user's desires, the end effector's angular velocity may be input in the program (see Figure 20). The user is prompted to input a numeric value if omega (radians of the end effector's angular velocity) is different than zero.

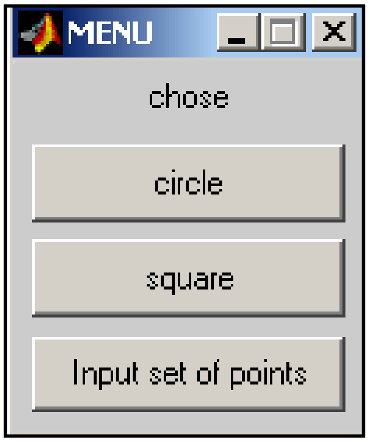


Figure 19 Program's graphic interface that allows the user to pick the path trajectory to follow

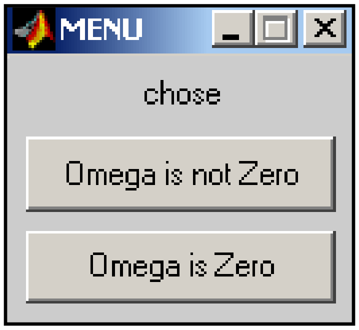
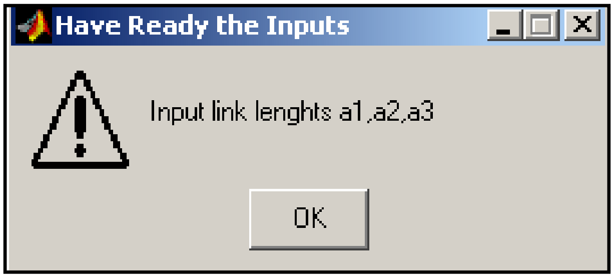


Figure 20 Program's graphic interface that allows the user to select the end-effector's angular velocity

The next prompted is let the user be ready and input the link lengths.



Input link lenghts a1,a2,a3

Figure 21 Input of the link-lengths. (optimized dimensions)

The Figure 22 represents the graphical movement of the links, including the end effector's position. (This is a way to verify that with this model the robot is able to cover every point along the path).

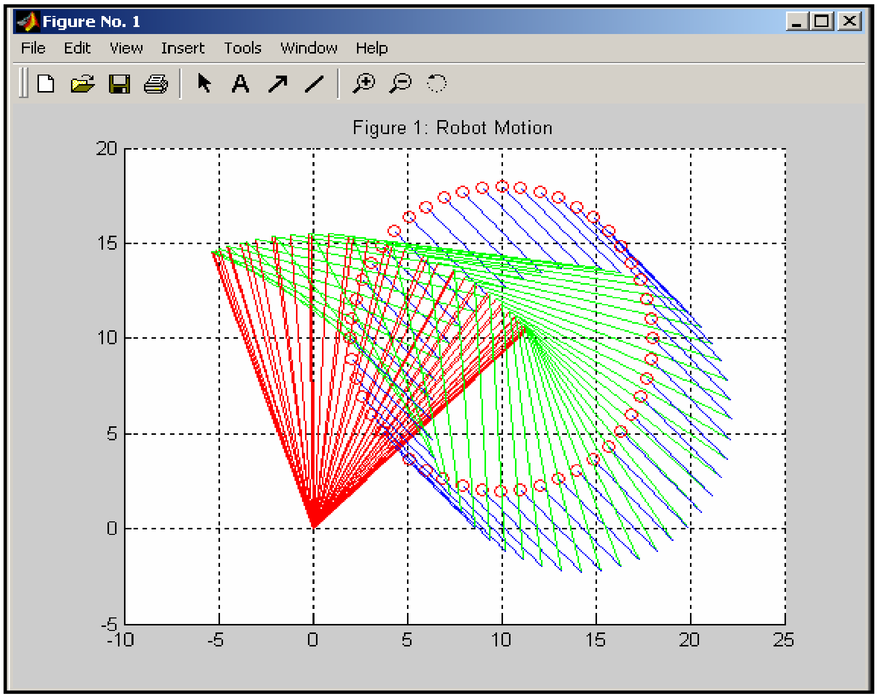


Figure 22 Display of the end-effector's motion

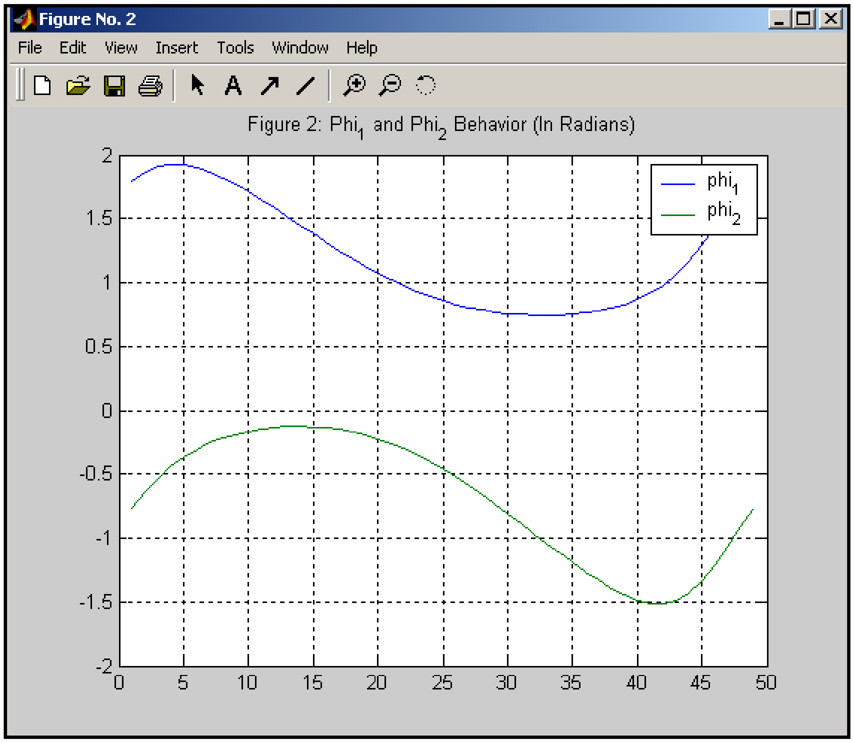


Figure 23 Display of the end-effector's displacement(in radians)

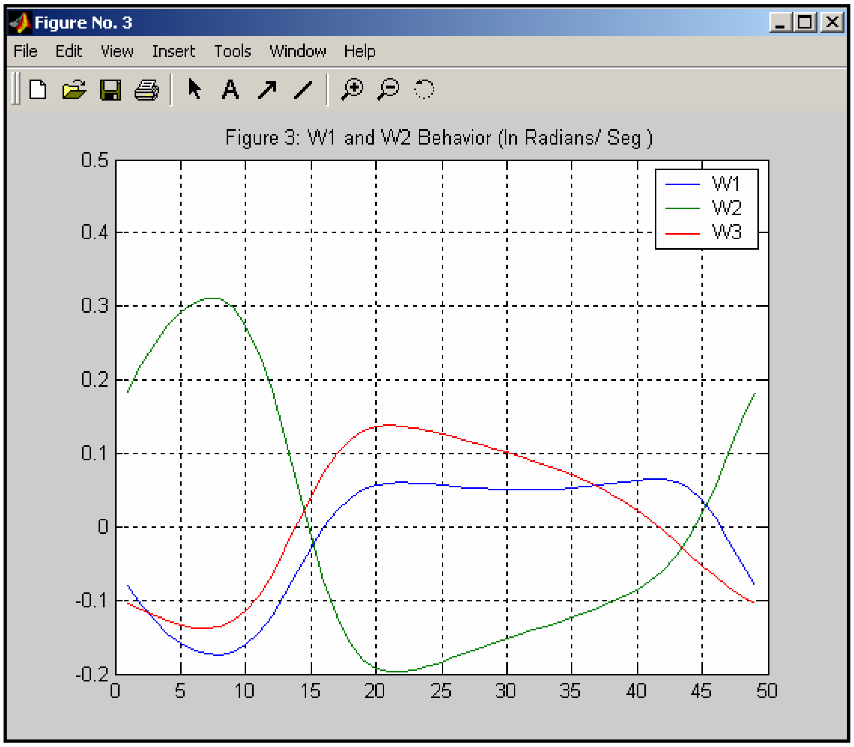


Figure 24 Display of the end-effector's angular velocity (in radians)

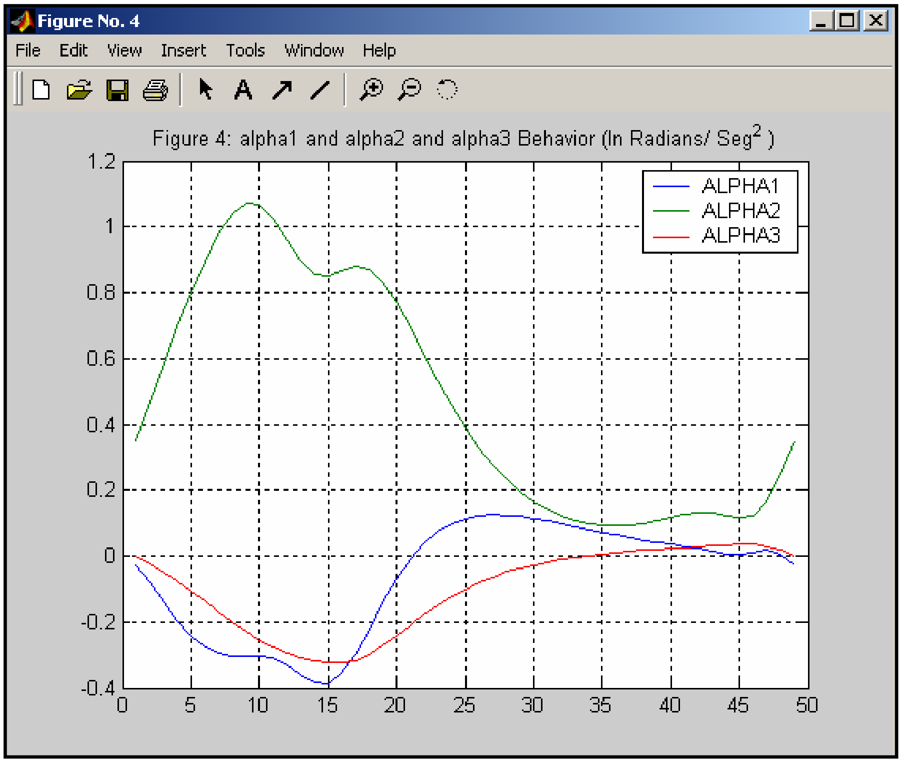
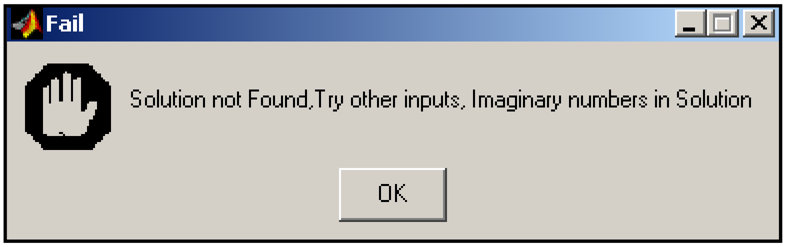


Figure 25 Display of the end-effector's angular acceleration

This program prompts the user if the given inputs are not suitable to provide a feasible solution of link lengths and follow the determined path gives what is shown below in Figure 26.



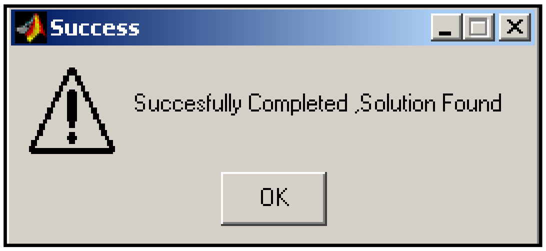
Solution not Found,Try other inputs, Imaginary numbers in Solution

Figure 26 Prompted given message that indicates if a solution is not found

A solution could no be found due to several reasons that involves the inverse kinematics' solution, these reasons could be:

1) Imaginary solutions
2) The arms are not physically reaching at least one point within the desired path

If the program finds a suitable solution the user is able to get the prompted message (see Figure 27)



Succesfully Completed ,Solution Found

Figure 27 Otherwise the designer is aware of the successful process

6.2 Extension of the Problem to the 3-D Space Configuration

Kinematics information about the robot manipulator is not a subject of study at this time, but it is given an insight to the stress behavioral problem that consist basically in three types of stress states further given.

In the stage before the robot was handled in the plane. (We only considered bending stress acting on the robots)

The problem in 3-D space behaves somehow different to the one in the planar space due to implications originated by the addition of other force components and stresses, such as: torsion stress and shear stress. Because of this incommensurate addition of forces we may end up dealing with a complex problem (for programming and optimization regulations), It is shown below certain configurations in 3-D space that causes the robot not only to have one stress condition that could be managed in the plane, but a combination of them, bending, torsion and shear stress that definitely lead us to a further step (see table 2).

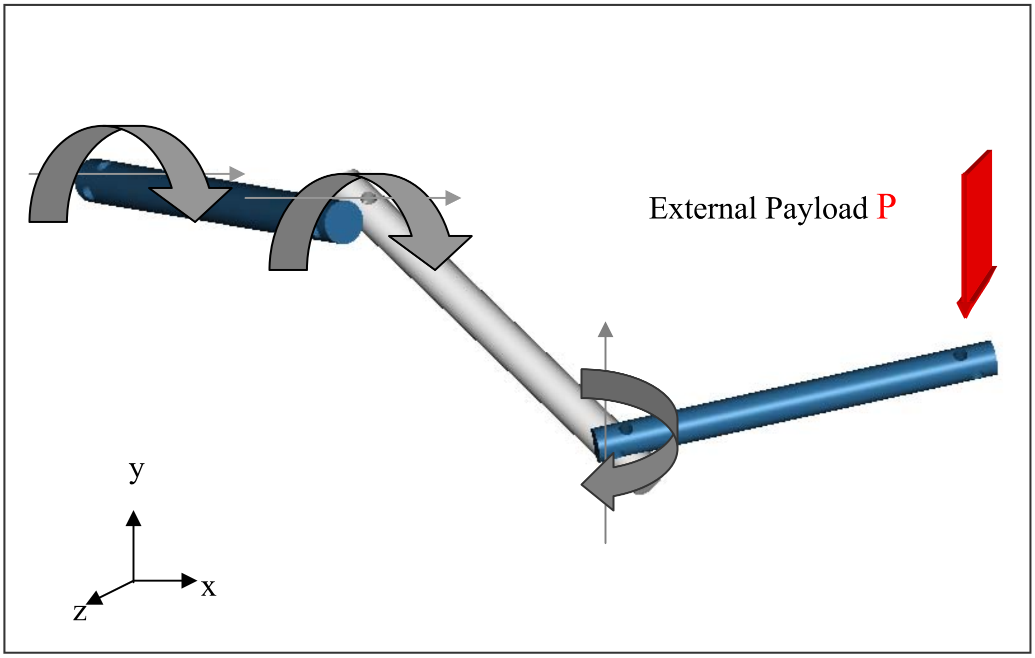


Figure 28 A possible configuration of a particular robot in the 3-D space

State of stress implication of adding the z component in our formulation inside the program (remember, code of the program is in appendix 2)

<table>
<tr><th colspan="2">TABLE 2. FORMULAS TO CALCULATE THE STRESS</th></tr>
<tr><td>Bending
$M3=P(l1+l2+x3) \rightarrow \sigma 3=\frac{M3\cdot c3}{I3}$
$M2=P(l2+x3) \rightarrow \sigma 2=\frac{M2\cdot c2}{I2}$
$M1=P(x3) \rightarrow \sigma 1=\frac{M1\cdot c1}{I1}$</td><td rowspan="3">Where:
P=external Payload
l1=length of link1
l2=length of link2
l3=length of link3
x3=projection of the length of the link1 on the plane XY
I3=Inertia Moment link3
I2=Inertia Moment link2
I1=Inertia Moment link1
σ3=bending stress on link 3
σ2=bending stress on link 2
σ1=bending stress on link 1
J3=Polar Moment of Inertia link 3
J2=Polar Moment of Inertia link 2
J1=Polar Moment of Inertia link 1
Γ3=Shear stress on link3
Γ2=Shear stress on link2
Γ1=Shear stress on link1
C3=centroid - moment of the cross area
C2=centroid - moment of the cross area.
C1=centroid - moment of the cross area.
Q3=first moment of area of link 3
Q2=first moment of area of link 2
Q1=first moment of area of link 1
T3=Thickness of link 3
T2=Thickness of link 2
T1=Thickness of link 1</td></tr>
<tr><td>Torsion
$T3= P3\cdot l3 \rightarrow \iota 3=\frac{T3\cdot c3}{J3}$
$T2=P3\cdot l3 \rightarrow \iota 2=\frac{T2\cdot c2}{J2}$
$T1=0 \rightarrow \iota 1=\frac{T1\cdot c1}{J1}$</td></tr>
<tr><td>Shear
$\Gamma 3=\frac{P\cdot Q3}{I3\cdot t3}$
$\Gamma 2=\frac{P\cdot Q2}{I2\cdot t2}$
$\Gamma 1=\frac{P\cdot Q1}{I1\cdot t1}$</td></tr>
</table>

With this definition of stress the design parameters *(x1,x2,..x12)* keep being the same, even though the program and hardware are more demanded. Now the next step will be to calculate the critical stress, consider the critical element, and add up the different type of stresses according to the von misses theory and introduce them into the program that takes care of the highlight debated here.

