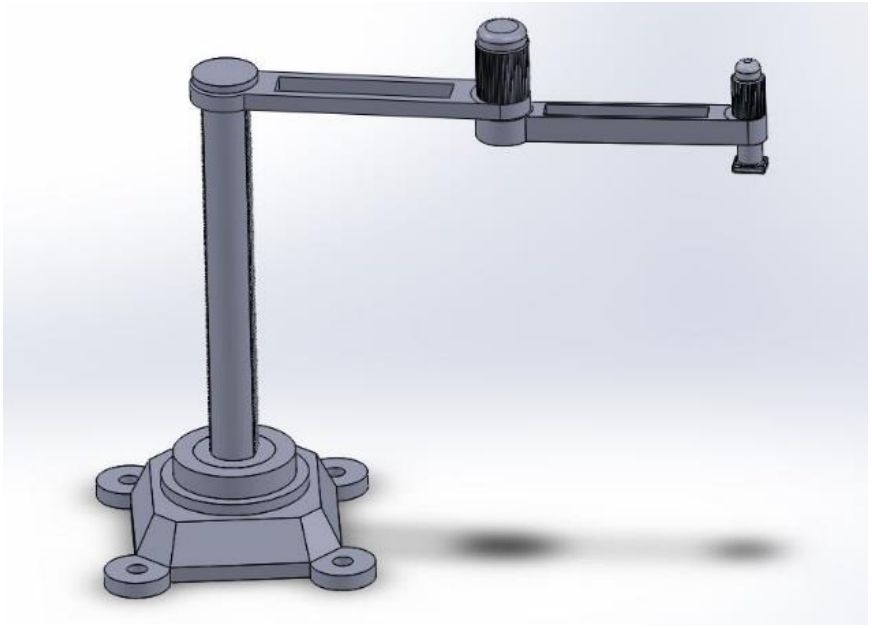

Optimization of Pallet Packaging Space and a Robotic SCARA Manipulator for Package Stacking

---

Group #4:, Erica Neuperger, Puneet Jethani, Siddhartha Kodgi, Zarvan Damania

MAE 598 DESIGN OPTIMIZATION

5/9/2015

Abstract

This project will focus on the optimization of stacking boxes in an industrial setting, using a SCARA robot. The task of optimizing the manipulator is divided into three individual subsystems (configuration, topology, and trajectory), which are independent enough to allow optimization to be conducted separately. For optimization of pallet space, the main objective is to obtain an optimum stacking order for the boxes. A genetic algorithm is run, which gives the order in which the boxes are to be stacked, their orientation, and their coordinates on the stock sheet. A single vertical layer of 2-D boxes is optimized to demonstrate the use of the algorithm. Next, the manipulator's link length is optimized for maximum manipulability, given torque limits at each joint. With this link length, optimal working posture is obtained. Once the link length is determined, the optimal cross section of the arm is found. Since mass has to be calculated every iteration, a neural network is used to train a model for mass and integrated it with the above system. The trajectory is then optimized to find a trajectory that maximizes profit by stacking as many boxes as possible while minimizing energy consumption. Here, a 15th order polynomial is obtained, whose profit is limited by the maximum allowable joint torques. Finally, the entire system is optimized until the values used for all of the subsystems converge. By the end of this study, the entire process of stacking boxes using a SCARA robot is obtained, where the final box positions, link lengths, link masses, and joint angular positions are determined.

Contents

1 Introduction

The number of robotic manipulators used in industry grows with each year. This is because of their ability to perform repetitive tasks both quickly and precisely. Here, the task of the robot for this project, is to pick up boxes off a conveyor and to place them onto a pallet. The boxes will be pre-sorted, in a specific order, to ensure that the pallet is loaded as efficiently as possible. In performing this stacking task, we want to robot to use a minimum amount of energy and time. We also want the structure of the robot to be as light as possible, without compromising the integrity of the structure. This project is categorized into 4 subsystems. They are:

1. Pallet Space optimization (subsystem 1).
2. Configuration optimization (subsystem 2).
3. Topology optimization (subsystem 3).
4. Trajectory optimization (subsystem 4).

The flow chart shown below show how the subsystems are integrated. A detailed description of each subsystem and their interdependence on other subsystem follows the flow chart.

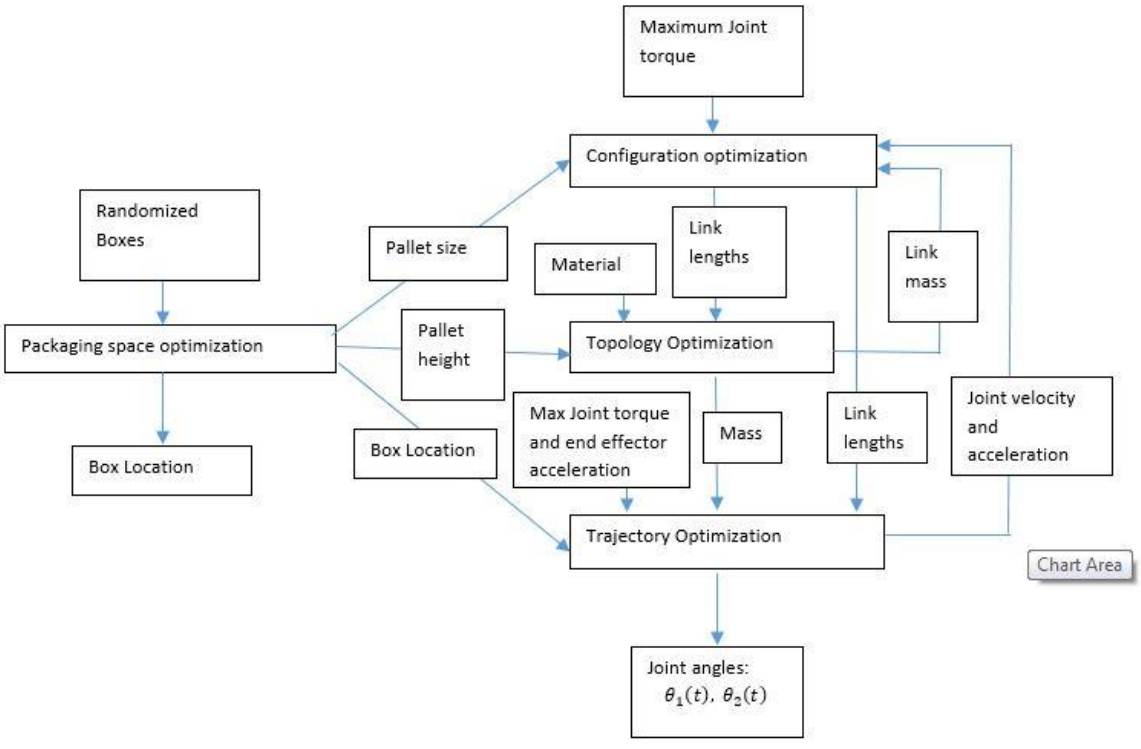

Figure 1.1:- System integration tree

1.1 Sub-system 1:

When stacking boxes on a pallet, a poor utilization of the space can result in using more pallets for packing less material. The boxes' stacking order is optimized to utilize the maximum volume and reduce the number of pallets used to transport same amount of material. This subsystem is constrained by the standard dimension of the pallet used for transportation. These constraints become inputs for other subsystems. The pallet size becomes an input for configuration optimization and the pallet height becomes input for the topology optimization. The results of this subsystem i.e, the optimum order of stacked boxes and their locations becomes the input for trajectory optimization

1.2 Sub-system 2:

While designing the robot it is necessary to find the best working posture (initial configuration) of a SCARA robot. An initial configuration of this robot is depicted in Fig. 1.2. Here, the robot's manipulability will be of utmost importance. Manipulability is a measure of the ability to move an arm from an initial configuration to some arbitrary position. This depends on link lengths and joint angles, which will affect the performance of manipulator. The link lengths *l*2 and *l*3 are to be optimized to obtain maximum manipulability. The height *l*1 depends on the pallet height here is fixed.