VM=sqtr(σ^2+3ι^2), then instead of using the stress quantifier S (S1 for link 1 and so on) (see appendix 2, use the new stress definition specified by the VM relationship(VM= Von Misses Stress).

7. EQUISTRONG PRINCIPLE

Equistrong design is the ability to design a mechanical component in such a manner that ideally every element in the part will withstand the same stress levels.

Equistrong designs require a uniform stress distribution along the whole structure. In other words, when a mechanical component is subjected to an external load it develops a stress distribution that must be equally disseminated thought out the device presenting the same stress value in every single point of the part.

Let us take a look to the basic formulations of the optimal design problems that use the equistrong principle:

7.1 Heavy Equistrong Bar

The set up of the problem consist to find the optimal design (cross sectional area and length) for a bar subjected to a normal-traction force as displayed in Figure 29.

$\sigma_b = \frac{N}{A}$;where, σ_b=Maximum allowable stress, σ_z=Design stress, A=Cross sectional area;

N= Normal force acting on A, ρ= density of the material, g= gravity force, V= Volume of the part.

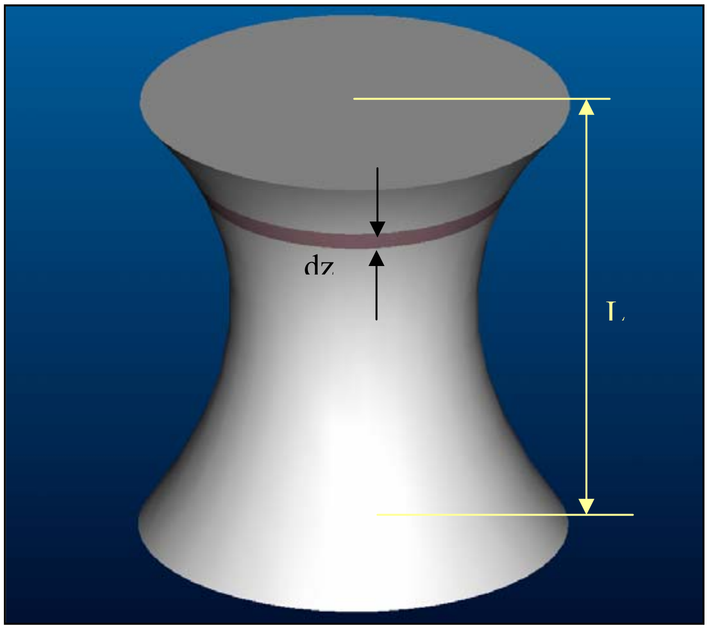


Figure 29 Heavy equistrong bar

let η=Safety Factor

$$\sigma_z = \eta \cdot \sigma b$$

$$\sigma_N = N \cdot g$$

$$dN = \rho \cdot g \cdot dV$$

$$dN = \rho \cdot g \cdot A(z) \cdot dz$$

$$(1)\quad \frac{dN}{dz} = \rho \cdot g \cdot A(z)$$

Here is shown the differential relationship between the Normal force and the length of the bar.

$$\sigma\ z = \frac{N}{A}$$

$$\eta \cdot \sigma_b = \frac{N}{A}$$

(2) A=$\frac{N}{\eta \cdot \sigma_b}$ -Relationship between the stress and the Area

Combining (1) and (2) we get:

$$\frac{dN}{dz} = \rho \cdot g \frac{N}{\eta \cdot \sigma_b}$$

Let $\lambda = \frac{\rho \cdot g}{\eta \cdot \sigma_b}$ -Introduction of the factor λ to simplify the formulation

$$\frac{dN}{dz} = \lambda$$

N= $\int \lambda \cdot dz$

Then, N=C·e $^{\lambda Z}$

To solve this integral it is necessary to input the boundary conditions to get the value of the constant C involved in this definition.

at z=ℓ, N=Nb -at the geometrical position of z=l the Normal force acting is the allowable

Nb=C* e $^{\lambda l}$

C=$\frac{N_b}{e^{\lambda l}}$

N=N$_b$ e $^{\lambda(Z-l)}$

Getting finally the formula that describes how the cross-sectional varies, as the bar is longer A(z)=$\frac{N_b \cdot e^{\lambda(z-l)}}{\eta \cdot \sigma_b}$ (7.1.1)

7.2 Galilei's Beam Problem

The set up of the problem consist to find the optimal design (cross sectional area and length) for a bar subjected to a bending payload at its extreme, as displayed in Figure 30.

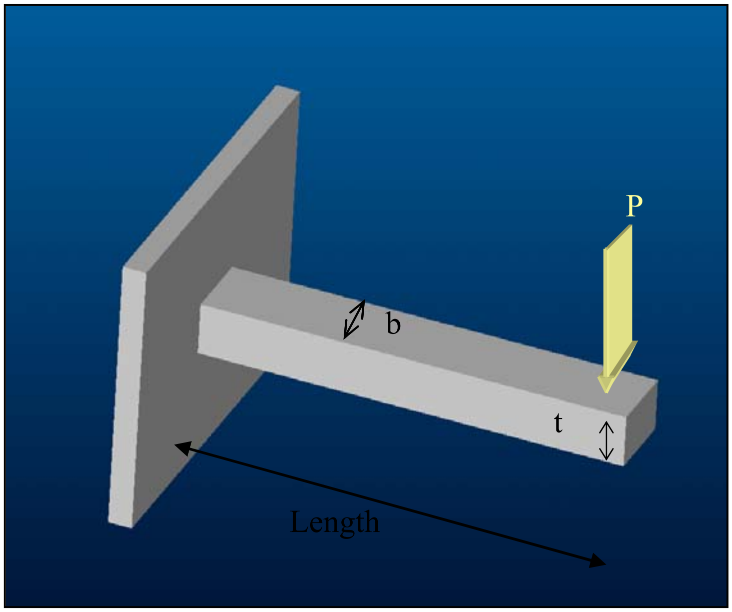


Figure 30 Heavy equistrong bar

σ x=Design bending stress, I=geometric moment of inertia, b=base of the beam, t= height of the beam, M=bending moment, η=safety factor, F=flexural load.

$\sigma_x = \frac{M \cdot C}{I}$ -Formula that defines the bending stress in a cantilever beam (7.2.1)

$I = \frac{b \cdot t^3}{12}$ -Formula that defines the geometric moment of inertia in a cantilever beam (7.2.2)

$\sigma_{max} = \sigma_{max}(C=t/2)$ –The maximum bending stress is exposed at the border of the height

Replacing (7.2.2.) into (7.1.1) we get , $\sigma_x = \frac{6 \cdot M}{b \cdot t^2}$ (7.2.3)

It is desirable that $\sigma_{max} = \eta \cdot \sigma_b$

M=F(l-x) - Variation of the moment of stress as the beam increases its length

Replacing M into eq.(7.2.3) we get, $\eta * \sigma_b = \frac{6 \cdot F(l-x)}{b \cdot t^2}$

Considering b constant it implies that the height varies along the beam as depicted in Figure 31 solving for t we get the final equation that calculates the appropriate height.

$$t(x) = [\frac{6 \cdot F(l-x)}{\eta \cdot \sigma_b \cdot b}]^{1/2} \qquad (7.2.3)$$

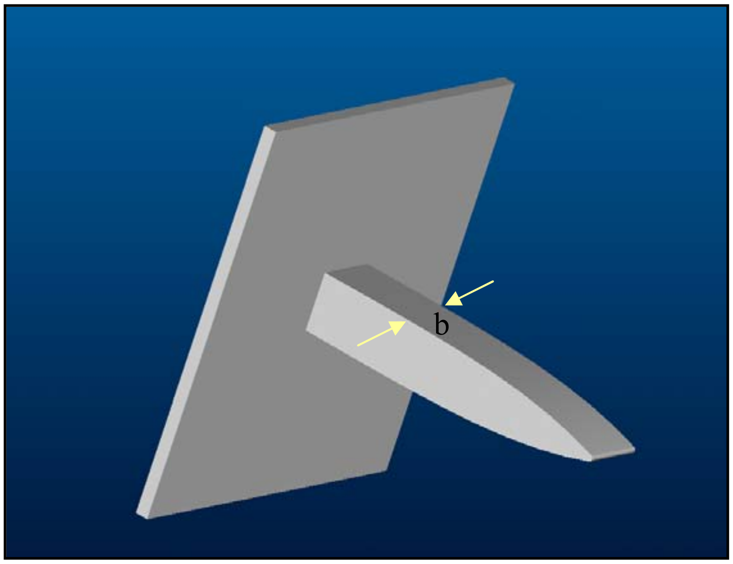


Figure 31 Heavy equistrong bar with a constant base

Considering t constant it implies that the base varies along the beam as depicted in Figure 32, solving for b we get the final equation that calculates the appropriate base.

$$b(x) = \frac{6 \cdot F(l-x)}{\eta \cdot \sigma_b \cdot t^2} \qquad (7.2.4)$$

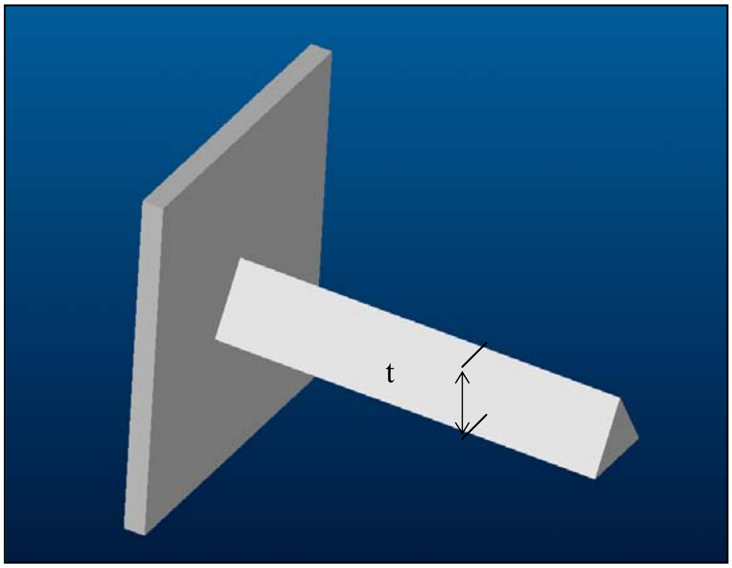


Figure 32 Heavy equistrong bar with a constant height

Concepts and advantages that involves optimal design problems using equistrong theory.

- ✓ This is an efficient criterion of optimal design.
- ✓ Methods for constructing permissible multitudes with high accuracy.
- ✓ Basic notions of parametric optimization.
- ✓ Problems of planning discretely equistrong systems.
- ✓ Problems of optimal distribution of matter in elastic thin-section structures.
- ✓ Optimal designing of plates.
- ✓ Optimization for proper value spectrum.

7.3 Implementation of the Equistrong Design to Complement and to Reduce the Weight of Robots

The basic concept of this theory is used to reduce additional weight of the beam that may be considered as a link of the robot (High performance shaped). This complementary model reduction procedure is introduced to improve the precision of the proposed method for beams with variable cross-section.

For simplification of this problem, the robot is analyzed with all the links in a fully extended configuration and a linear stress behavior is assumed. Stress increases linearly along a link, and the stress at the end of a link is the same as the stress at the beginning of the next link (below example of the stress behavior of link 2 is displayed, see Figure 33).

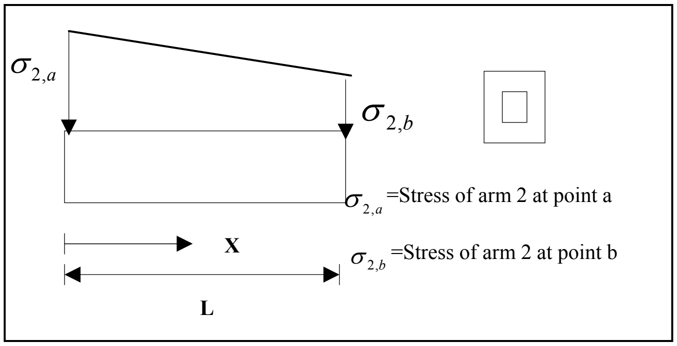

$\sigma_{2,a}$
$\sigma_{2,b}$
$\sigma_{2,a}$ =Stress of arm 2 at point a
X
$\sigma_{2,b}$ =Stress of arm 2 at point b
L

Figure 33 Display of linear stress distribution

The stresses at the ends of a link are calculated using the load applied on the robot and the distance to that point. Since we want the stress to be uniform, the cross-sectional areas at such points are varied accordingly. A constraint for defining the cross-sectional areas is that the wall thickness of each link must be constant. Although the design could be improved by varying the link thickness to get a more precise curve of stress distribution (equistrong theory), it will not be practical for manufacturing purposes.

Thus, once the areas are solved for the endpoints, the link geometry is completely defined, by assuming tha these two cross sections are linearly varied

7.4 Summary

In this chapter, an additional weight reduction factor is introduced to the optimal design. We covered the definition of the equistrong theory, its importance to reduce material in a mechanical component. Furthermore, a couple of meaningful examples are given to let the reader appreciate its basic principles. Later an implementation to our robotics design is put into action and applied.

8. OPTIMIZATION OF ROBOTS

8.1 Optimization of Pre-Designed Robots

Control, consumption of energy and efficiency are directly related to the weight of the robot. A lighter robot needs smaller gear size to provide its motion, the precision of the movement is better controlled, and in a few words reduction in cost is and increases in performance achieved. Next an illustrative example is given to demonstrate to the reader the optimization of a robot structure by minimizing its weight through the use of an FEA (Finite Elements Analysis) package.[22]

The complexity of the geometry that most of times is complex; this factor makes it awkward to calculate the stress levels by hand. For this reason, an FEA package is needed; in this particular case Pro/Mechanica is used. This software will calculate how the von Misses stress is distributed along the links (results obtained compile bending, shear and torsion stresses). The results will be compared to the permissible or yield stresses, which will make it possible to know if any link is under failure mode.

This work is concerned with the structure itself; how to improve robot's design by reconsidering some of the dimensions and checking iteratively against a possible mode of failure. The von Misses failure criterion is a theory based on the distortion energy in a given material; it deals with the energy associated with changes in the shape of the material.

8.1.1 Process Scheme of the Optimization Process (case study 1)

As the robot model is developed and implemented in Pro/Engineer, the stress and deflection analysis in Pro/Mechanica, the FEA consists of the following steps [22]:

8.1.1.1 Inside Pro/Engineer

- ✓ Define the robot model: Draw each component and create the parts that make the robotic system. The geometry needs to be well defined. In our example we use a Schilling robot as illustrated in Figure 34. Its Pro/Engineer representation is depicted in Figure 35.

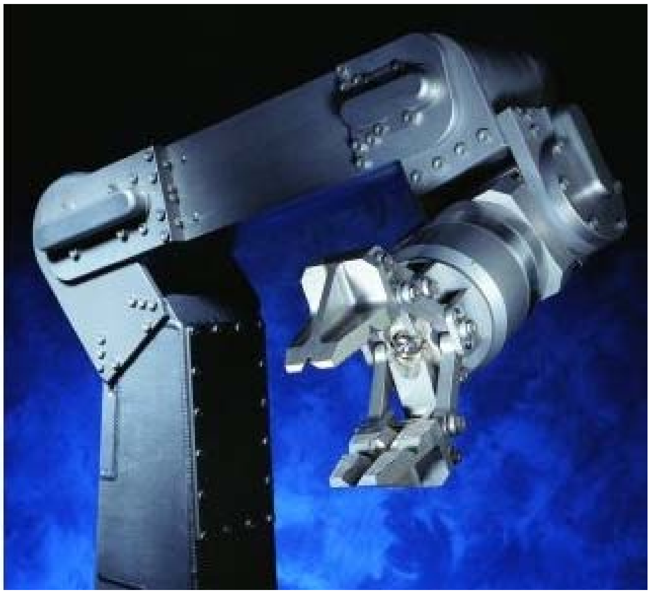


Figure 34 **Schilling robot**

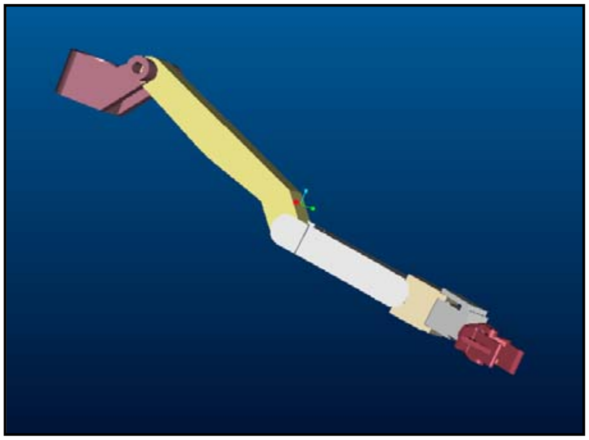


Figure 35 Model of the schilling robot created in pro/engineer

8.1.1.2 Inside Pro/Mechanica (FEA Package)

In an average engineering problem the redesign variables selected from the following set of variables:

1. Characterization of loading, materials, constraints, and type of connections between the links.
2. Then, design parameters are identified. That is, those design variables that will minimize or maximize a given performance index.

This optimization process will include and evaluate the proper dimension that a robotic arm should have. The resulting design will be lighter but strong enough to avoid failure or excessive deflection that could cause improper performance of the whole structure.

It is very important to analyze the geometry of the parts; since the more complex the structure is the more difficult it is for the software/hardware to perform the task (mesh creating). It is also important to have the proper meshing size because otherwise the calculations done by Pro/Mechanica will not simulate the real behavior of the continuous system when checking for numerical convergence of the solution.

The next step is to run an analysis where Pro/Mechanica plays an important role, it calculates the von Misses Stress on the Schilling robot. Conceptually, the design of a robotic system is improved by implementing a good design strategy departing from the geometry and dimensioning of the parts leading to a modification of the current arm, sometimes the shape is affected in this process.

8.1.2 Formulation of the Optimization Problem

The robot arm analyzed in example case study consists of a Schilling industrial robot used for oil exploration made out of titanium (see Figure 34). For the purpose of simplicity, the following assumptions will be applied to the model used in this study. Due to the large number of equations used by FEA, no hydrostatic load will be applied and only the first three links will be analyzed. Furthermore, instead of titanium, the material used in this model is selected as steel (with a yield strength of 62,000 psi). As illustrated in Figure 36, a load of 1,500 lb will be applied and the safety factor is set to a value of 3.

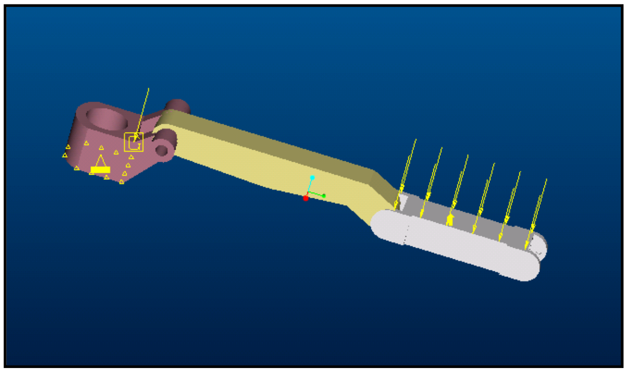


Figure 36 Model simplification of the schilling robot to be analyzed

A sensitivity study is developed to find the optimum values of some of the design parameters. Specific design goals for each design parameter is to minimize mass, reduce a material thickness of the links and make sure that the maximum von Misses stress will not exceed the yield strength of the material used.

When reducing the mass of a robot, the reduced inertial loads produced by lighter arms make it easier to control the robot itself. The attention of this analysis will be centered in the first three links of the robot because they support the weight of practically the whole structure

It is recommended to choose as few design parameters as possible, since dimensionality of the problem increases the complexity of the optimization problem and numerical difficulties (precision, even numerical instabilities) as well. Sensitivity studies are helpful in deciding which design variables are important to evaluate. In particular, local sensitivity studies help to identify design parameters that do not affect the design significantly [19].

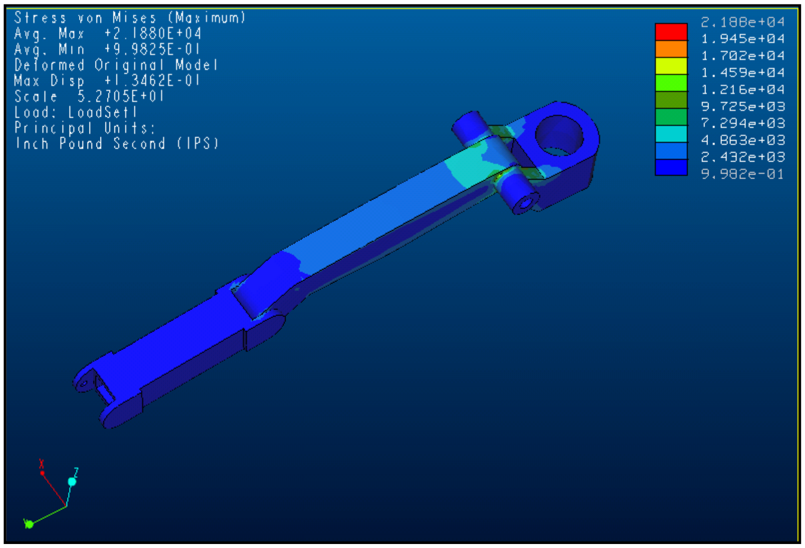


Figure 37 FEA that calculates the Von Misses stress distribution before optimization.

Now based on this stress distribution, we proceed with the selection of design parameters. The bottom line is to reach a certain point where the stress achieves its minimal level while searching for those design parameter that yield a lighter link or robot. Moreover, the optimization process may not only consider the *critical dimensions* but also some others that are not subjected to a high stress on the display.

Parameter Number	Notation	Description
1	α	Base angle as shown in Figure 38
2	t2	Thickness of link 2 as shown in Figure 39
3	h3	Dimension of link 3 as shown in Figure 40
4	r4	Radius of the hole in link 3 as shown in Figure 41
4	d5	Diameter of the hole in link 1 as shown in Figure 42

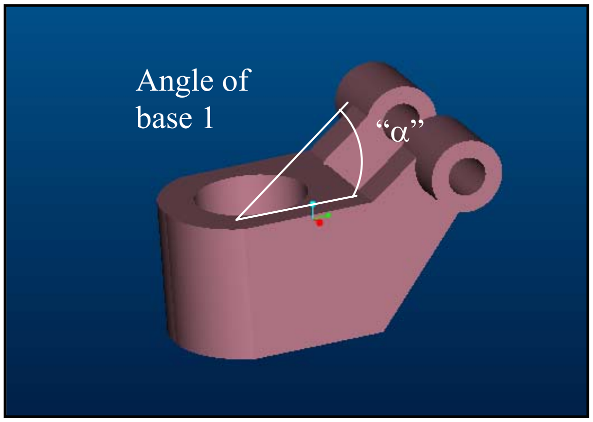


Figure 38 Design parameter 1

As indicated in the Figure above α (angle of base 1) is chosen, as the first design parameter because despite it will not reduce the weight significantly, the stress distribution will vary remarkably. The initial value of this parameter is set to 39°degrees

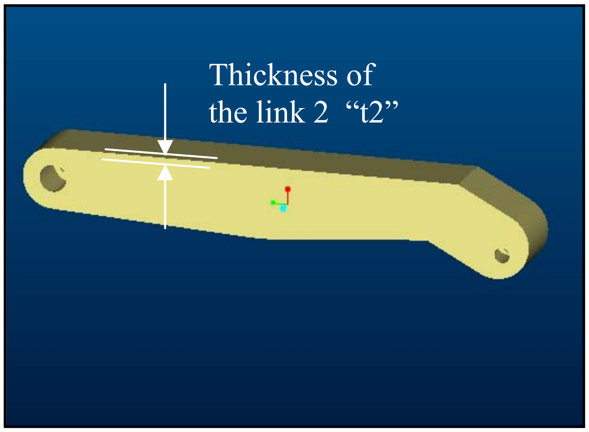


Figure 39 Design parameter 2.

The second design parameter, t2, is selected as indicated in Figure 39 because it is considered that by redefining this geometry the stress distribution will greatly vary. The initial value of this parameter is set to 0.78 inch

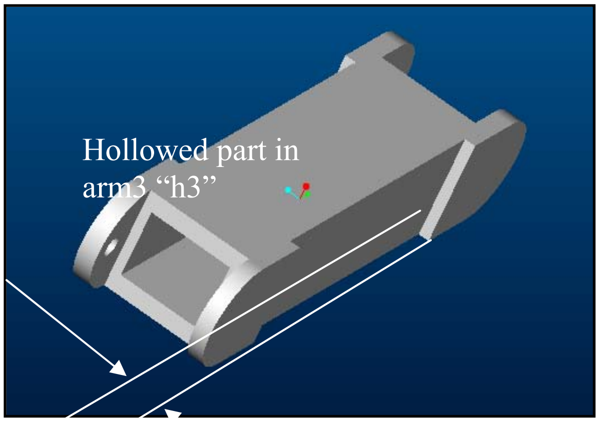


Figure 40 Design parameter

As designated in the Figure above, h3 is chosen as half of the hollowed part of link 3. Since the outside dimensions of this link are fixed, increasing this geometry will make the walls of the member thinner causing the link to become lighter. The initial value of this parameter is set to 2.0 inches

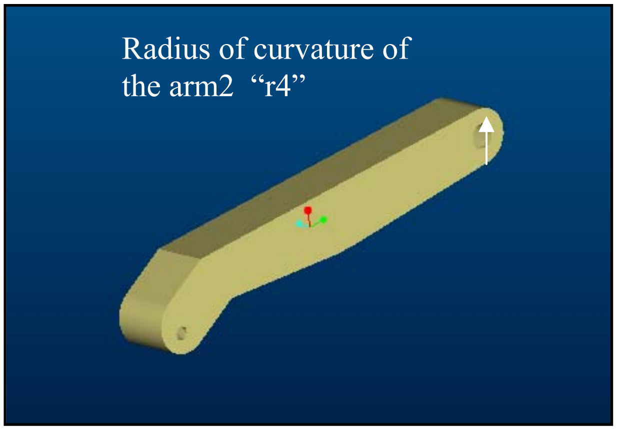


Figure 41 Design parameter 4

Design parameter r4 defines the thickness of the link 2 at its joint with link 1. It is chosen to visualize how much the stress is distributed on the link. The initial value of this parameter is set to 1.6 inches

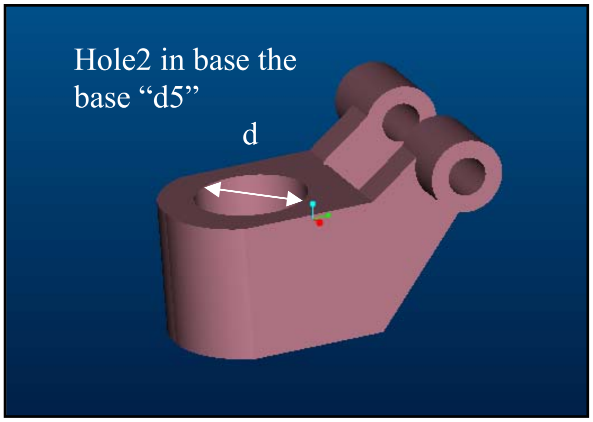


Figure 42 Design parameter 5

The diameter d5 is selected as the fifth design parameter (The diameter of the hole inside the Base. The weight can be reduced by increasing the hole size without increasing the stress levels significantly. (The initial value of this parameter is set to 5.75 inches)

8.1.3 Results of the Analysis

The following charts illustrate the changes both in weight and stress of the parts as the individual design parameters were changed. The point of interest is where the maximum strength is achieved while maintaining the minimum weight.