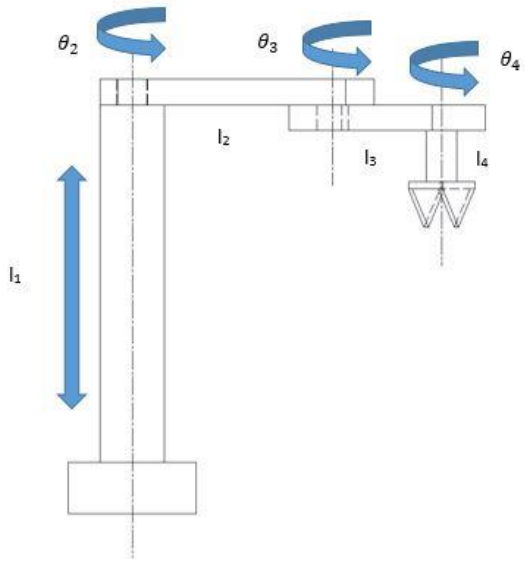

Figure 1.2 : SCARA Robot Configuration depicting the link lengths (l_1, l_2, l_3), rotational position of the joints $(\theta_1, \theta_2, \theta_3)$, and position of the prismatic joint (d)

1.3 Sub-system 3:

The arms of the robot will be a cantilever beam. The stress and strain on the arm will be examined under the load of a package to ensure that the manipulator's structure does not fail. Solid works is used to design a CAD model to determine the stress and strain load factors on the individual parts of the robot. The link lengths are obtained from subsystem 2 and optimized for mass. This is an iterative process which gives the optimum value of link lengths and mass of those links. These values are then given as inputs to the trajectory optimization subsystem.

1.4 Sub-system 4:

The last subsystem that will be optimized is the manipulator's trajectory. An optimal trajectory will be found that maximizes the total profit. This is done by maximizing the number of boxes stacked and reducing the amount of energy used. Here, the positions of the joints in time and the total amount of time required for the pick and place task will dictate the trajectory's performance. The link lengths and masses are obtained from the configuration and topology optimizations and the final position of the box is obtained from the packaging space optimization. Finally, the determined angular velocities and accelerations should be compared to the maximum values that were that were assumed in the configuration's subsystem. Should any of the values not agree, the optimization process should be repeated until the values converge.

2 Packaging Order Optimization--Puneet Jethani (Subsystem 1):

2.1 Problem Statement

This part of the project addresses the problem of placing rectangles onto a larger rectangular object in order to minimize the height of the nest. All items are rectangular boxes which are to be placed in a given space, by finding the best possible order of stacking. In the process of finding the best possible order of stacking the boxes may be rotated by 90 degrees so that they occupy the maximum possible space on the sheet. Also when all the boxes are placed, we look for the box contributing the highest height to the solution. We then try to place it by rotating it by 90 degrees. If it's not possible to fit the box by rotating we remove it from the solution.

2.2 Nomenclature

Stock sheet: The larger rectangle in which smaller rectangles are to be placed

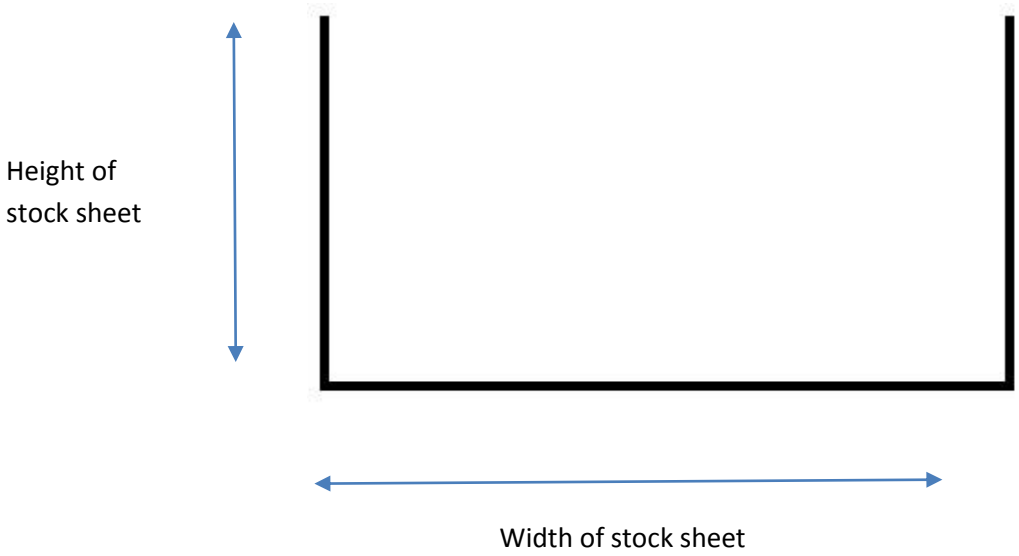

Figure 2.1:- Stock sheet

Ws	= Width of stock sheet
Hs	= Height of stock sheet
Gap_location	= X location of lowest gap
Niche	= Lowest available gap.
Towers	= Boxes with much more height than width.
Height_location	= Y Location of height
Height_array	= array of height of boxes placed in the order of placement.

Box = Matrix containing the list of boxes.

Solution_final = Matrix containing the order of boxes stacked, their orientation and the (X,Y) location.

2.3 Mathematical Model

As mentioned earlier, the main objective is to minimize the height of the nest, i.e the total height after placement of boxes.

Let's say the box at the maximum height is Y.

So the objective function would be:

$$\text{Min}(\text{Max}(Y)).$$

The width of the stock sheet is constant for a particular case and but can be specified by the user and that becomes a constraint.

$$\text{Width of stock sheet} = Ws,$$

where Ws is provided by user.

2.4 Model Analysis

The placement of these boxes is governed by three placement policies and has 3 stages.

These are the placement policies:

1) Place Leftmost:

   The leftmost niche-placement policy places a non-exact fitting rectangle at the left side of the niche.

2) Place Next to Tallest Neighbor.

   In this placement policy we examine the two rectangles on the stock sheet that define the lowest gap. We then place the best-fitting rectangle within the gap next to the tallest gap-defining rectangle.

3) Place Next to Shortest Neighbor:

   This placement policy is the opposite of the tallest-neighbor policy described above. This time, we place rectangles next to the shortest neighbor.

A detailed description and their computational cost will be discussed later in the report.

The three stages in the placement of boxes are:

2.4.1 Pre-processing stage:

All the boxes are denoted by a (width, height) pair. We examine all the rectangles first all do the following things.

1) If the height is bigger than the width, rotate the rectangles by 90 degrees such that width becomes more than height.
2) After doing step 1, arrange all the rectangles in decreasing order of their width.

Example: Consider the following set of boxes;
After step 1:

(3, 5), (5, 2), (1, 1), (7, 3), (1, 2) becomes
(5, 3), (5, 2), (1, 1), (7, 3), (2, 1)

After applying step 2:

(5, 3), (5, 2), (1, 1), (7, 3), (2, 1) becomes
(7, 3), (5, 3), (5, 2), (2, 1), (1, 1)

2.4.2 Packaging stage:

The basic procedure is to find the lowest gap and find the best fitting rectangle in that gap among all the rectangles available. Once we find the lowest gap and an appropriate rectangle that fits the rectangle, we place that rectangle in the solution and remove it from the list of available boxes. We then find the lowest available gap and find the appropriate rectangle and keep doing this procedure until all boxes are placed.

2.4.3 Post-Processing stage:

After the boxes are stacked, we look for towers and the boxes violating constraints. Since our main objective is to reduce the height of the nest, we try to rotate the towers and place them in the solution. If it's not possible to place the box by rotating it, we remove it from the solution. Also the boxes that violate the height constraint are removed from the solution.

2.5 Algorithm and optimization

After reading a lot of papers on this topic, a paper titled "…." which describes in detail the algorithm to place boxes and place them in order to minimize the height of the pallet

This is how the algorithm works

1. An array named ws_a is initialized. Each element of this array indicates the height at that particular location. The lowest number on this array denotes the lowest gap. The number of times that lowest number is repeated defines the width of the gap.
2. After the boxes are sorted as described in pre-processing stage, the entire list of boxes available are scanned.
3. Then the box that fits the lowest gap is found from the list, the solution is updated and the box placed in the solution is from the list of available boxes.
4. This is an iterative process and the iteration keeps on going until the constraints are violated or all the boxes in the list are placed.

2.5.1 Choosing between placement policies

In section 1.4, I described briefly three placement policies for placement of boxes. The placement policy implemented by me is the 'Place left most policy'.

The 3 policies are described in detail below:

- Place leftmost policy:

  The leftmost niche-placement policy places a non-exact fitting rectangle at the left side of the niche. This policy involves least time complexity.

- Place Next to Tallest Neighbor:

  In this placement policy we examine the two rectangles on the stock sheet that define the lowest gap. We then place the best-fitting rectangle within the gap next to the tallest gap-defining rectangle as shown in figure. If the lowest gap is defined by a rectangle and the sheet side, we place next to the sheet side.

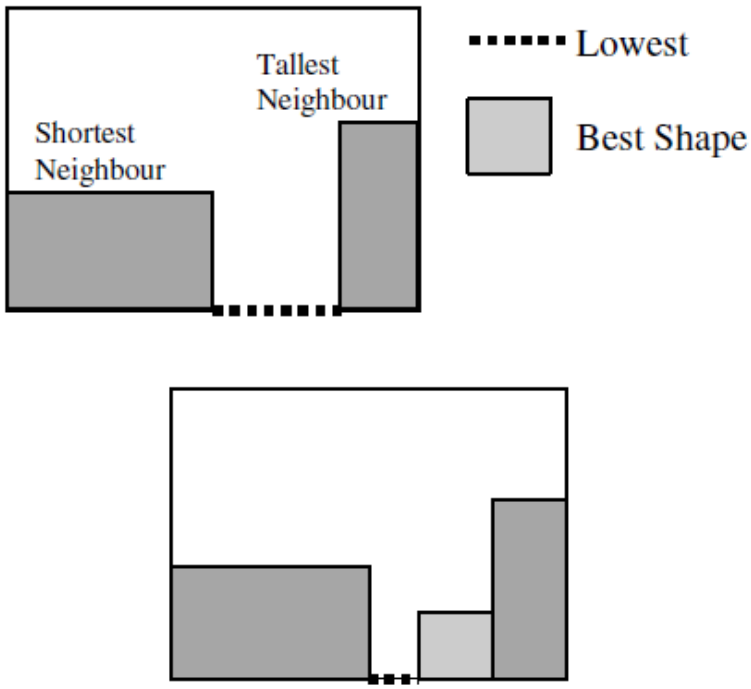

Figure 2.2:-Place next to tallest neighbor

After the tallest neighbor and the best fit are determined, the best fitting box is place next to the tallest neighbor as shown below.

- Place next to shortest neighbor:

This placement policy is the opposite of the tallest-neighbor policy described above. This time, we place rectangles next to the shortest neighbor.

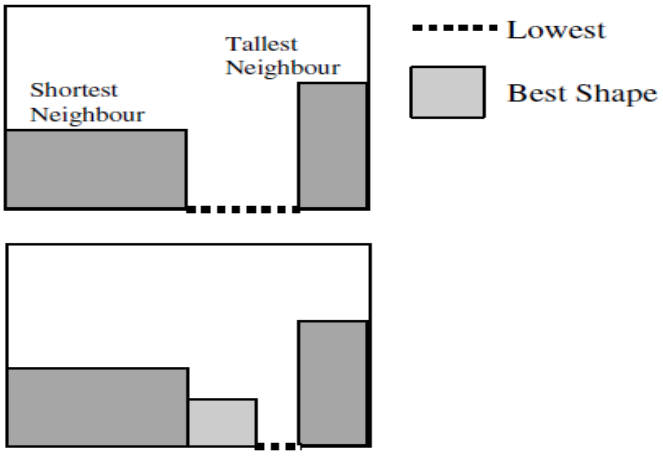

Figure 2.3:- Place next to shortest neighbor

2.5.2 Computational cost:

I. Place at leftmost:

The main computational cost involved in this policy is the iterative process of:

a. Finding the lowest gap.
b. Examining the list of available boxes.
c. Finding the best fit and its location.

As the number of boxes increase and with large variations in the size of the boxes, the number of iterations also increase.

II. Place next to tallest neighbor:

The main computational cost involved in this policy is the same as in the place leftmost policy.

But it involves a few additional steps which have to be performed in each iteration. They are:

a. Determining the tallest neighbor.
b. Placing the best fit next to the tallest neighbor.

Again as the number of boxes increase, the number of steps in each iteration increase which increase its computational cost. It also increases programming and implementation complexity as the code requires additional loops to do the additional steps.

III. Place next to shortest neighbor:

Things are same in this placement policy as in the place to tallest neighbor. There are a few extra steps, which are:

c. Determining the shortest neighbor.
d. Placing the best fit next to the tallest neighbor.

Again as the number of boxes increase, the number of steps in each iteration increase which increase its computational cost. It also increases programming and implementation complexity as the code requires additional loops to do the additional steps.

2.5.3 The trade-off

Considering the computational cost and the programming complexity, implementing the leftmost policy is the best trade-off for this problem.