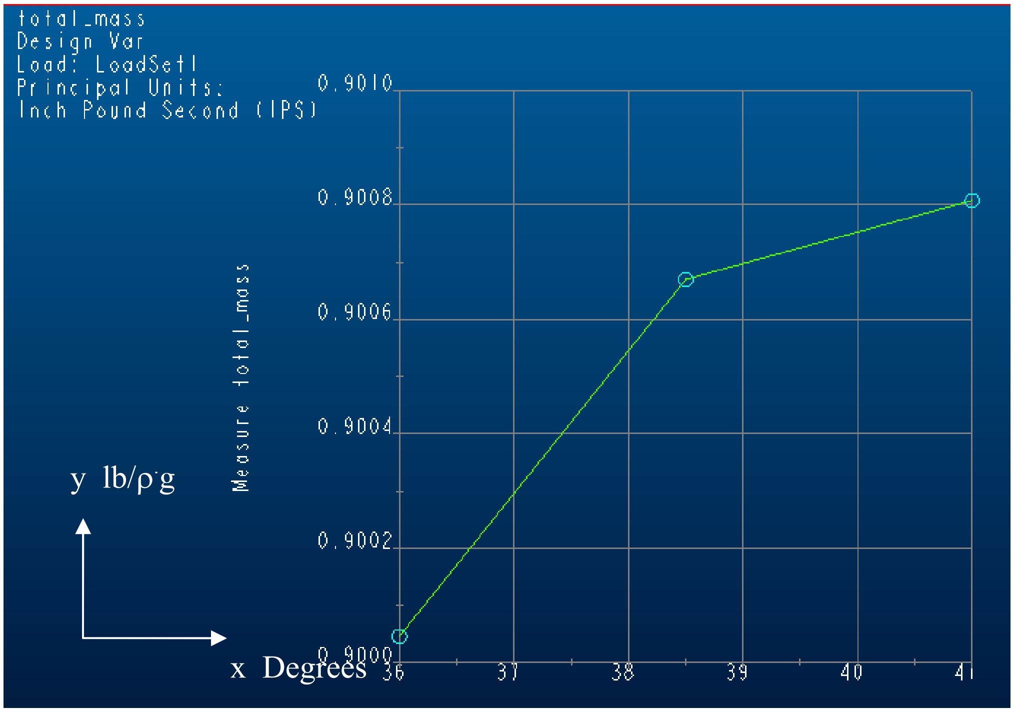


Figure 43 Mass vs. α

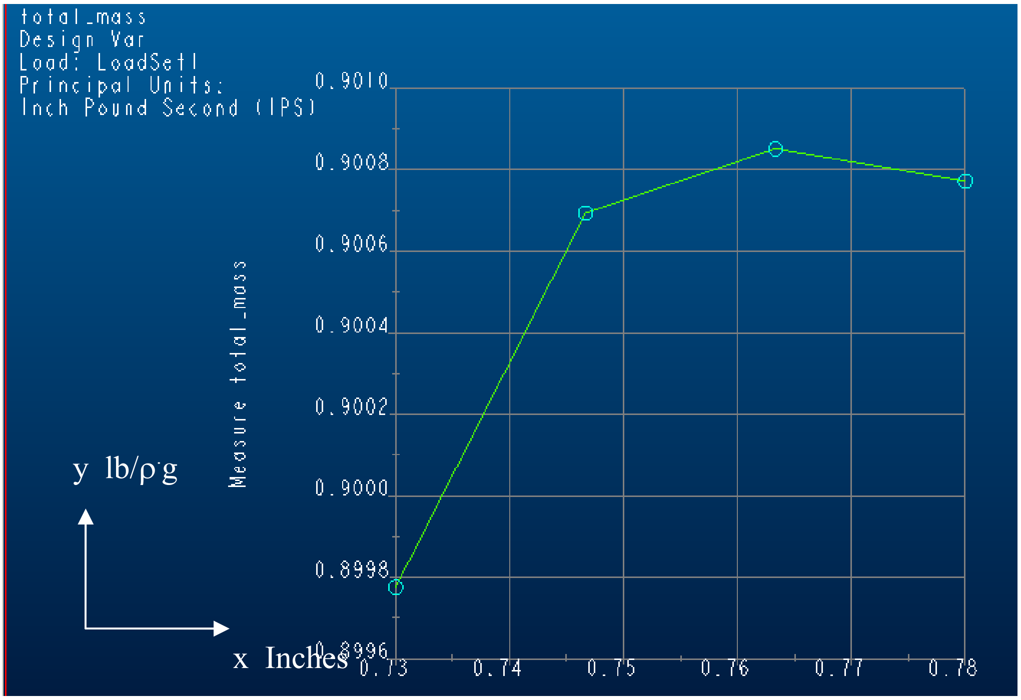


Figure 44 Mass vs. t2

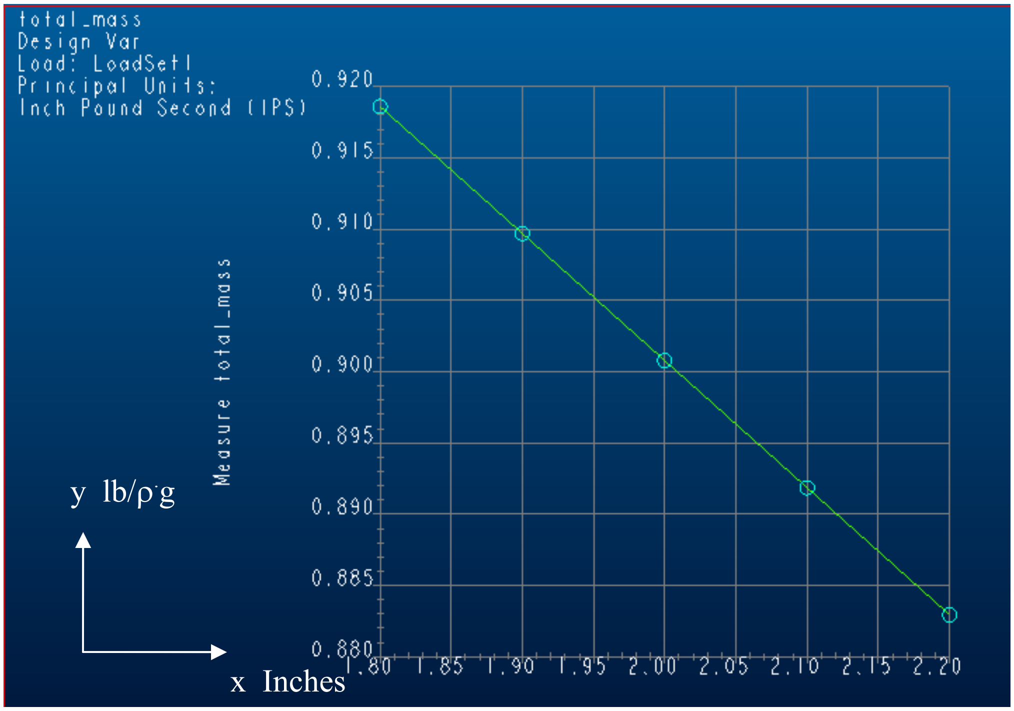


Figure 45 Mass vs. h3

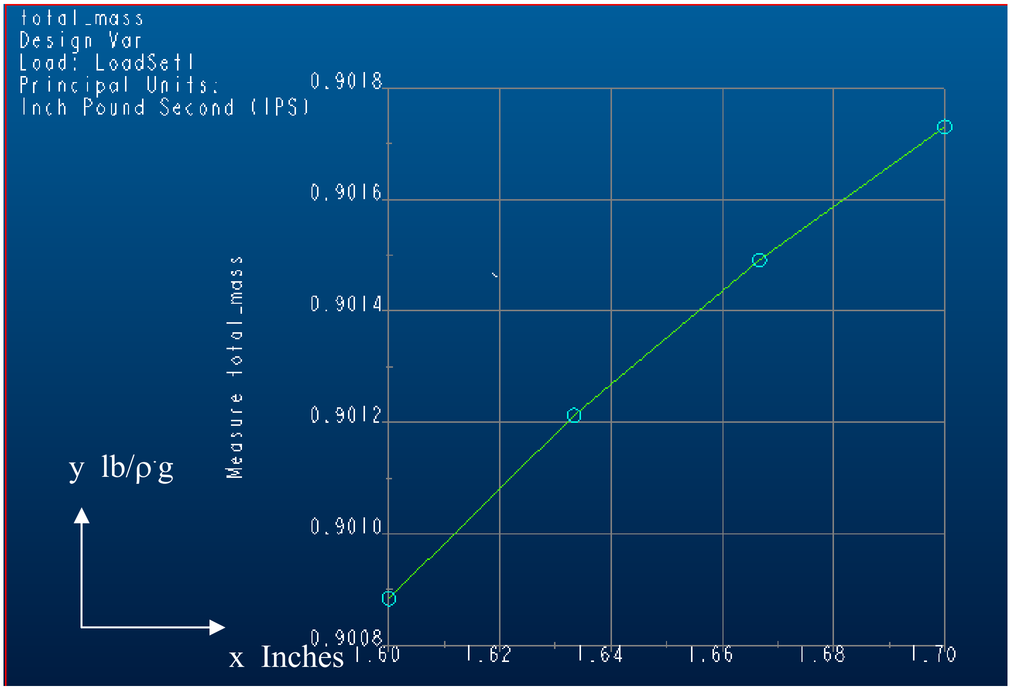


Figure 46 Mass vs. r4

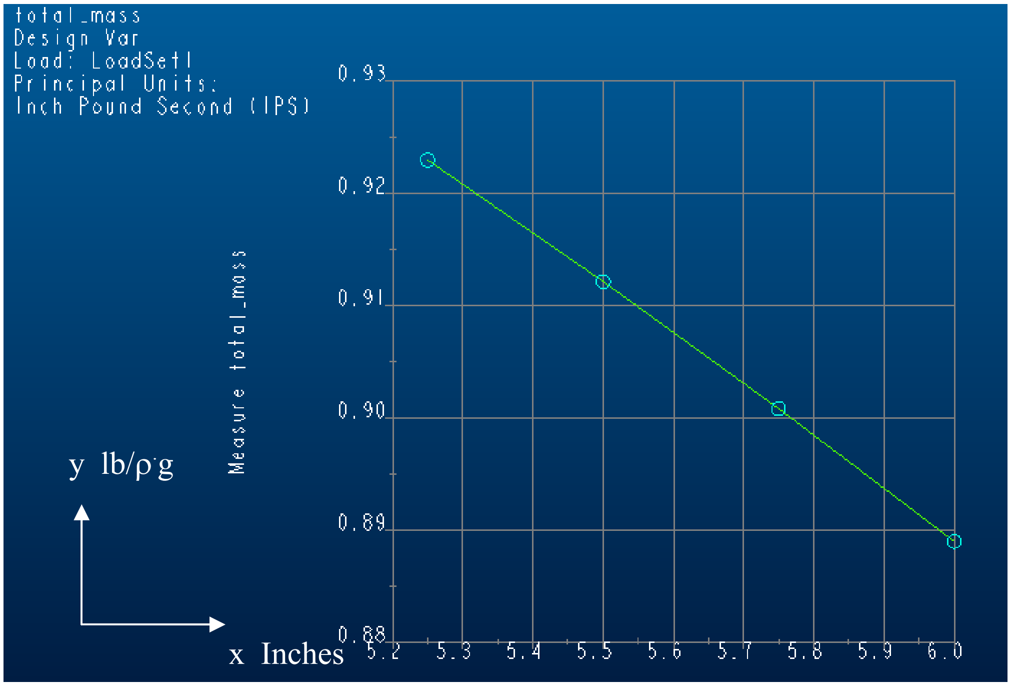


Figure 47 Mass vs. d5

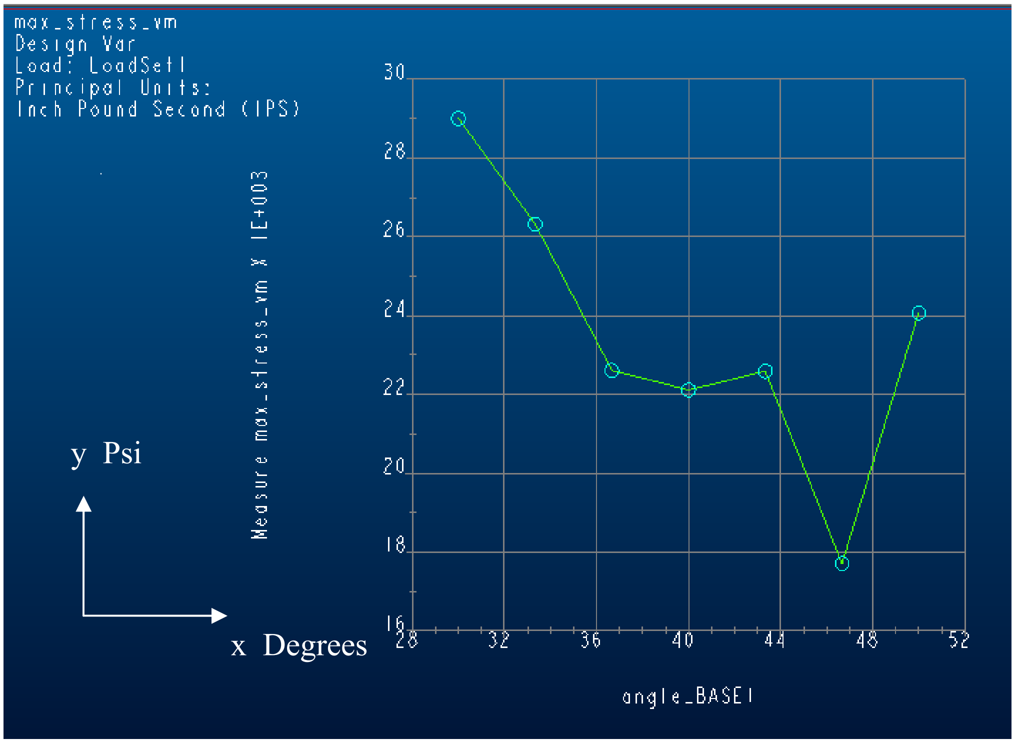


Figure 48 Von Misses Stress level vs. α

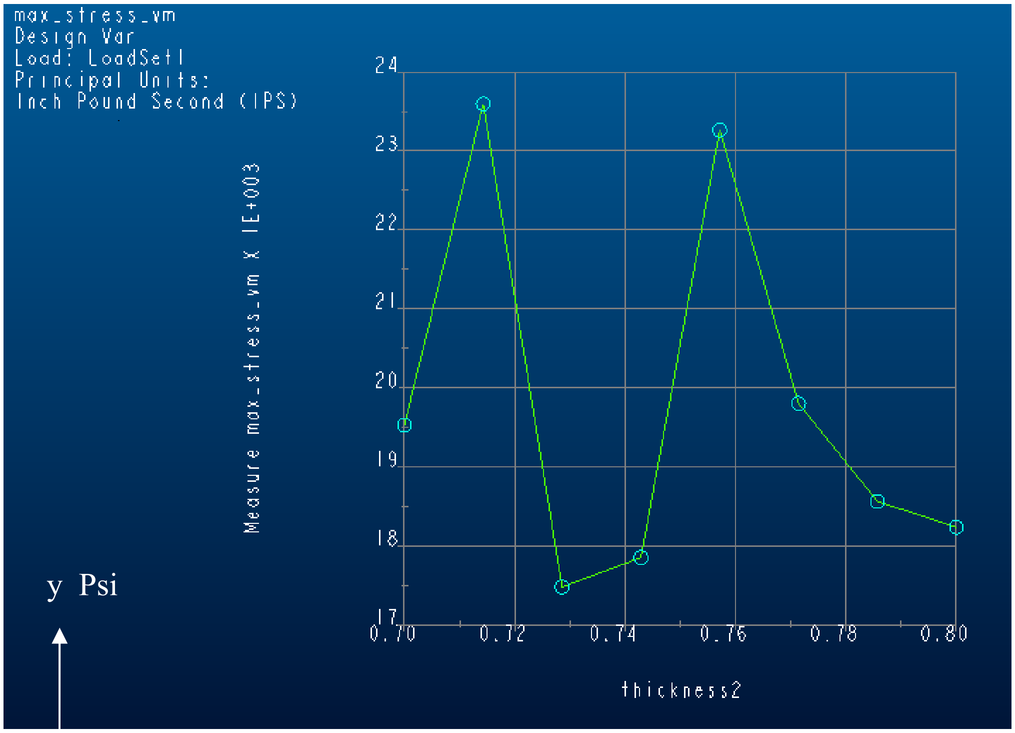


Figure 49 Von Misses Stress level vs. t2

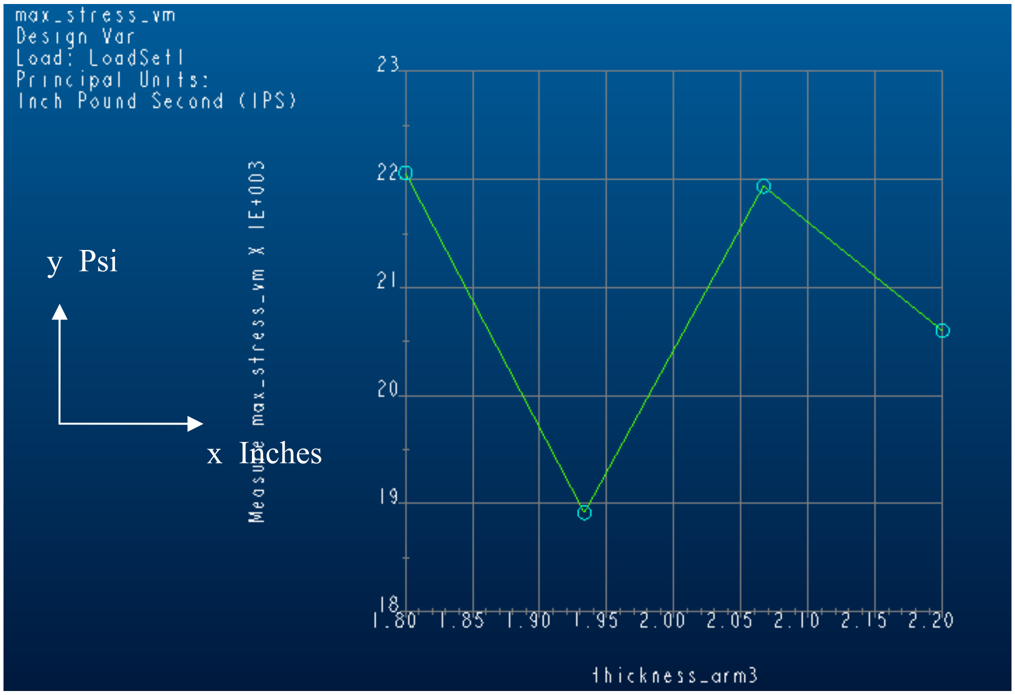


Figure 50 Von Misses Stress level vs. h3

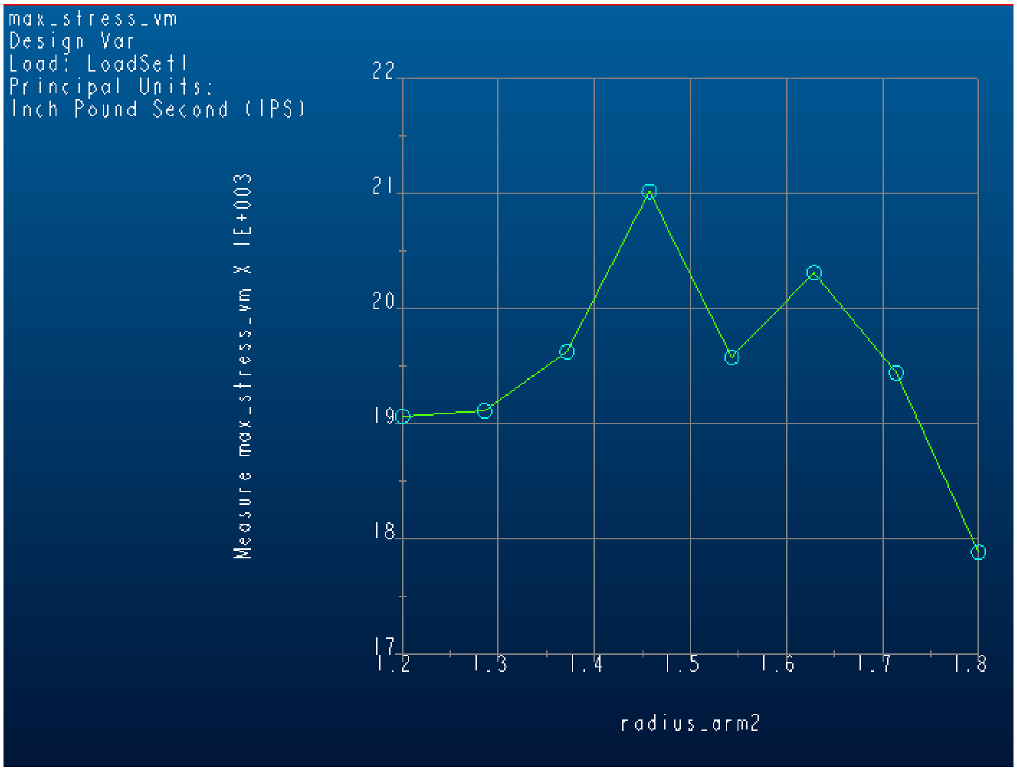


Figure 51 Von Misses Stress level vs. r4

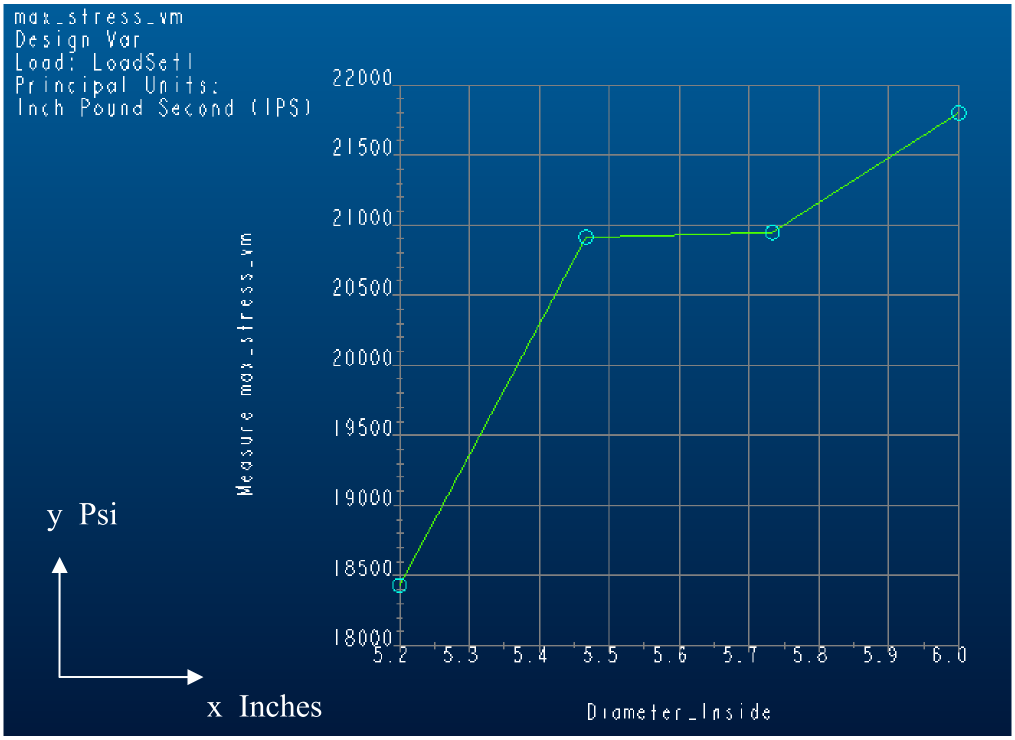


Figure 52 Von Misses Stress level vs. d5

8.1.4 Discussion of Results

In Figures 43 through 47, in general almost a linear relationship trend is observed between the selected design parameters and the mass of the corresponding link (as it was expected). It is not likely to find many local inflexion points in those graphs because the selected parameters involved belong to a not very complex geometry. In other words, the more robust (bigger) the link is, the heavier the mass. It is interesting to visualize how the stress level changes for a particular point when a selected dimension of the link (design parameters) is varied.

Interpreting the results became a non-straightforward task due to the following reason: when selecting a design parameter, Pro/Mechanica varies the initial value but in some instances, the resulting value yields an impossible geometry. Some examples will be a wall with a thickness of 0 or even negative value, or an open geometry. The first aspect to consider is the feasibility of this *new geometrical configuration*. For Instance, consider the following arguments.

In Figure 48 it can be seen that when the base angle (design parameter 1) is close to 46.75 degrees, the lowest von Misses stress is obtained. However, 46.75 is not the exact solution because in order to set the value of design parameter α above 42 degrees, it is necessary to vary the joint radius, which is a fixed dimension. Therefore, the FEA result will be slightly different to what it is expected to be. Even though the Sensitivity Analysis gives a very good result to the real behavior of design parameters (see Figure 53).

In Figure 49 it can be seen that the lowest stress is reached when t2 is between 0.72 and 0.74 inch. Even lower stress levels can be found for points beyond 0.78 which will make the part stronger but also heavier. If the objective of this process were to increase the safety factor these points would definitely be considered.

By looking at the slope of the stress on Figure 50 when the value oh h3 is 2.2, it may be seen that larger values of h3 would yield better results. While this is partially true a value of h3 that is too high would define a wall thickness of 0.

In Figure 52, it is quite clear and simple to understand how the stress increases as the hole in the base increases.

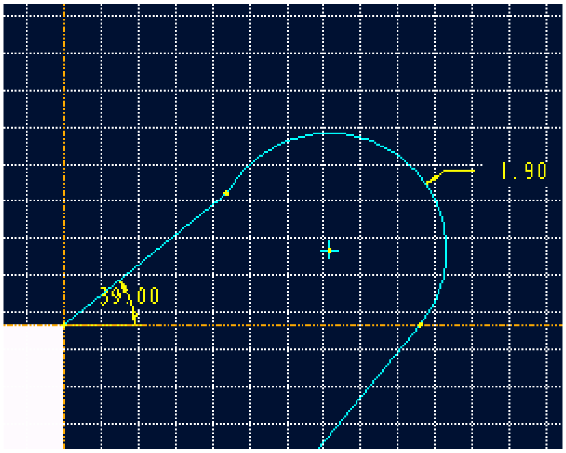


Figure 53 Varying dimension to α=39 deg.
(Design parameter 1)

In Figure 53, as the radius is incremented (design parameter 4), the volume of the link is lesser and the stress lowers. But when the design parameter is set to be 1.8, its geometry is not validated, and then the Pro/Mechanica extrapolates to a wrong output result.

When varying any design parameter, every value through its range must validate the "new" geometry. In other words every value assigned to the design parameters must not affect or change any other geometrical dimension of the link. The results are summarized below.

Reduction in Volume $= \dfrac{Vf - Vi}{Vi} * 100$; where:

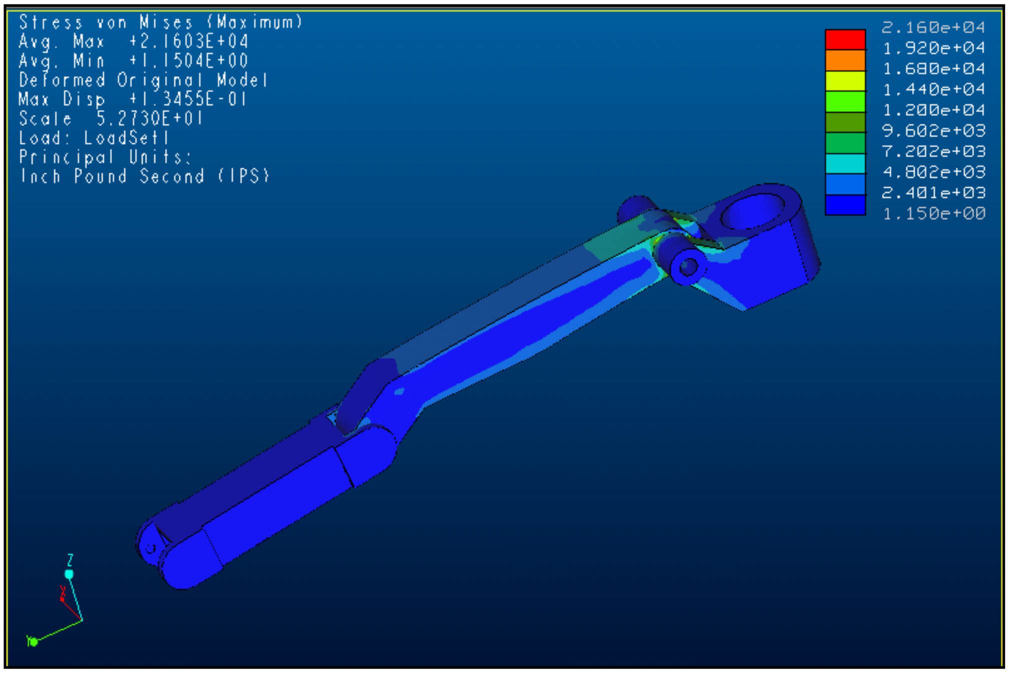
Vf = Final Volume (After Optimization). Vi = Initial Volume (Before Optimization).

As it is known: $\rho = \dfrac{m}{V}$ then $m = \rho \cdot V$

$$\rho_{\text{Steel}} = 0.00073234 \frac{lb * \sec^2}{in^4}$$

$$m = (1.2248 \cdot 10^3 - 1.1537^3)\ \text{in}^4 \cdot\cdot\ 0.00073234 \frac{lb * \sec^2}{in^4}$$

Design Parameter	Initial value	Optimal value	
“α”	39° deg.	37° degrees.	Reduction in volume of first link 5.88%
“t2”	0.78 in	0.73 in.	Reduction in volume of second link 0.36%
“h3”	2.0 in.	2.2 in.	Reduction in volume of third link 13.2%
“r4”	1.7 in.	1.6 in.	Reduction in volume of third link 13.2%
“d5”	5.75 in.	6.15 in.	Reduction in volume of first link 5.88%
Total weight	346.2 lb	326.1	Total reduction in weight: 5.82%= 20.1 lb!!
Max. Displacement	$1.346 \cdot 10^{-1}$ in.	$1.352 \cdot 10^{-1}$ in.	0.44%;



Stress von Mises (Maximum)
Avg. Max +2.1603E+04
Avg. Min +1.1504E+00
Deformed Original Model
Max Disp +1.3455E-01
Scale 5.2730E+01
Load: LoadSet1
Principal Units:
Inch Pound Second (IPS)

Figure 54 Schematic display of the von misses stress levels on the robot after minimizing its weight

Note:

1. The difference in magnitude of the Von Misses stress levels before (2.18x10 ^{4} Psi) and after the optimization (2.16x10 ^{4} Psi) called our attention because even though material is taken off from the robot the strength of it is increased at the same time. The critical location where the maximum stress is still noticed in the connection between the link 1 and the base.
2. The difference in the displacement is of the order of 0.44%; therefore, it is negligible. Hence, lighter robot does not deflect more.

8.1.5 Summary

The results found in this first case-study are significantly important because this analysis technique is able to reduce 5.82% of the weight of a specific industrial arm, which (depending on the material) can be as much as 20.1 lb. This will have noticeable effects on the control and payload of the robot. It is very important to keep in mind that some suggested changes are not feasible due to geometric implications (could create overlapping parts or hollowed spaces inside the arm) such errors wouldn't allow the FEA solver to reach a solution and get the deformations, stress levels, etc. without incurring in errors or illogical approximations. Even more precise results can be obtained if a local analysis is used that consists of varying the design parameters over a smaller range in order to gain accuracy in the optimized dimensions.

8.2 Design of a Robot from Scratch

When a robot of N degrees of freedom and M number of links is subject to an external payload, it develops a stress condition that depends both intrinsic (internal payloads and geometry) and extrinsic (materials and external payloads) parameters.[23]

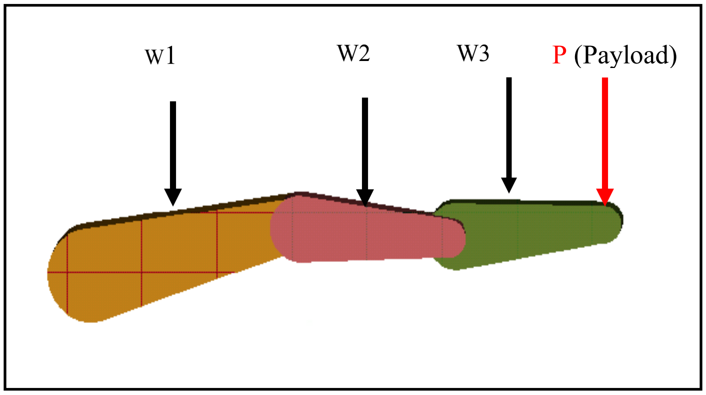


Figure 55 Robotic system of 3 degrees of freedom (dof)

The reader must remember that the strategy to approach to the solution of this problem. The two stages are graphically illustrated in Figure 10. Various design parameters (as listed below) are depicted in Figure 56.

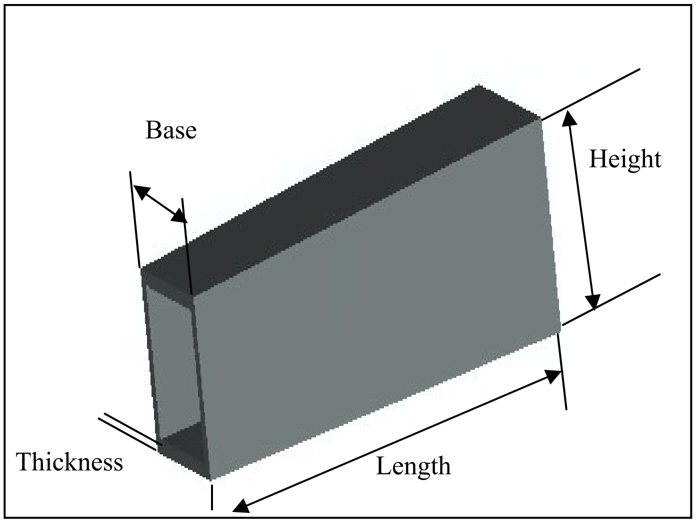


Figure 56 Pre-configured design of each link

x(1)=length of link 1

x(2)=base of link1

x(3)=thickness of link1

x(4)=length of link 1

x(5)=length of link 2

x(6)=base of link2

x(7)=thickness of link2

x(8)=length of link 2

x(9)=length of link 3

x(10)=base of link3

x(11)=thickness of link3

x(12)=length of link 3

8.2.1 Process Scheme of the Optimization Process (case study 2)

8.2.1.1 Matlab Optimization Routine

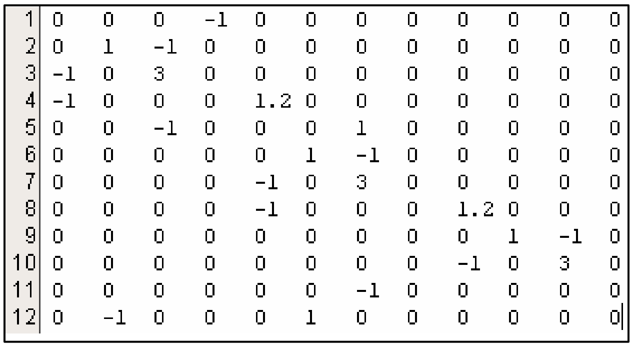
This process solves for the optimal values of the unknown vector X, which contains the optimal geometric measures of the robot. This vector X is constrained by the next parameters described before in chapter 4 (equation 4.11). These factors are next explained in detail.

$$\begin{aligned}&1)\,c(x) \leq 0\\&2)\,ceq\ (x) = 0\\&3)\,A * x \leq b\\&4)\,Aeq\ (x) = beq\\&5)\,lb \leq x \leq ub\end{aligned}$$

Due to the nature of our problem definition no formulation for c(x) and ceq(x) are needed to characterize. The program also has as inputs the external payload at the end effector's position (1000 lb), the yield stress for the material of the arms (20000 Psi, aluminum), and the safety factor (set to 5) to establish the system's reliability.

This very demanding nature of Multi-variable, non-linear optimization process, There is no theoretical guarantee that a solution determined is an absolute resolution for this type of an optimization problem.[24]

It is possible to converge to a local minimum that may be ok as there are infinitely many possible solutions and we are merely selection one that attempts to improve the overall system characteristic by reducing the weight.



1	0	0	0	-1	0	0	0	0	0	0	0	0
2	0	1	-1	0	0	0	0	0	0	0	0	0
3	-1	0	3	0	0	0	0	0	0	0	0	0
4	-1	0	0	0	1.2	0	0	0	0	0	0	0
5	0	0	-1	0	0	0	1	0	0	0	0	0
6	0	0	0	0	0	1	-1	0	0	0	0	0
7	0	0	0	0	-1	0	3	0	0	0	0	0
8	0	0	0	0	-1	0	0	0	1.2	0	0	0
9	0	0	0	0	0	0	0	0	0	1	-1	0
10	0	0	0	0	0	0	0	0	-1	0	3	0
11	0	0	0	0	0	0	-1	0	0	0	0	0
12	0	-1	0	0	0	1	0	0	0	0	0	0

Figure 57 Matrix A(that defines the linear-inequality terms)

The vector b (dimensionality 12x1) is defined as filled of zeros to enforce that all the values for the unknown vector x will be greater than zero; it means that geometric variables are always positives.

- ✓ i.e the first row we have : (-1)x(4)<0, then x(4)>0

- this was done to enforce the thickness definition for the first arm to be greater than zero.

- ✓ i.e the second row we have : (1)x(2)-(1)x(3)<0, then –x(2)+x(3)>0, then x(3)>x(2)

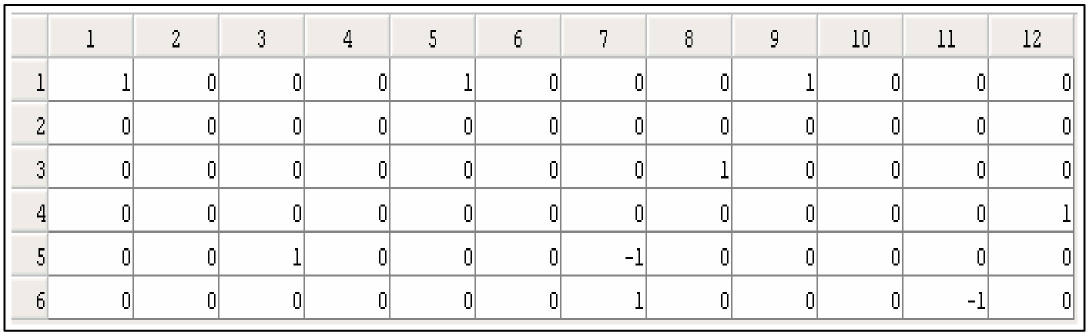
-This definition assures that the height of the first arm is thicker than its thickness

- ✓ i.e the third row we have : (-1)x(1)+(3)x(3)<0, then x(1)>x(3) to finally make x(1)>3x(3)

- this definition is to make sure that the length of the first arm is greater that its height.

✓ i.e the fourth row we have : (-1)x(1)+1.2x(5)<0, then x(1)-1.2x(5)>0 and finally x(1)>1.2x(5)

- The definition from before highlight that the link-length for the first arm is at least one point two times larger than the subsequent link



	1	2	3	4	5	6	7	8	9	10	11	12
1	1	0	0	0	1	0	0	0	1	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	1	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	1
5	0	0	1	0	0	0	-1	0	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0	-1	0

Figure 58 Matrix Aeq (that defines the linear-equality terms)

The vector beq (dimensionality 6x1) is defined by the maximum reachable distance that the robot (end effector's position) must accomplish and by the thickness of the three every arms among others.

✓ i.e the first row we have : x(1)+x(5)+x(9)= critical distance (See Figure 60)

- the link-length must not exceed the maximum reachable distance that the end-effector's position achieves.

✓ i.e the third row we have : thickness of the first arm=some value

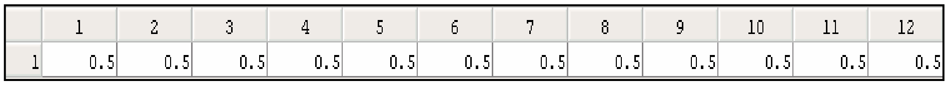
-we pre-define this thickness to be constant

✓ i.e the fifth row we have defined : x(4)-x(8)<0, then x(4)<x(8)

- The algorithm's calculation tend to equate these thicknesses

✓ i.e the sixth row we have defined : x(8)-x(12)<0, then x(8)<x(12)

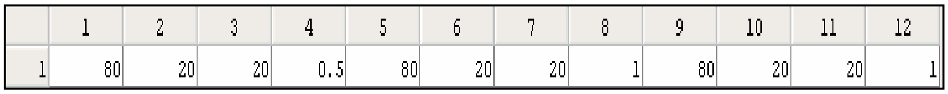
-The algorithm's calculation tend to equate these thicknesses



	1	2	3	4	5	6	7	8	9	10	11	12
1	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5

Figure 59 Vector Lb (this constraint provides not to exceed the lower boundary)

In the latest definition it is confirmed that no geometrical measures will acquire negative values.