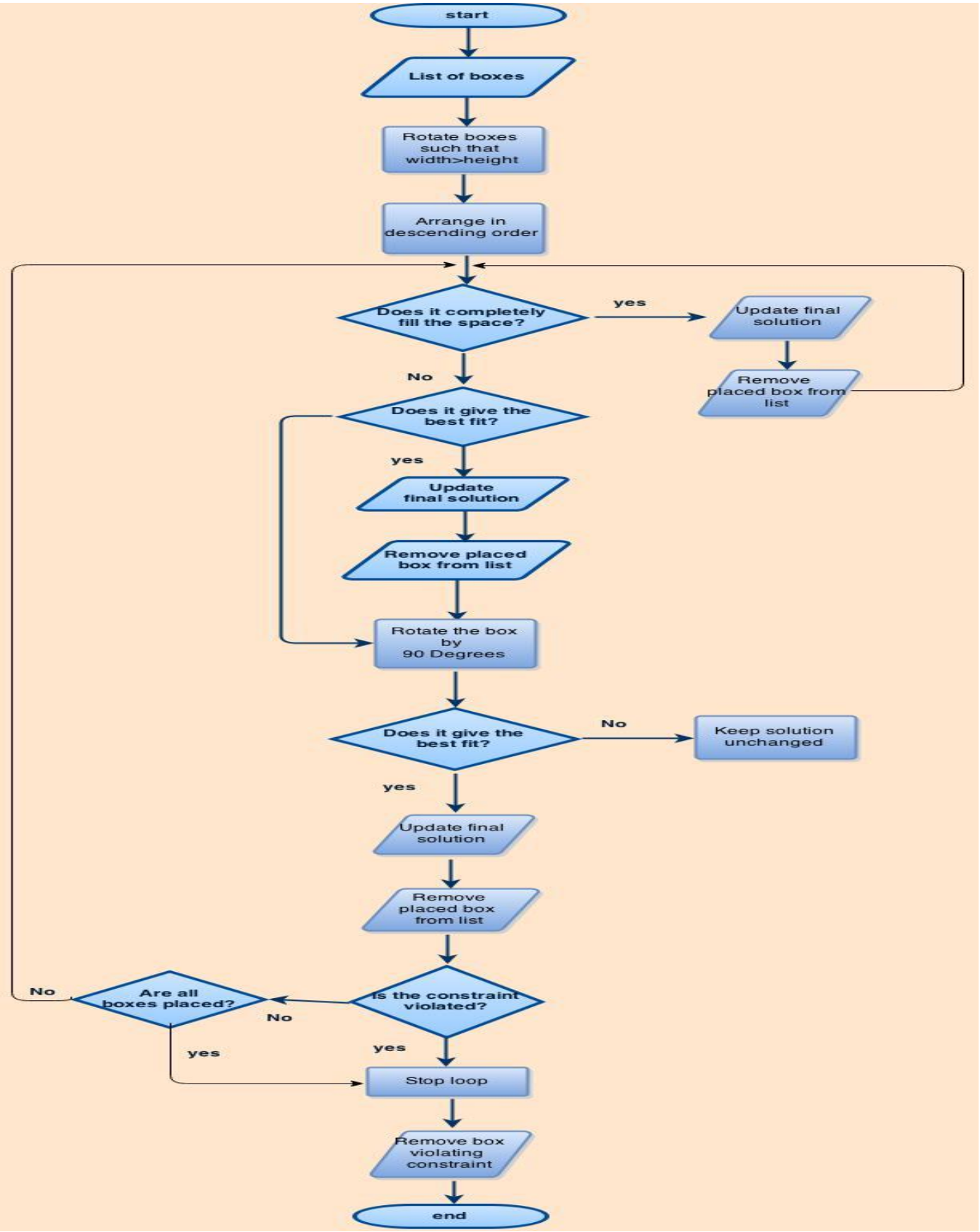

Figure 2.4:- Flow chart of the algorithm

2.6 Implementation of the algorithm

2.6.1 Unsuccessful attempt:

In the first attempt to program the algorithm, the array ws_a described above was not used to store the cumulative height at different locations. Instead the elements of the array were deleted when the boxes were placed. The number of elements deleted were equal to the width of the box placed on the stock sheet. The main problem was that the program used to stop after one layer of boxes were placed on the stock sheet. I reached a dead end as there was no feasible method to find the lowest gap to place the next box.

Course of correction:

The ws_a array was used for storing cumulative height at different locations instead of deleting elements as the boxes were placed. And as explained above, the elements in the ws_a were used to determine the lowest gap and the best fit.

2.6.2 Corrected implementation

Here, a few iterations of the process are described to explain how the code and the algorithm works.

1st iteration:

When the 1st iteration is run, the packaging space is totally empty so the entire space is the lowest gap and ws_a comprises of only zeros. After 1st iteration is run, these are the output.

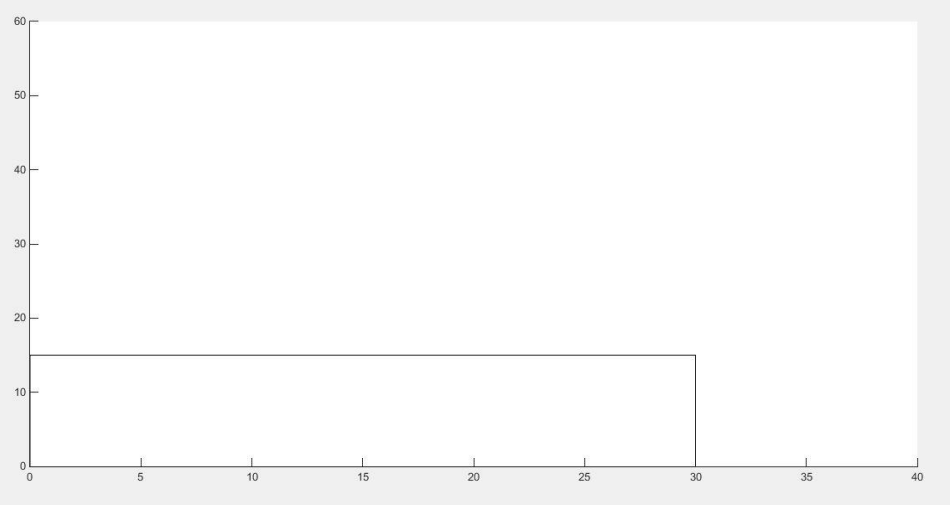

Figure 2.5:- Iteration-1

The solution is: 30, 15, 0, 0, where 30, 15 are the box dimensions and 0, 0 are the coordinates of the bottom-left point of the box.

This is the value of the ws_a after the 1st iteration is run.

```
ws_a =

  Columns 1 through 22

    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15

  Columns 23 through 40

    15    15    15    15    15    15    15    15     0     0     0     0     0     0     0     0     0     0
```

Figure 2.6:- ws_a array after 1st iteration

2nd iteration:

The ws_a array is examined. As we can see, the lowest element in the array is 0. So the lowest gap available to place the box would be 0. This 0 is repeated 10 times. So the width of the lowest gap is 10. So any box which has a width less than or equal to 10 can be placed in this gap directly or by rotating, whichever gives a better fit. This is the result of the 2nd iteration.