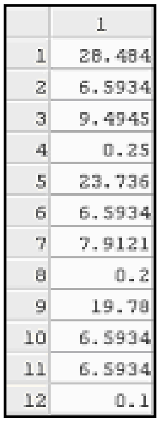
	1	2	3	4	5	6	7	8	9	10	11	12
1	80	20	20	0.5	80	20	20	1	80	20	20	1

Figure 60 Vector Up (this constraint provides not to exceed the upper boundary)

In the latest definition it is confirmed the Upper values that every variable involved in the optimization takes. This will prevent the Optimization function to search for values to far from reasonable values are.(i.e not going to infinity or blow up one variable and the others close to zero)

Output

The Optimized values of x are given as in Figure 61



	1
1	28.484
2	6.5934
3	9.4945
4	0.25
5	23.736
6	6.5934
7	7.9121
8	0.2
9	19.78
10	6.5934
11	6.5934
12	0.1

Figure 61 Vector X (optimized values, in inches)

8.2.1.2 Regeneration of links in Pro/Engineer:

Once the optimal values are determined, then the design is proceeded with the design regeneration of each component, as indicated below.

To have a pre-configured design with the cross-sectional properties and length defined by the design parameters, the final design is used to perform the stress analysis.

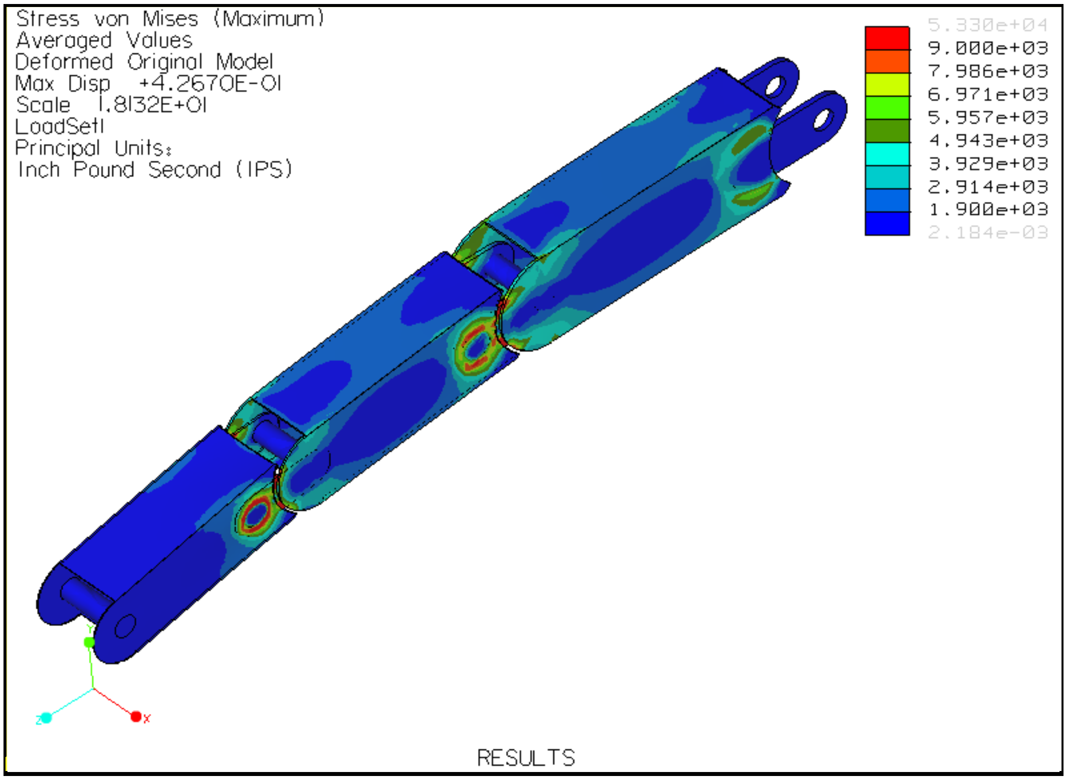
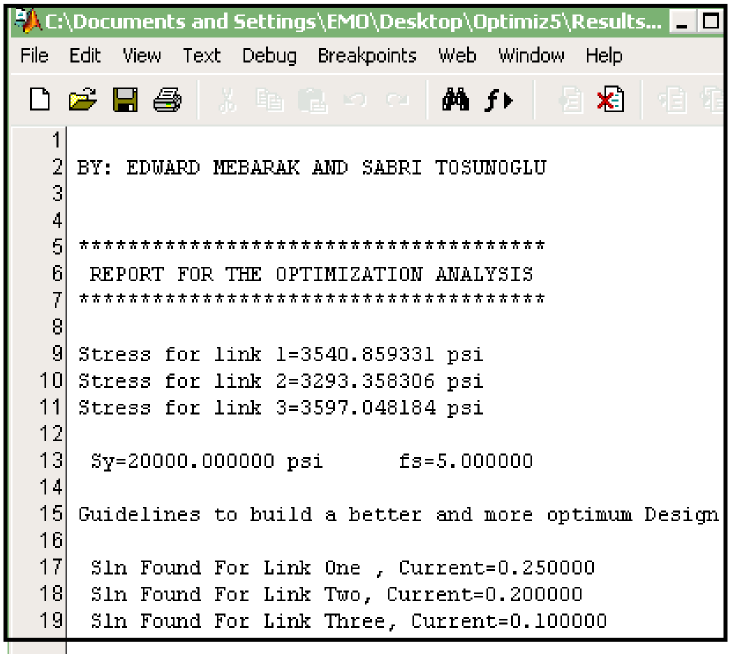


Figure 62 Stress distribution of the designed robot

By performing a FEA on the designed robot the program's results and accuracy are tested, in this way the user gains reliability in using the optimization routine that is proposed in this work to safely design robots.

A report file that contains the optimized values is written as an output so the designer can closely follow for the highest stress level met in every link, it also reminds the selected material to manufacture the robot, the thickness' are shown so the designer can minimize them in order to get a lighter structure without exceeding the yield stress.



```

BY: EDWARD MEBARAK AND SABRI TOSUNOGLU


****************************************
 REPORT FOR THE OPTIMIZATION ANALYSIS
****************************************

Stress for link 1=3540.859331 psi
Stress for link 2=3293.358306 psi
Stress for link 3=3597.048184 psi

 Sy=20000.000000 psi      fs=5.000000

Guidelines to build a better and more optimum Design

 Sln Found For Link One , Current=0.250000
 Sln Found For Link Two, Current=0.200000
 Sln Found For Link Three, Current=0.100000
```

Figure 63 guidelines and design recommendations (output from the optimization routine)

8.2.2 Summary

It is realized that the stress levels throughout the arms yielded by Pro/Mechanica (See Figure 60) coincide with the ones calculated by Matlab (See Figure 61). However, the user should be aware that there might be higher stresses due to stress concentration resulting from the joints, which is not the focus in this work [20].

9. CONCLUSION

In this thesis we have addressed the design of optimal robots using professional software packages. Optimal robots, as defined in this work, are those that are designed for minimum weight, which still withstand the highest levels of allowable stresses while carrying design payloads, have an efficient performance index and reduce the overall manufacturing and operational costs of a robot. Two different approaches have been used depending on the nature of the problem.

The first approach is used when we have pre-designed robots (robot's geometry, payload conditions, etc. are specified *apriori*). In this case the optimization is carried out through a selection of parameters that are improved in order to make the robot's weight lighter and in some cases end up with a stronger robot design (which also indicates that a lighter robot can actually carry a higher payload). In this work, a commercial robot was analyzed as a case study; the optimization results provided a lighter geometry and a reduction of 5.82% in weight was achieved.

The second approach is used when the objective is to design a robot from scratch. In this approach, we assume that the payload conditions, number of links in the robotic manipulator to be designed and some other external conditions are defined beforehand. This work was made possible by implementing an automated interaction between the Matlab and Pro/Engineer software packages. We have carried out the optimization process within Matlab, and the optimal design results were automatically shipped to Pro/Engineer to regenerate the 3D graphical representation of the final robot design. This design process can easily be applied to any mechanical component in which its geometry is driven by a set of design parameters that are subject to optimize a certain design criterion. Therefore, the integration of two software packages has far more application areas than just robotics.

This study demonstrated the automation of optimal design process in terms of a 3-degree-of-freedom industrial robot. However, in the future the capabilities of the developed process can be further extended to include:

a) More degrees of freedom, sliders, pivots that rotate in different planes or a combination of them; therefore, covering more possible robot configurations.

b) Increase the number of robotic manipulators (arms) in the case of multi-robot applications.

c) Develop the software integration at a more formal level; for instance, the output from Matlab can be written directly to Excel files so that Pro/Engineer can read them as instances in a family table feature.

d) Seamless integration of the mentioned software packages can lead the integration to be transparent to the average user. Hence, such an interface may lead to the development of a commercial software package to integrate Matlab and Pro/Engineer.

REFERENCES

[1] Definition of Robots

Webster's Dictionary, Merriam-Webster, Inc., July 1994.

[2] History and development of Robotics,

http://www.robotikitsdirect.com/html_files/history.html, Carson, CA, 2003.

[3] Rajeev S., "Optimizing Robot Motion Strategies for Assemblies with Stochastic Models of the Assembly," *IEEE Trans. on Robotics and Automation*, Vol. 12, pp.160-163, April 1991.

[4] Hert S., and Lumelsky V., "Motion Planning in R3 for Multiple Tethered Robots," *IEEE Trans. on Robotics and Automation*, Vol. 15, pp 623-629, August 1999.

[5] Optimization in Pro/Mechanica: http://pergatory.mit.edu/perg/resources/Design%20Optimization%20using%20Pro/Mechanica.htm, MIT, Massachusetts, MI, 2001.

[6] Leger C., "Performance Characterization of an Automated System for Robot Configuration Synthesis," *Proceedings of SPIE*, Vol. 4196, November 2000.

[7] Rao, S. S., "Optimization Theory and Applications," John Wiley & Sons, New York, pg 5-12, 156-220, 660-665, 1979.

[8] Stefani G., and Zezza P., "Optimality Conditions for a Constrained Control Problem", SIAM Journal on Control and Optimization, Volume 34, Number 2, pp. 635-659, 1996.

[9] Janos D. Pinter, "Global Optimization in Action: Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications," Kluwer Academic Publishers, 1995.

[10] Vasiliev V., "Optimal Design, Theory and Applications to Materials and Structures," pg. 31-47, 61-67, Technomic Publishing Co., Lancaster, PA, 1999.

[11] Goldfarb, D., and Polyak, R., "A modified barrier-augmented Lagrangian method for constrained minimization," *Comp. Optimization & Applications*, pp. 55-74, vol. 14, 1999.

[12] Method of Approach for Optimization Used in Matlab, http://www.wam.umd.edu/~petersd/660/fmincon_ex.html, University of Maryland, Maryland, MD, 2003.

[13] Capabilities of the Abaqus Package, general-purpose finite element analysis program http://www.cc.ic.ac.uk/services/asg/software/ansys.htm, London, UK, 2003.

[14] Potential of the Algor Software Package, http://www.algor.com/products/detail.asp?id=218, ALGOR, Inc, Pittsburgh, PA, 2003.

[15] Beer F., and E. R. Johnston, McGraw-Hill Ryerson, 2nd Edition, 1992.

[16] Jaydeep R., and Whitcomb L. "Structural Design Optimization and comparative Analysis of a New High –Performance Robot Arm Via Finite Element Analysis," *IEEE International Conference on Robotics and Automation*, Albuquerque, New Mexico, Vol. 1, pp. 2190-2197, April 1997.

[17] McCarthy M., "Mechanism Synthesis Theory and the Design of Robots," *IEEE International Conference on Robotics and Automation*, CA, April 24-28, 2000.

[18] Leger C., Automated Synthesis and Optimization of Robot Configurations: An Evolutionary Approach, PHD Dissertation, Carnegie Mellon University. Pittsburgh, PA, 2001.

[19] McCarthy M., "Mechanism Synthesis Theory and the Design of Robots," University of California, *IEEE International Conference on Robotics and Automation*, San Francisco, CA, April 24-28, 2000.

[20] Murli A. and Toraldo G., "Computational Issues in High Performance Software for Nonlinear Optimization," *Journal on Computational Optimization and Applications*, Volume 7, No. 1, January 1997.

[21] Finite Element Analysis and Optimization, http://www3.sympatico.ca/peter_budgell/FEA_intro.html, ANSYS, Inc, Burlington, Ontario, Canada, 2003.

[22] Mebarak E. & Tosunoglu S., “Finite Element Analysis to Optimize Robot Design: A Case Study on an Industrial Robot,” *Florida Conference on Recent Advances in Robotics.* Tallahassee, Fl, May 10, 2001.

[23] Mebarak E. & Tosunoglu S., “Optimal Design of a Robotic System via the Automated Integration of MatLab and Pro/Engineer Software Tools,” *Conference on* Recent Advances in Robotics and Robot Showcase (FCRAR 2002), Miami, Fl, May 23, 2002.

[24] Mebarak E. & Tosunoglu S., “On The Development of an Automated Design Interface for Optimal Design of Robotic Systems,” *World Automation Congress* (WAC2002), Orlando, Fl, June 9, 2002.

APPENDICES

Appendix 1

Code 1: Program that considers and evaluates the end-effector’s position.

```
K = MENU('choose','circle','square','Input set of points')
global a1 a2 a3 r incx incy x0 y0
np=48;%input('number of points to be evaluated=')
% 50,30,3,10,30,30,135 for ome=0
% 15,13.5,7.5,8,10,10,135 for ome=0
uiwait(msgbox('Input link lenghts a1,a2,a3','Have Ready the Inputs','warn'));
a1=15.5;%input('First link length a1=')
a2=13.5;%input(' Second link length a2=')
a3=6;%input('First link third length a3=')
%input('angle phi_h=')130
check= MENU('chose','Omega is not Zero','Omega is Zero')
t=0;
% determining the accuracy  and convergence og phi_1 and phi_2
%==========================================
if (np<=20&np>0)%checking Continuity for Phi_1 and Phi_2
   conv1=.7;
   conv2=.7;
elseif (np<=60&np>20)
   conv1=.4;
   conv2=.4;
elseif (np<100&np>60)
   conv1=.3;
   conv2=.3;
end
%============================================
if check==1
   ome=input('Omega for the end effector position=')
   phi_hFIX=135;
   uiwait(msgbox('Omega is not Zero','Phi_h = Will Vary','warn'))
else
   ome=0
   check=2;
   uiwait(msgbox('Omega is Zero','Phi_h=Constant','warn'));
   phi_hFIX=135;%input('In Degrees phi_h='),135;
   %320;%input('In Degrees phi_h='),30
end
```

```
t=0;
Tetha=0;

x1=10;
y1=5;
x2=20;
y2=5;
x3=20;
y3=15;
x4=10;
y4=15;
incx=(x3-x1)/(np/4);
incy=(y3-y1)/(np/4);


for i=1:np+1
   switch K
   case 1,
      y0=10;%input('Center in Y of the circle='),30
      x0=10;%input('Center in Y of the circle='),30
      r=8;

      ax=.2;
      ay=.2;

      [Vx,Vy,x_hc,y_hc,phi_hc]=circle(i,phi_hFIX,check,ome,t,Tetha);
      Tetha=Tetha+360/np;
      t=t+0.433;
      %Calling function angles
      x_h(i)=x_hc;
      y_h(i)=y_hc;
      phi_h(i)=phi_hc;
      [phi_1f,phi_2f] = angles(i,phi_h(i),y_h(i),x_h(i));
      phi_1(i,1)=phi_1f;
      phi_2(i,1)=phi_2f;
   case 2,

[x1,y1,x2,y2,x3,y3,x4,y4,ax,ay,Vx,Vy,x_hc,y_hc,phi_hc]=square(x1,y1,x2,y2,x3,y3,x4,y
4,i,phi_hFIX,check,ome,t,Tetha,np);

      Tetha=Tetha+360/np;
      t=t+0.433;
      x_h(i)=x_hc;
      y_h(i)=y_hc;
      phi_h(i)=phi_hc;
```

```
    [phi_1f,phi_2f] = angles (i,phi_h(i),y_h(i),x_h(i));
    phi_1(i,1)=phi_1f;
    phi_2(i,1)=phi_2f;

  case 3,
    setpoints=load('setpoints.txt');
    x_h=setpoints(:,1);
    y_h=setpoints(:,2);

    if (check==1)
      phi_hc=ome*t*57.29;
    else
      phi_hc=phi_hFIX;
    end
    Tetha=Tetha+360/np;
    t=t+0.433;
    Vx=.5
    Vy=.5
    ax=.5
    ay=.5
    phi_h(i)=phi_hc
    [phi_1f,phi_2f] =angles(i,phi_h(i),y_h(i),x_h(i))
    phi_1(i,1)=phi_1f;
    phi_2(i,1)=phi_2f;

  end
  %evaluating the inputs for a feasible solution

  %To make possible the Velocity analysis, we should assign values a scalar to phi_1
and phi_2

  syms X a11 a21 a31 ph_1 ph_2 ph_h ph_3 ;% Establishing all variables (new
variables) as symbols
  X=a11.*cos(ph_1)+a21.*cos(ph_1+ph_2)+a31.*cos(ph_1+ph_2+ph_3);
  Y=a11.*sin(ph_1)+a21.*sin(ph_1+ph_2)+a31.*sin(ph_1+ph_2+ph_3);
  %getting the Matrix "G"
  %======================================
  Gx1=diff(X,ph_1);%Defifning G(1,:)
  Gx2=diff(X,ph_2);
  Gx3=diff(X,ph_3);
  Gy1=diff(Y,ph_1);%Defifning G(2,:)
  Gy2=diff(Y,ph_2);
  Gy3=diff(Y,ph_3);
  %======================================
  %getting the Matrix "h"
  %=================================
  hx11=diff(Gx1,ph_1); %Defining hx(1,:)
  hx12=diff(Gx1,ph_2);
  hx13=diff(Gx1,ph_3);
  hx21=diff(Gx2,ph_1); %Defining hx(2,:)
  hx22=diff(Gx2,ph_2);
  hx23=diff(Gx2,ph_3);
  hx31=diff(Gx3,ph_1); %Defining hx(3,:)
  hx32=diff(Gx3,ph_2);
  hx33=diff(Gx3,ph_3);
  hy11=diff(Gy1,ph_1);%Defining hy(1,:)
  hy12=diff(Gy1,ph_2);
  hy13=diff(Gy1,ph_3);
  hy21=diff(Gy2,ph_1);%Defining hy(2,:)
  hy22=diff(Gy2,ph_2);
  hy23=diff(Gy2,ph_3);
  hy31=diff(Gy3,ph_1);%Defining hy(3,:)
  hy32=diff(Gy3,ph_2);
  hy33=diff(Gy3,ph_3);

  h11=1;
  h12=1;
  h13=1;
  %===============================================
  a11=a1;
  a21=a2;
  a31=a3;
  ph_1=phi_1(i,1);
  ph_2=phi_2(i,1);
  ph_h=phi_h(i);
  if check==1
    ph_3=(ph_h)-phi_1(i,1)-phi_2(i);
  else
    ph_3=(phi_hFIX/57.29)-phi_1(i,1)-phi_2(i);
  end
  G=[Gx1 Gx2 Gx3;Gy1 Gy2 Gy3;1 1 1]; %Grouping the derivatives in G
  eval(G);
  hx=[hx11 hx12 hx13;hx21 hx22 hx23;hx31 hx32 hx33];%Grouping the derivatives in
hx
  hy=[hy11 hy12 hy13;hy21 hy22 hy23;hy31 hy32 hy33]; %Grouping the derivatives in
hy

  h=[1 0 0;0 1 0;0 0 1];
  eval(hx);
  eval(hy);
  V=[Vx;Vy;ome];
  W=(eval(inv(G)*V));
```

```
   a(:,i)=W(:,:);
   W1(i,1)=a(1,i);
   W2(i,1)=a(2,i);
   W3(i,1)=a(3,i);
   WW=[W1(i);W2(i);W3(i)]; %Setting the angular vel. in a vector WW=[W1;W2;W3]

   %Building the f(phi,ome)  same: Wt* h *W matrix
   %==============================================
   f1=WW'*hx*WW;
   f2=WW'*hy*WW;
   f3=WW'*h*WW;

   f=[f1;f2;f3];
   eval(f);
   A=[ax;ay;.1];
   alpha=eval(inv(G))*A-eval(f);
   b(:,i)=alpha(:,:);
   alpha1(i,1)=b(1,i);
   alpha2(i,1)=b(2,i);
   alpha3(i,1)=b(3,i);

   ===============================================
   if i==1
      sw(1)=1;

   elseif ((abs(phi_1(i)-phi_1(i-
1))/abs(phi_1(i))<conv1)&((imag(phi_1(i)))==0)&(abs(phi_2(i)-phi_2(i-
1))/abs(phi_1(i))<conv2)&((imag(phi_1(i)))==0))
      sw(i)=1;
   else sw(i)=0;
  end

end
%==============================================================
=========================
if all(sw)==1
   fid=fopen('Dimensions.txt','a');
   fprintf (fid,'a1=%f\t a2=%f\t a3=%f\t phi_h=%f\n',a1,a2,a3,phi_h);
   (msgbox('Succesfully Completed ,Solution Found','Success','warn'))
else
   errordlg('Solution not Found,Try other inputs, Imaginary numbers in
Solution','Fail','warn')
end

phi_1;
phi_2;
```

```
figure
for w=1:np+1
   hold on;
   g=[0;a1.*cos(phi_1(w))];   % Drawing First Arm
   h=[0;a1.*sin(phi_1(w))];
   line(g,h,'lineWidth',.5,'Color','r');
   g=[a1.*cos(phi_1(w));a2.*cos(phi_2(w))+a1.*cos(phi_1(w))];     %   Drawing   Second
Arm
   h=[a1.*sin(phi_1(w));a2.*sin(phi_2(w))+a1.*sin(phi_1(w))];
   line(g,h,'lineWidth',.5,'Color','g');

g=[a2.*cos(phi_2(w))+a1.*cos(phi_1(w));a3.*cos(phi_h(w)/57.29)+a2.*cos(phi_2(w))+a
1.*cos(phi_1(w))];  % Drawing Third Arm

h=[a2.*sin(phi_2(w))+a1.*sin(phi_1(w));a3.*sin(phi_h(w)/57.29)+a2.*sin(phi_2(w))+a1.
*sin(phi_1(w))];
   line(g,h,'lineWidth',.5,'Color','b');
   px=a3.*cos(phi_h(w)/57.29)+a2.*cos(phi_2(w))+a1.*cos(phi_1(w));
   py=a3.*sin(phi_h(w)/57.29)+a2.*sin(phi_2(w))+a1.*sin(phi_1(w));
   plot(px,py,'ro'); %Drawing the circle path
   pause(0.01);%make a pause before the next figure
end
title('Figure 1: Robot Motion')
grid on;
hold off;
pause(2)
figure
w=(1:1:np+1);
plot(w,phi_1,w,phi_2,'-');
title('Figure 2: Phi_1 and Phi_2 Behavior (In Radians)')
legend('phi_1','phi_2',1)
grid on;
x_h=x_h';
y_h=y_h';
pause(2)
figure
plot(w,W1,w,W2,w,W3);
title('Figure 3: W1 and W2 Behavior (In Radians/ Seg )')
legend('W1','W2','W3',1)
grid on;
pause(2)
figure
plot(w,alpha1,w,alpha2,w,alpha3);
title('Figure 4: alpha1 and alpha2 and alpha3 Behavior (In Radians/ Seg^2 )')
legend('ALPHA1','ALPHA2','ALPHA3',1)
```

```
grid on;
end
%fid=fopen('setpoints.xls','a')
%fprintf (fid,'%d\t%d\n',x_h(i),y_h(i));
%fid=fclose('all')
(function)= angles
function [phi_1f,phi_2f] = angles(i,phi_h,y_h,x_h)
global a1 a2 a3
A_1(i)=x_h-a3*cos(phi_h/57.29);
A_2(i)=y_h-a3*sin(phi_h/57.29);
U(i)=2*a1.*A_2(i);
V(i)=2*a1.*A_1(i);
W(i)=(a2.^2)-(a1^2)-(A_1(i).^2)-(A_2(i).^2);
X_1(i)=(-U(i)-(sqrt(U(i).^2+V(i).^2-W(i).^2)))/(W(i)-V(i));
phi_1f=2*atan(X_1(i));
phi_2f=atan((y_h-a3*sin(phi_h/57.29)-a1*sin(phi_1f))/(x_h-a3*cos(phi_h/57.29)-
a1*cos(phi_1f)));
========================
(function)=circle
function[Vx,Vy,x_hc,y_hc,phi_hc]=circle(i,phi_hFIX,check,ome,t,Tetha,x0,y0)
global r x0 y0

Vh=1;
Vy=Vh*cos(Tetha/57.29);
Vx= Vh*sin(Tetha/57.29);
v_c=r*cos(Tetha/57.29);
if Tetha<=180
   y_hc=y0+sqrt(r.^2-(v_c^2));
else
   y_hc=y0-sqrt(r.^2-(v_c^2));
end
if (check==1)
   phi_hc=ome*t*57.29;
else
   phi_hc=phi_hFIX;
end
x_hc=x0-r*cos((Tetha./57.29));
+++========================================
(function)=squared
function
[x1,y1,x2,y2,x3,y3,x4,y4,ax,ay,Vx,Vy,x_hc,y_hc,phi_hc]=square(x1,y1,x2,y2,x3,y3,x4,y
4,i,phi_hFIX,check,ome,t,Tetha,np,incx,incy)
global  incx incy
if (i<=np/4)
   x_hc=x1;
   y_hc=y1;
   Vx=1;
   Vy=0;
   ax=.2;
   ay=0;
   x1=x1+incx;
elseif ((i>np/4)&(i<=np/2))
   x_hc=x2;
   y_hc=y1;
   Vx=0;
   Vy=1;
   ax=0;
   ay=0.2;
   y1=y1+incy;
elseif((i>np/2)&(i<=np*3/4))

   x_hc=x3;
   y_hc=y3;
   Vx=-1;
   Vy=0;
   ax=-.2;
   ay=0;
   x3=x3-incx;
elseif((i>np*3/4)&(i<=np+1))

   x_hc=x3;
   y_hc=y3;
   Vx=0;
   Vy=-1;
   ax=0;
   ay=-0.2;
   y3=y3-incy;
end

if (check==1)
   phi_hc=ome*t*57.29;
else
   phi_hc=phi_hFIX;
end
```

Appendix 2

Code 2: Program that minimizes weight and stress on the robot's structure.

```
close all;
clc;
clear all;
sw=0;
x0=[26 6.5 7 .1 25 5 6 1.5 2 5 7 .1];%starting values for optimization
b=[0;0;0;0;0;0;0;0;0;0;0;0];
K= MENU('chose','Predefined Thicknesses','same base and predefined thickness','Look for Thicknesses');
thick=load('thick.txt');
d=thick(1,:);
t2=thick(3,:);
t3=thick(4,:);
A=load('c.txt');% defining the inequality constraints
while sw==0
   Aeq=[1 0 0 0 1 0 0 0 1 0 0 0]; % defining the equality constraints
   beq=[d] ;
   switch K
   case 1,
      Aeq(2,:)=[0 0 0 1 0 0 0 0 0 0 0 0];
      Aeq(3,:)=[0 0 0 0 0 0 0 1 0 0 0 0];
      Aeq(4,:)=[0 0 0 0 0 0 0 0 0 0 0 1];
      beq(2,1)=[t1] ;
      beq(3,1)=[t2] ;
      beq(4,1)=[t3] ;

   case 2,
      beq(3,1)=[t2] ;
      Aeq(3,:)=[0 0 0 0 0 0 0 1 0 0 0 0];
      Aeq(4,:)=[0 0 0 0 0 0 0 0 0 0 0 1];
      Aeq(5,:)=[0 1 0 0 0 -1 0 0 0 0 0 0];
      Aeq(6,:)=[0 0 0 0 0 1 0 0 0 -1 0 0];
      beq(4,1)=[t3] ;
      beq(5,1)=0;
      beq(6,1)=0;
   end

   options = optimset('maxfunevals',10000);
   lb=[0.5 0.5 0.01 0.01 0.5 0.5 0.5 0.1 0.01 2 0.5 0.01];
   ub=[80 30 30 2 80 30 30 2 80 30 30 2];
   ub(1,4)=thick(2,:);
```

```
  [x,feval,g,H]=fmincon('fc1',x0,A,b,Aeq,beq,lb,ub);
  feval;
  x=x';
  %Desining  the arm1
  d(1)=x(1);
  d(2)=x(2)/2;
  d(3)=x(3)/2;
  d(4)=x(3)/2-x(4);
  d(5)=x(4);
  d(6)=x(4);
  d(7)=x(4);
  dif=0;
  d(8)=dif;
  d(9)=dif;
  if d(8)==0
     m=1;
  else
     m=0;
  end
  if d(7)>=d(1);
     A(12,6)=1.1;%enforcing base of link 1 to be greater than base link 2.
     sw=0;
     uiwait(msgbox('enforcing base of link 1 to be greater than base link 2','base1 = base
2','warn'))
  end
  oldDir = pwd;
  cd('c:\working3');
  %writing in a txt file the design for arm1
  fid=fopen('arm1.txt','w');
  for i=1:9
     fprintf (fid,'d%d=%f\n',i,d(i));
  end
  fprintf (fid,'m=%f\n',m);
  %Desining  the arm2
  d(1)=x(5);
  d(2)=x(6)/2;
  d(3)=x(7)/2;
  d(4)=x(7)/2-x(8);
  d(5)=x(8);
  d(6)=x(8);
  d(7)=x(8);
  dif=x(4)-x(8)
  d(8)=dif;
  d(9)=dif;

  if d(8)==0
```

```
      m=1;
   else
      m=0;
   end
   %writing in a txt file the design for arm2
   fid=fopen('arm2.txt','w');
    for i=1:9
      fprintf (fid,'d%d=%f\n',i,d(i));
   end
   fprintf (fid,'m=%f\n',m);

   %Desining  the arm3
   d(1)=x(9);
   d(2)=x(10)/2;
   d(3)=x(11)/2;
   d(4)=x(11)/2-x(12);
   d(5)=x(12);%-x(12);
   d(6)=x(12);%-x(12)just in case..!
   d(7)=x(12);
   dif=x(8)-x(12)
   d(8)=dif;
   d(9)=dif;
   if d(8)==0
      m=1;
   else
      m=0;
   end

   fid=fopen('arm3.txt','w');
   %writing in a txt file the design for arm3
   for i=1:9
      fprintf (fid,'d%d=%f\n',i,d(i));
    end
   fprintf (fid,'m=%f\n',m);
   fid=fclose('all');
   sw=1;
end

%the b(15), should be a ratio relating the load and the cross areas.
% do if statements to define b(15) depending the load value.
```

Function

```
function [f] = fc(x)
%Input the Payload thath the end-effector position must keep-up
%weight arm3
input=load('input.txt');
```

```
P=input(1,:);
Sy=input(2,:);
fs=input(3,:);
B=x(6)*x(7);
P2=x(2)+x(3);
P1=x(6)+x(7);
b=(x(6)-2*x(4))*(x(7)-2*x(4));
p2=x(2)+x(3)-4*x(4);
p1=x(6)+x(7)-4*x(4);
w1=(1/3*x(1)*B*(1+(P2/P1)+(P2/P1)^2)-(1/3*x(1)*b*(1+(p2/p1)+(p2/p1)^2)))*.28;%
total weight for arm1
%bending stress for arm3
%-----------------------------------------------------------------------
%weight arm1
B=(x(10)-2*x(12))*(x(11)-2*x(12));
P2=x(10)+x(11);
P1=x(10)+x(11)-4*x(12);
b=(x(10)-4*x(12))*(x(11)-4*x(12));
p2=x(10)+x(11)-4*x(12);
p1=x(10)+x(11)-8*x(12);
w3=(1/3*x(9)*B*(1+(P2/P1)+(P2/P1)^2)-(1/3*x(9)*b*(1+(p2/p1)+(p2/p1)^2)))*.28;%
total weight for arm3
%--------------------------------------------------------------------------------
%weight arm2
B=x(10)*x(11);
P2=x(6)+x(7);
P1=x(10)+x(11);
b=(x(10)-2*x(8))*(x(11)-2*x(8));
p2=x(6)+x(7)-4*x(8);
p1=x(10)+x(11)-4*x(8);
w2=(1/3*x(5)*B*(1+(P2/P1)+(P2/P1)^2)-(1/3*x(5)*b*(1+(p2/p1)+(p2/p1)^2)))*.28;%
total weight for arm2
%--------------------------------------------------------------------------
I1=((x(2)*x(3)^3)-(x(2)-2*x(4))*(x(3)-2*x(4))^3)/12;
sw1=(w3*(x(1)+x(5)+x(9)/2)+w2*(x(1)+x(5)/2)+w1*x(1)/2)*x(3)/2/I1;%flexion
moment due to the weight itself
pw1=((P*(x(1)+x(5)+x(9)))*x(3)/2)/I1;
%flexion moment due to the Payload
s1=(sw1+pw1);
%----------------------------------------------------------------------------------
%bending stress for arm2
I2=((x(6)*x(7)^3)-(x(6)-2*x(8))*(x(7)-2*x(8))^3)/12;
sw2=(w3*(x(5)+x(9)/2)+w2*x(5)/2)*x(7)/2/I2;%stress due to the weight itself
pw2=(P*(x(9)+x(5)))*x(7)/2/I2;
s2=(sw2+pw2);
%----------------------------------------------------------------------------------
```

```
%bending stress for arm1
I3=((x(10)*x(11)^3)-(x(10)-2*x(12))*(x(11)-2*x(12))^3)/12;
sw3=w3*x(9)/2*x(11)/2/I3;%stress due to the weight itself
pw3=P*x(9)*x(11)/2/I3;
s3=(sw3+pw3);
[f]=s1+s2+s3;
fid=fopen('Results.txt','w');
fprintf (fid,'\n');
fprintf (fid,'BY: EDWARD MEBARAK AND SABRI TOSUNOGLU \n');
fprintf (fid,'\n');
fprintf (fid,'\n');
fprintf (fid,'************************************** \n');
fprintf (fid,' REPORT FOR THE OPTIMIZATION ANALYSIS \n');
fprintf (fid,'************************************** \n');
fprintf (fid,'\n');
fprintf (fid,'Stress for link 1=%f psi\t  \n',s1);
fprintf (fid,'Stress for link 2=%f psi\t  \n',s2);
fprintf (fid,'Stress for link 3=%f psi\t  \n',s3);
fprintf (fid,'\n');
fprintf (fid,'weight for link 1=%f lb\t  \n',w1);
fprintf (fid,'weight for link 2=%f lb\t  \n',w2);
fprintf (fid,'weight for link 3=%f lb\t  \n',w3);
fprintf (fid,'\n');
fprintf (fid,' Sy=%f psi\t  fs=%f\n',Sy,fs);
fprintf (fid,'\n');
fprintf (fid,'Guidelines to build a better and more optimum Design');
fprintf (fid,'\n');
if (s1>=Sy/fs)

   fprintf (fid,'\n you must increase thickness 1 or reduce fs., Current=%f ',x(4)');
else
   fprintf (fid,'\n Sln Found For Link One , Current=%f ',x(4));
end

if (s2>=Sy/fs)
   fid=fopen('Results.txt','a');
   fprintf (fid,'\n you must increase thickness 2 or reduce fs., Current=%f ',x(8)');
else
   fprintf (fid,'\n Sln Found For Link Two, Current=%f ',x(8)');
end

if (s3>=Sy/fs)
   fid=fopen('Results.txt','a');
   fprintf (fid,'\n you must increase thickness 3 or reduce fs., Current=%f ',x(12)');
else
```

```
   fprintf (fid,'\n Sln Found For Link Three, Current=%f ',x(12)');
end

close all;
```

The whole information of the part is contained in this single script file that can be modified at any time and then the part is intended to be regenerated.