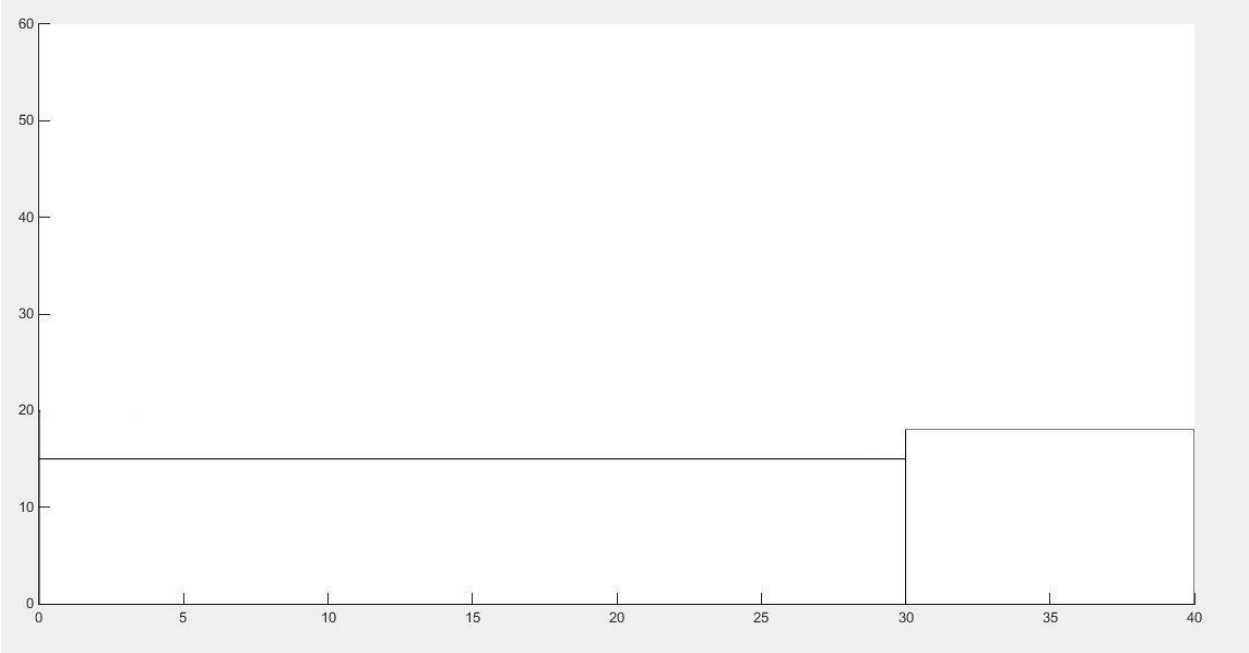

Figure 2.7:- Iteration-2

The ws_a array after the 2nd iteration result.

```
ws_a =

 Columns 1 through 22

   15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15    15

 Columns 23 through 40

   15    15    15    15    15    15    15    15    18    18    18    18    18    18    18    18    18    18
```

Figure 2.8:- ws_a array after 2nd iteration

3rd iteration:

At the end of 2nd iteration, we can see from ws_a that the lowest gap is 15 and it's just 30 unit long. This is what we get as a result.

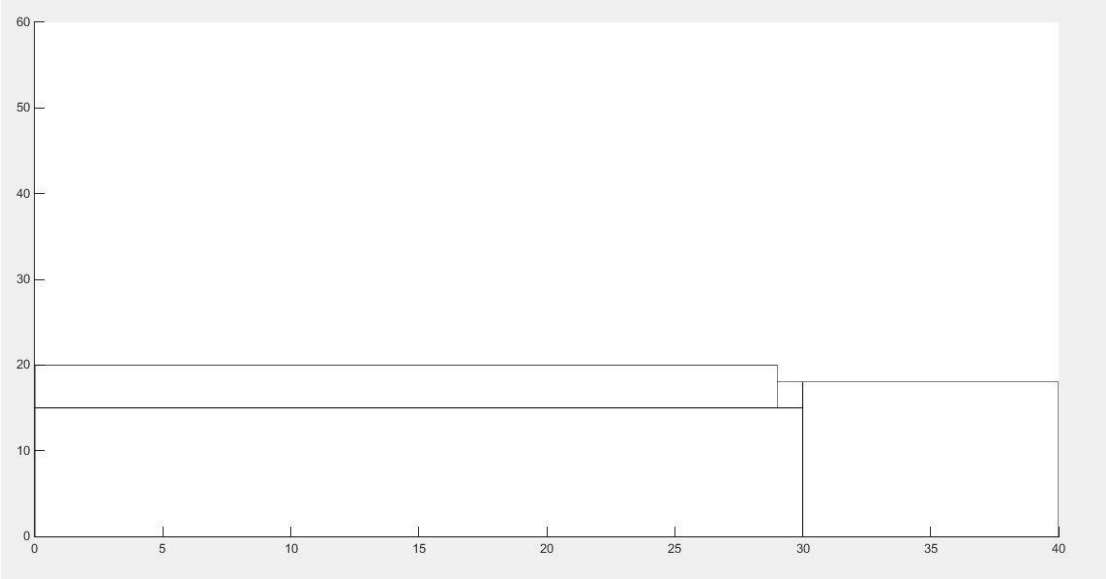

Figure 2.9:- Iteration-3

The ws_a array after the 3rd iteration result.

```
ws_a =

 Columns 1 through 22

   20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20

 Columns 23 through 40

   20    20    20    20    20    20    20    15    18    18    18    18    18    18    18    18    18    18
```

Figure 2.10:- ws_a array after 3rd iteration

So for the next iteration, the lowest gap is 15 with gap width as 1. Since we don't have a box with any dimension, we fill the gap. This what the ws_a looks like after this iteration.

```
ws_a =

  Columns 1 through 22

    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20    20

  Columns 23 through 40

    20    20    20    20    20    20    20    18    18    18    18    18    18    18    18    18    18    18
```

Figure 2.11:- Filling the hole

This process keeps on continuing until all boxes are placed or the constraints are violated.

2.7 Parametric study

The flow chart described above was tested with 3 different set of boxes. The results were obtained for all set of boxes. The efficiency of packing depends on the set of boxes and their dimensions. Details on how best the result is will be discussed in the next section.

2.8 Discussion of results

The results obtained by using 3 different sets of boxes are shown below. The efficiency of packing is defined by the empty space or holes created in the packing until constraints are violated. The following applies to results from all 3 set of boxes:

- The circled boxes are the ones that violate constraints. They are removed solution afterwards.
- The solid black part are the holes where no boxes can be filled. So it's filled with some packing material.

Comparisons with earlier Heuristic methods:

Earlier methods like bottom-left place, used to place boxes in the bottom-left most position and in the order the boxes were supplied to them. So there were possibilities of boxes over-lapping. Since this algorithm previously sorts the boxes before it is fed to the algorithm, it finds the best-fit for the space. It also dynamically searches for the best-fitting box and its proper orientation. Thus it gives a better utilization of space.

Results with 1st set of boxes:

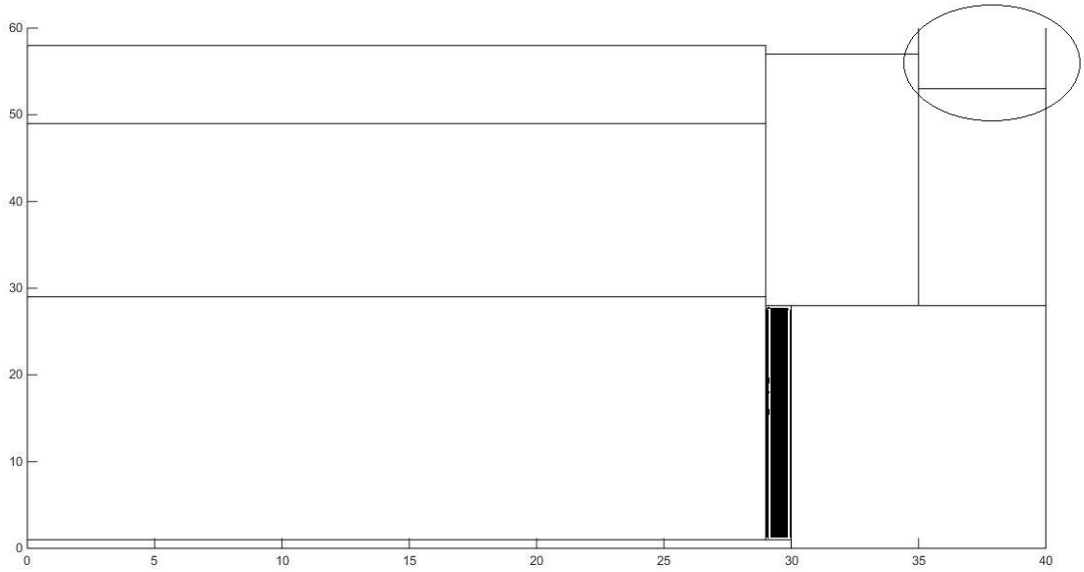

Figure 2.12:- Box violating constraints

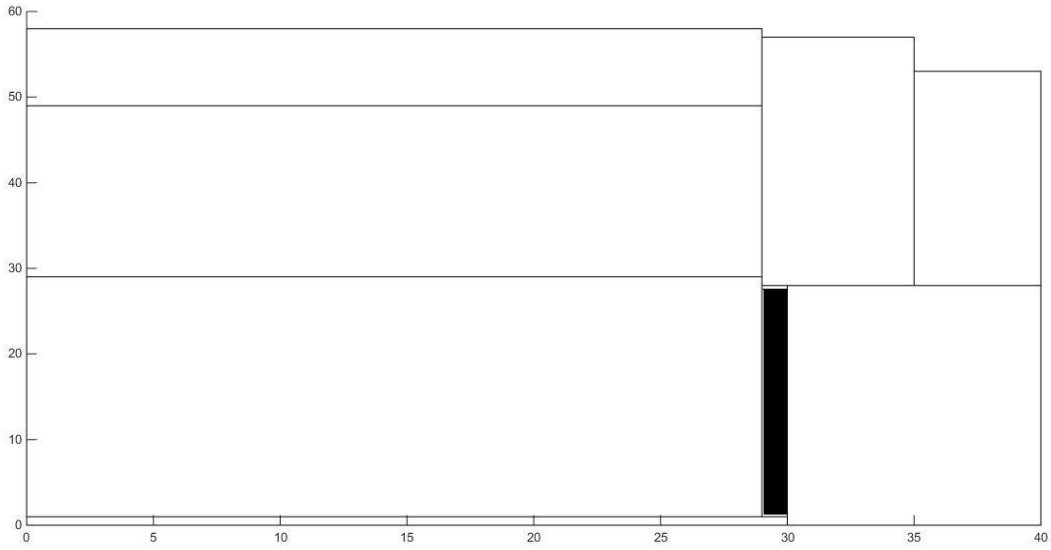

Figure 2.13:- Corrected results

Result from 2nd set of boxes:

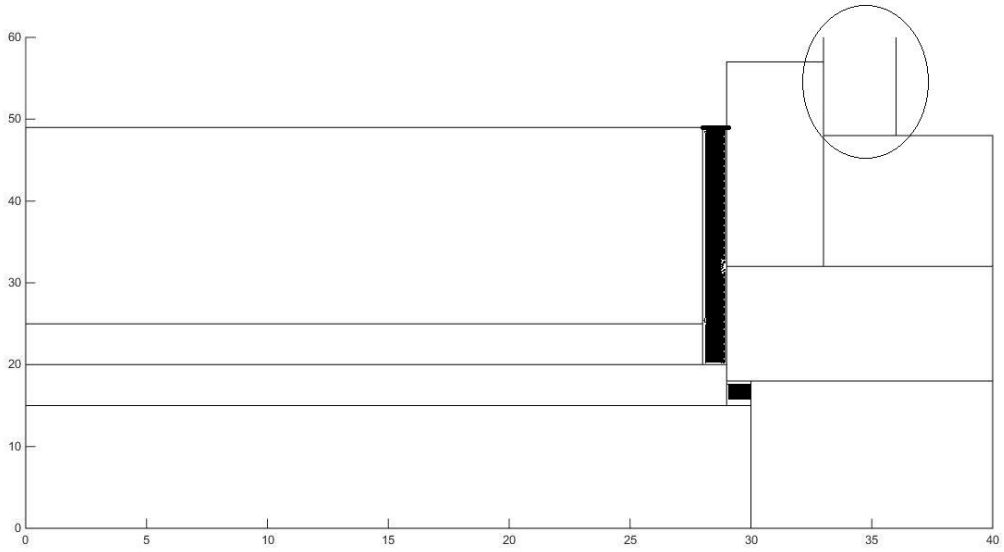

Figure 2.14:- Box violating constraints

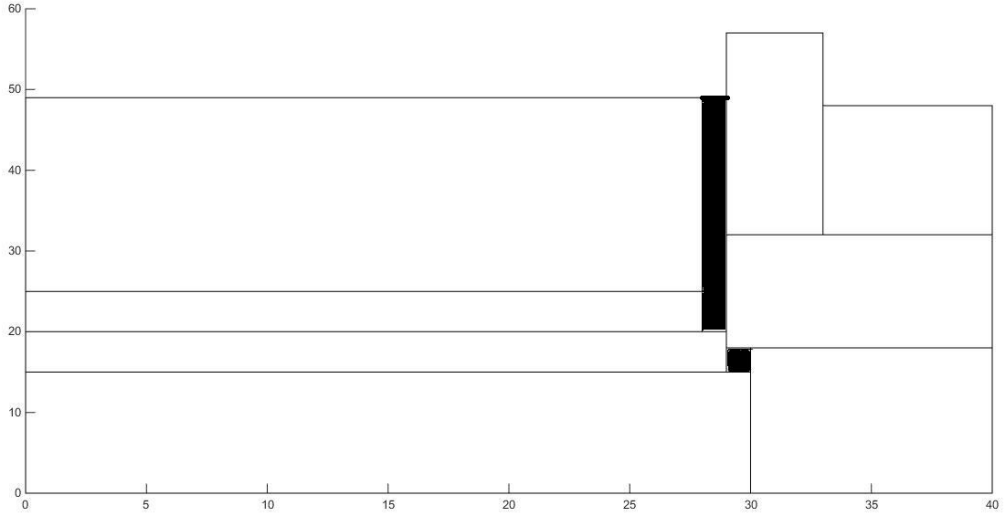

Figure 2.15:- Corrected results

Results from 3rd set of boxes:

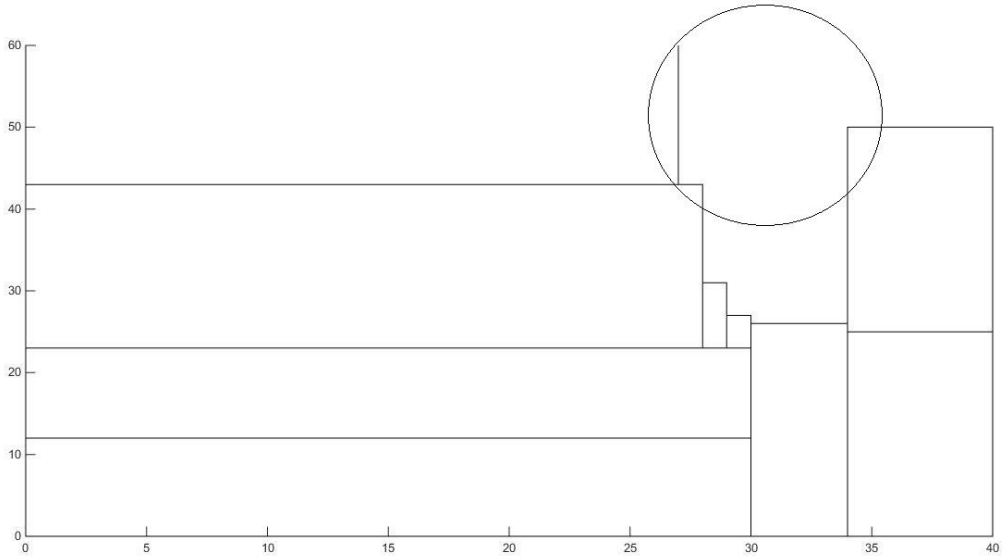

Figure 2.16:- Box violating constraints

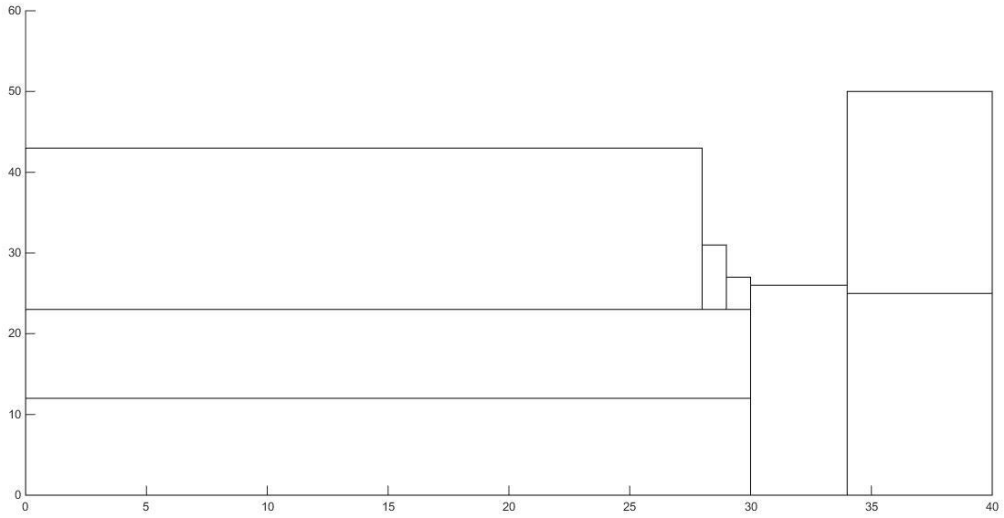

Figure 2.17:- Corrected results

2.8.1 Interpretation of results

Since the set of boxes available are different, the results obtained and the efficiency of packing will differ every time. As you can see, the result in box set 1 has 7 boxes and 8 boxes for in the result from set 2 and 3. The algorithm described above does a very good job in identifying the best-fitting box and placing it in the right orientation.

3 Robot arm configuration optimization - Siddhartha Kodgi (Subsystem 2)

3.1 Problem

In the design phase of the robotic arm it is necessary to consider the initial orientation of the robot along with load capacity, speed, workspace etc... This is because there will be different orientation of robotic arm for different length and joint angles which will directly affect the performance of the manipulator. Even when modifying an existing design of robotic arm, small changes to one part of the robot can drastically alter the performance of the whole manipulator. Changing a single Denavit-Hartenberg parameter for a manipulator may reduce the dimensionality of the robot's workspace and by increasing the size of one actuator may cause other actuators torque limits to be exceeded, also by increasing the mass of one link may require other links to be strengthened. And also increasing or decreasing joint velocity and acceleration will affect the joint torques which in turn affect link length. Therefore, a quantitative measure of performance of arm for its initial configuration will be beneficial. The concept manipulability measure is one such measure which directly relates performance with the arm specifications. So, it is imperative to select an initial position of robotic arm such that the effort to move the arm to an arbitrary location from initial position will be minimum.

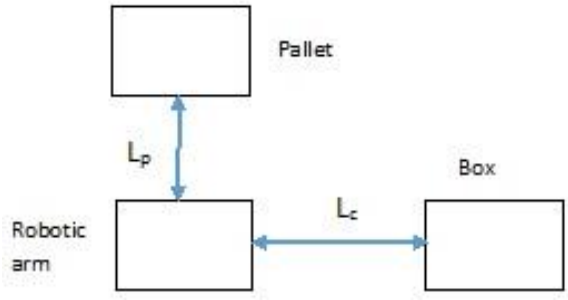

Figure 3.1:- Layout

For the given robot configuration, the Scara robot, we would like to design the optimal working posture. The design problem is that the motor is specified, that means there is restriction in terms of maximum torque available at each joint. By maximizing the manipulability measure, an optimal posture will be found based on limits imposed by torque requirements. The torque restriction sets the upper limit for the length link lengths, while the manipulability measure pushes for larger link. So, we can find the optimal posture that is within our torque limitation.

3.2 Nomenclature

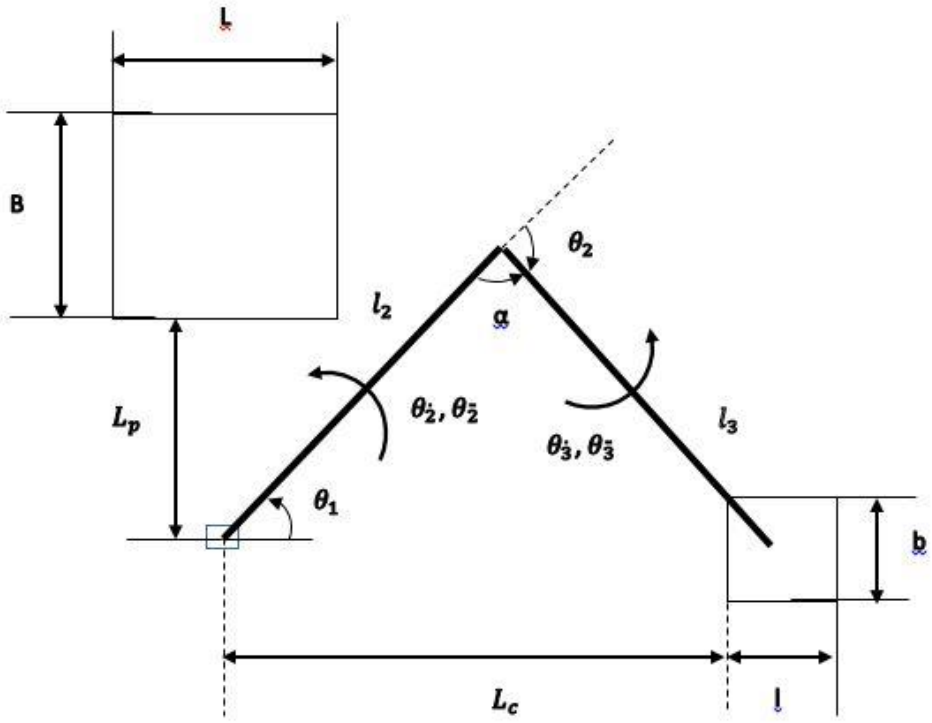

Figure 3.2:- Nomenclature

Joint angles: θ_1, θ_2

Joint velocity: $\dot{v_2}$, $\dot{v_3}$

Joint acceleration: $\ddot{v_2}$, $\ddot{v_3}$

Link lengths: l_2, l_3

Masses of links:m_{l_2}, m_{l_3}

Mass of joints:m_{m_2}, m_{m_3}

Mass of end effector: m_e

Max torque allowed for joint: τmax_i

Torque at each joint: τ_i

Mass of box: m_b

Distance of arm from pallet: L_p

Distance of arm from conveyor: L_c

Pallet dimension: L, B, H

Box dimension: l, b, h

w = manipulability measure

L_{g2} = Centre of mass of link-2

L_{g3} = Centre of mass of link-3

I_2 = moment of inertia of link-2 about CG

I_3 = moment of inertia of link-3 about CG

3.3 Objective function

The objective of the problem is to maximize the manipulability of the robot manipulator. Manipulability is the ability of manipulating or moving robotic arm to some arbitrary position at minimum effort. The movement of arm is dependent on the initial configuration and path planning. So this sub-system focuses on finding the optimized initial configuration of robot manipulator, which is dependent on link length and joint angles.