Appendix 3

Pro/Engineer Design Parameters

```
VERSION
REVNUM 4333
LISTING FOR PART G2

INPUT
 D1 NUMBER
 D2 NUMBER
 D3 NUMBER
 D4 NUMBER
 D5 NUMBER
 D6 NUMBER
 D7 NUMBER
 D8 NUMBER
 M NUMBER
END INPUT

RELATIONS
RADIO=0.3*D2
RME=0.3*D2
RMAY=0.5*D3

L=D1/4
TH=2*D5
RCUT=1.3*D3
ANG1=45
ANG2=45
RADI=0.4*D3
LSAL=D1-1.8*D3
RSAL=0.35*D3
ROUND2=D5/2

RMAY2=.5*D3
RMEN2=.3*D2
L2=1.8*D3
RR=0.6*D3
THICK=2*D5

END RELATIONS

ADD FEATURE (initial number 1)
```

```
INTERNAL FEATURE ID  1
TYPE = DATUM PLANE
NAME = RIGHT


  FEATURE IS IN LAYER(S) :
    01___PRT_ALL_DTM_PLN - OPERATION = SHOWN
    01___PRT_DEF_DTM_PLN - OPERATION = SHOWN

END ADD


ADD FEATURE (initial number 2)
INTERNAL FEATURE ID  3
TYPE = DATUM PLANE
NAME = TOP


  FEATURE IS IN LAYER(S) :
    01___PRT_ALL_DTM_PLN - OPERATION = SHOWN
    01___PRT_DEF_DTM_PLN - OPERATION = SHOWN

END ADD


ADD FEATURE (initial number 3)
INTERNAL FEATURE ID  5
TYPE = DATUM PLANE
NAME = FRONT


  FEATURE IS IN LAYER(S) :
    01___PRT_ALL_DTM_PLN - OPERATION = SHOWN
    01___PRT_DEF_DTM_PLN - OPERATION = SHOWN

END ADD


ADD FEATURE (initial number 4)
INTERNAL FEATURE ID  7
TYPE = COORDINATE SYSTEM
NAME = PRT_CSYS_DEF


  FEATURE IS IN LAYER(S) :
    05___PRT_ALL_DTM_CSYS - OPERATION = SHOWN
    05___PRT_DEF_DTM_CSYS - OPERATION = SHOWN

END ADD


ADD FEATURE (initial number 5)
INTERNAL FEATURE ID  39
PARENTS = 1(#1) 3(#2) 5(#3)

PROTRUSION: Blend, Parallel, Regular Sections

NO. ELEMENT NAME    INFO                                    STATUS
--- ------------ -----------                                ------
 1  Attributes   Straight                                     Defined
 2  Section      Sk. plane - Surface FRONT of feat #3 (DATUM PLANE) Defined
 3  Direction                                               Defined
 4  Depth        Blind                                        Defined

SECTION NAME = S2D0003


FEATURE'S DIMENSIONS:
d1 = 23.74
d3 = 3.96
d5 = .20
d2 = 3.30
d4 = 3.76
d7 = .20
d6 = .20
END ADD


ADD FEATURE (initial number 6)
INTERNAL FEATURE ID  90
PARENTS = 3(#2) 39(#5)

PROTRUSION: Extrude

NO. ELEMENT NAME    INFO                                    STATUS
--- ------------ -----------                                ------
 1  Attributes   One Side                                      Defined
 2  Section      Sk. plane - Surface of feat #5 (PROTRUSION)       Defined
 3  MaterialSide                                             Defined
 4  Direction                                               Defined
 5  Depth        Upto Surface                                    Defined
```

```
SECTION NAME = S2D0004
OPEN SECTION


END ADD


ADD FEATURE (initial number 7)
INTERNAL FEATURE ID  140
PARENTS = 3(#2) 5(#3) 39(#5)

CUT: Extrude

NO. ELEMENT NAME    INFO                                  STATUS
--- ------------ -----------                              ------
 1  Attributes   One Side                                  Defined
 2  Section      Sk. plane - Surface of feat #5 (PROTRUSION)       Defined
 3  MaterialSide                                           Defined
 4  Direction                                              Defined
 5  Depth        Through All                               Defined

SECTION NAME = S2D0001
OPEN SECTION


FEATURE'S DIMENSIONS:
ang1 = 45.00
ang2 = 45.00
rcut = 5.14R
END ADD


ADD FEATURE (initial number 8)
INTERNAL FEATURE ID  225
PARENTS = 90(#6) 1(#1) 3(#2)

PROTRUSION: Extrude

NO. ELEMENT NAME    INFO                                  STATUS
--- ------------ -----------                              ------
 1  Attributes   One Side                                  Defined
 2  Section      Sk. plane - Surface RIGHT of feat #1 (DATUM PLANE) Defined
 3  Direction                                              Defined
 4  Depth        Through All                               Defined

SECTION NAME = S2D0002
```

```
  FEATURE IS IN LAYER(S) :
   02___PRT_ALL_AXES - OPERATION = SHOWN

FEATURE'S DIMENSIONS:
radio = .99R
END ADD

IF M==1

   ADD FEATURE
   INTERNAL FEATURE ID  1228
   PARENTS = 5(#3) 140(#7) 3(#2) 39(#5)

PROTRUSION: Extrude

NO. ELEMENT NAME    INFO                                    STATUS
--- ------------ -----------                          ------
 1  Attributes   One Side                                 Defined
 2  Section      Sk. plane - Surface of feat #5 (PROTRUSION)        Defined
 3  Direction                                        Defined
 4  Depth        Blind, depth = 0.4                       Defined

   SECTION NAME = S2D0002

  FEATURE IS IN LAYER(S) :
   02___PRT_ALL_AXES - OPERATION = SHOWN

   FEATURE'S DIMENSIONS:
   rmay = 1.98R
   l = 5.93
   th = .40
   rme = .99R
   END ADD

ELSE

   ADD FEATURE (initial number 9)
   INTERNAL FEATURE ID  450
   PARENTS = 3(#2) 225(#8) 39(#5)

PROTRUSION: Extrude

NO. ELEMENT NAME    INFO                                    STATUS
--- ------------ -----------                          ------
 1  Attributes   One Side                                 Defined
 2  Section     Sk. plane - Surface of feat #5 (PROTRUSION)       Defined
 3  Direction                                        Defined
 4  Depth       Blind, depth = 0.05                            Defined

   SECTION NAME = S2D0002

  FEATURE IS IN LAYER(S) :
   02___PRT_ALL_AXES - OPERATION = SHOWN

   FEATURE'S DIMENSIONS:
   rsal = 1.38R
   lsal = 16.62
   d8 = .05
   END ADD


   ADD FEATURE (initial number 10)
   INTERNAL FEATURE ID  579
   PARENTS = 3(#2) 5(#3) 450(#9)

PROTRUSION: Extrude

NO. ELEMENT NAME    INFO                                STATUS
--- ------------ -----------                      ------
 1  Attributes   One Side                              Defined
 2  Section     Sk. plane - Surface of feat #9 (PROTRUSION)       Defined
 3  Direction                                        Defined
 4  Depth       Blind, depth = 0.4                            Defined

   SECTION NAME = S2D0001

  FEATURE IS IN LAYER(S) :
   02___PRT_ALL_AXES - OPERATION = SHOWN

   FEATURE'S DIMENSIONS:
   thick = .40
   rr = 2.37R
   rmay2 = 1.98R
   rmen2 = .99R
   l2 = 7.12
   END ADD


   ADD FEATURE (initial number 11)
   INTERNAL FEATURE ID  1421
   PARENTS = 39(#5) 450(#9) 579(#10)
```

```
ROUND: General

NO. ELEMENT NAME    INFO                              STATUS
--- ------------ -----------                         ------
 1  Round Type   Simple                                 Defined
 2  Attributes   Constant, Edge Chain                      Defined
 3  References                                       Defined
 4  Radius       Value = 0.1000                         Defined
 5  Round Extent                                       Optional
 6  Ambiguity    All pieces                             Defined
 7  Attach Type  Make Solid - Feature has solid geometry           Defined



   FEATURE'S DIMENSIONS:
   round2 = .10R
   END ADD
END IF


ADD FEATURE (initial number 12)
INTERNAL FEATURE ID  662
PARENTS = 1(#1)

MIRRORED GEOMETRY

NO. ELEMENT NAME    INFO                              STATUS
--- ------------ -----------                         ------
 1  Ref model    G2.PRT                                 Defined
 2  Copy datums  Copy datums                               Defined
 3  Dependency   Dependent                               Defined



  FEATURE'S ENTITIES AND GEOMETRY ARE IN LAYER(S):

  Datum plane surface DTM1, model G2.
    01___PRT_ALL_DTM_PLN - OPERATION = SHOWN
    01___PRT_DEF_DTM_PLN - OPERATION = SHOWN

  Datum plane surface DTM2, model G2.
    01___PRT_ALL_DTM_PLN - OPERATION = SHOWN
    01___PRT_DEF_DTM_PLN - OPERATION = SHOWN

  Datum plane surface DTM3, model G2.
```

```
    01___PRT_ALL_DTM_PLN - OPERATION = SHOWN
    01___PRT_DEF_DTM_PLN - OPERATION = SHOWN

   Curve (SYMMETRY LINE) in feature 12 (MERGE) of part G2.
    02___PRT_ALL_AXES - OPERATION = SHOWN

   Curve (SYMMETRY LINE) in feature 12 (MERGE) of part G2.
    02___PRT_ALL_AXES - OPERATION = SHOWN

   Curve (SYMMETRY LINE) in feature 12 (MERGE) of part G2.
    02___PRT_ALL_AXES - OPERATION = SHOWN


 Type of feature placement: fixed.
 MIRROR GEOMETRY OF ALL FEATURES BEFORE IT
 THIS FEATURE WAS CREATED BY MIRRORING
 END ADD

MASSPROP
END MASSPROP
```

The Pro/Engineer Assemby information is fully contained in this single script file that reacommodate the parts at your convenience.

Appendix 4

Pro/Engineer Assembly Integration Model

```
VERSION
REVNUM 147
LISTING FOR ASSEMBLY GGG

INPUT
END INPUT

RELATIONS
END RELATIONS


 ADD FEATURE (initial number 1)
 INTERNAL FEATURE ID  1
 TYPE = DATUM PLANE
 NAME = ASM_RIGHT


   FEATURE IS IN LAYER(S) :
     01__ASM_ALL_DTM_PLN - OPERATION = SHOWN
     01__ASM_DEF_DTM_PLN - OPERATION = SHOWN

 END ADD


 ADD FEATURE (initial number 2)
 INTERNAL FEATURE ID  3
 TYPE = DATUM PLANE
 NAME = ASM_TOP


   FEATURE IS IN LAYER(S) :
     01__ASM_ALL_DTM_PLN - OPERATION = SHOWN
     01__ASM_DEF_DTM_PLN - OPERATION = SHOWN

 END ADD


 ADD FEATURE (initial number 3)
 INTERNAL FEATURE ID  5
 TYPE = DATUM PLANE
 NAME = ASM_FRONT


   FEATURE IS IN LAYER(S) :
     01__ASM_ALL_DTM_PLN - OPERATION = SHOWN
     01__ASM_DEF_DTM_PLN - OPERATION = SHOWN

 END ADD


 ADD FEATURE (initial number 4)
 INTERNAL FEATURE ID  7
 TYPE = COORDINATE SYSTEM
 NAME = ASM_DEF_CSYS


   FEATURE IS IN LAYER(S) :
     05__ASM_ALL_DTM_CSYS - OPERATION = SHOWN
     05__ASM_DEF_DTM_CSYS - OPERATION = SHOWN

 END ADD

 ADD PART G1
 INTERNAL COMPONENT ID 39
 END ADD

 ADD PART G2
 INTERNAL COMPONENT ID 42
 PARENTS = 39(#5)
 END ADD

 ADD PART G3
 INTERNAL COMPONENT ID 44
 PARENTS = 42(#6)
 END ADD

MASSPROP
END MASSPROP
```

Appendix 5

Pro/Engineer Capabilities and Advantages

Every part that is built in Pro/Engineer will contain its own layout structure that specifies geometry, features, planes, cuts and holes by using its own Pro/E script.

In this way we can access to them and have our own “pre-configured” parts, so parts could be customized and adapted to aim the design.

Basic Design Automation with Pro/PROGRAM Pro/ENGINEER 2000i.

This presentation highlights the basic concepts of using Pro/PROGRAM to manipulate/automate a model. Assembly techniques are not represented here, although the concept is similar.

Overview

- ✓ Pro/PROGRAM allows you to vary your design by incorporating user prompts into the regeneration cycle.

- ✓ Use Pro/PROGRAM to manually delete, reorder, and suppress features, modify dimensions, and pause the regeneration process to add additional features.

Pro/PROGRAM vs. Family Tables

- ✓ Family tables are effective when you know the variations of the design or are sure they are not going to change, as in the case with part libraries (standard parts).

- ✓ Pro/PROGRAM is useful when you do not know the variations of the design in advance. You can create prompts for different values and parameters to display upon regeneration and build appropriate variations “on the fly.” After generating the variations of your design, you can save them to a family table.

Structure

- ✓ Header - First three lines of the program containing model name and program revision information. Input - Where user prompts and parameters are stored. This section is initially empty.
- ✓ Relations - This section contains all part or assembly relations.
- ✓ Model Section - Section in which you actually build the model. Contains series of paragraphs that contain information about each feature or component. You can build variations of your design by manipulating this section. Massprops - Use this section to automatically update the mass properties of the model when they change. Initially empty.

✓ Create generic model for basis of the design. Include features for necessary design variation. Add input statements - Create prompts to give model appropriate information.

✓ Write relations - Convey information from input statements to the model parameters. (Can also be added with Edit Rel option from the RELATIONS menu.) Edit model section - Add logic statements, generally “IF - THEN” statements, based on the input statements and relations.

✓ Create generic model for basis of the design. Include features for necessary design variation. Add input statements - Create prompts to give model appropriate information.

✓ Write relations - Convey information from input statements to the model parameters. (Can also be added with Edit Rel. option from the RELATIONS menu.) Edit model section - Add logic statements, generally “IF - THEN” statements, based on the input statements and relations.

- ✓ Create generic model for basis of the design.
- ✓ Include features for necessary design variation. Add input statements - Create prompts to give model appropriate information.

- ✓ Write relations - Convey information from input statements to the model parameters. (Can also be added with Edit Rel option from the RELATIONS menu.) Edit model section - Add logic statements, generally "IF - THEN" statements, based on the input statements and relations.

- ✓ Create generic model for basis of the design. Include features for necessary design variation. Add input statements - Create prompts to give model appropriate information.

Manipulating Features with Pro/PROGRAM

- ✓ Delete a feature or component - Delete all lines between and including the ADD and END ADD for that feature or component. •Reorder a feature or component - Cut all lines between and including the ADD and END ADD for that feature or component and paste it at another location in the program file. •Suppress a feature or component - Add the word "SUPRESSED" after the word ADD for that feature or component. • Resume a feature or component - Delete the word "SUPRESSED" after the word ADD for that feature or component. •Modify a dimension - Add the word “MODIFY” before a dimension in the model section of the program, then enter a new value for that dimension. Pause the regeneration - Add an “INTERACT” statement anywhere in the model section. When Pro/E regenerates the model, it pauses at this statement and asks if you want to add another feature to the model. Each time you add a new feature, it pauses again, allowing you to add more features. It executes the remainder of the program when you respond with a “no”. The “INTERACT” statement is then removed from the program and is replaced with the new features that you added.