The objective function of this study is to maximize the manipulability measure (w). So we need to find optimum length and joint angles for which we have maximum manipulability measure. The equation for manipulability measure.

$$w=\sqrt{\det(J(\theta)*(J^T(\theta)))}$$

where J is jacobian matrix and θ is joint angles, with the help Denavit-Hartenberg parameter J for our case is derived

$$J=\begin{bmatrix} 0 & -\left((l_3*s_{12})+(l_2*s_1)+(l_2*s_{123})\right) & -\left((l_3*s_{12})+(l_2*s_{123})\right) & -\left((l_3*s_{123})\right) \\ 0 & \left((l_2*c_{12})+(l_1*c_1)+(l_3*c_{123})\right) & \left((l_2*c_{12})+(l_3*c_{123})\right) & \left((l_3*c_{123})\right) \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

$s_{123}=\sin(\theta_1+\theta_2+\theta_3)$, $c_{123}=\cos(\theta_1+\theta_2+\theta_3)$, $s_{12}=\sin(\theta_1+\theta_2)$, $c_{12}=\cos(\theta_1+\theta_2)$

$s_1=\sin(\theta_1)$, $c_1=\cos(\theta_1)$

There for $$w=l_1*l_2*\sin(\theta_2)$$

From the figure

$$\alpha=\cos^{-1}\frac{l_1^2+l_2^2-(L_c+0.5*b)^2}{2*l_1*l_2}$$

$$\theta_2=\pi-\alpha$$

Substituting for θ_2 in w, we get

$$w = 0.5 * \sqrt{((4 * l_1^2 * l_2^2) - (l_1^2 + l_2^2 - (Lc + 0.5 * b)^2)^2)}$$

So the objective is to maximize w for l_1 and l_2

3.4 Constraints:

Since we are designing for maximum manipulability, the best solution will be when l_2 and l_3 are approaching infinity. So to bound variables for upper side we use torque of the motor-1 and motor-2 as constraint. Torque of each motor is dependent on the inertial force, Coriolis force frictional force and gravitational force. Since Scara robot is moving parallel to the ground, link-2 and link-3 is not affected by gravity. So torque for motor-2 and motor-3 are

$L_{g2} = (m_{l_2} * l_2/2 + m_{l_2} * l_2)/ (m_{m_2} + m_{l_2})$;
$L_{g3} = (m_{l_3} * l_3/2 + (M) * l_3)/ (m_{l_3} + M)$;

$I_2 = (l_2 - L_{g2})^2 * m_{m_2} + 0.8333 * m_{l_2} * l_2^2 + m_{l_2} * (0.5 * l_2 - L_{g2})^2$;
$I_3 = (l_3 - L_{g3})^2 * M + 0.8333 * m_{l_3} * l_3^2 + m_{l_3} * (0.5 * l_3 - L_{g3})^2$;

$\tau_2 = ((I_2 + m_{l_2} * L_{g2} + 0.4001 + I_3 + m_{l_3} * (l_2^2 + L_{g3}^2 * + 2 * l_2 * L_{g3} * \cos\theta_2) + m_{m_3} * l_2^2) * \ddot{v_2} +$
$(I_3 + m_{l_3} * (L_{g3}^2 + l_2 * L_{g3} * \cos\theta_2) + 0.02) * \ddot{v_3} - (m_{l_3} * l_2 * L_{g3} * \sin\theta_2) * (2 * \dot{v_2} * \dot{v_3} - \dot{v_3} * \dot{v_3})$

$\tau_3 = (I_3 + m_{l_3} * (L_{g3}^2 + l_2 * L_{g3} * \cos\theta_2) + 0.02) * \ddot{v_2} + (I_3 + m_{l_3} * L_{g3}^2 + 0.4) * \ddot{v_3}$
$+ m_{l_3} * l_2 * L_{g3} * \sin\theta_2 * \dot{v_3} * \dot{v_3}$

Where

$$g_1: \tau_2 \leq \tau max_2$$
$$g_2: \tau_3 \leq \tau max_3$$

But the maximum torque value is the random value assumed for our problem. But it is not sufficient that l_2 and l_3 we get will ensure maximum manipulability for given torque constraints, But it must reach every required point in the workspace. So it must satisfy these constraints also.

$$g_3: \sqrt{(L + L_p)^2 + (0.5 * B)^2} \leq l_1 + l_2$$
$$g_4: -l_1 \leq 0$$

$$g_5: -l_2 \leq 0$$

In order to optimize the length of the arm we need to select a point on the path of the arm movement. Since we don't know the path we have assumed that the center of robot base and center if box lie on the same axis at the distance of Lc along x-axis. Therefore the model should satisfy this equation.

3.5 Design Variables and Parameters:

3.5.1 Design Variables:

The design variables are link length l_2,l_3 and distance between box and manipulator base L_c. We are not considering link 1 because it will not affect other links. The link 1 length is dependent on the height of the pallet, and it is fixed.

3.5.2 Parameters:

mm_2 = 4.5 kg

mm_3 = 3.5 kg

M = 22 kg

τmax_1 = 80 N.m

τmax_2 = 10 N.m

Lc= 0.94 m

Lp= 0.2 m

L = 1.21 m

B = 1.016 m

l = 0.3 m

b = 0.3 m

$v_{\dot{1}}$ = 1 rad/sec

$v_{\dot{2}}$ = 1.5 rad/sec

$v_{\ddot{1}}$ = 8 rad/sec^2

$v_{\ddot{2}}$ = 20 rad/sec^2

3.6 Model Analysis

The task of this subsystem is to get an optimized initial configuration for robot manipulator. So to perform this task we divided the optimization process into two steps. First step is calculating optimal length for the given torque limit for some arbitrary length between manipulator base and box (L_c). When we did monotonicity analysis for objective function we found link lengths l_2,l_3 are monotonically increasing. But it's difficult to perform monotonicity analysis for constraints equation because equations are complex, but from intuition we can see that torque constraints should be active. This is because those are the only constraints which is bounding the lengths from upper side. This means link lengths are monotonically decreasing in torque constraints.

Second step is to get optimized length between manipulator base and box with the optimized length calculated from the step 1 for better manipulability. In this step we will find how well we can place the manipulator for given link lengths. Even for this case the objective is to maximize manipulability, objective function is the same. The length L_c assumed in the first step is varied until we get the output of second step as same as first one.

We can see that torque of each joint is dependent on mass of the link. But it's difficult to find mass because we don't know the cross section of the link. In order to find mass for every iteration we used neural network to train the data and generate a model, and used neural network model to calculate mass.

3.7 Optimization Study:

The optimal solution was found using Matlab fmincon function with the starting point [1, 1]. Used three methods to solve the optimization problem active-set, interior-point and Sequential Quadratic programming (SQP) to solve the optimization problem. The optimal solution from different method was found to be

Table 3.1:- Optimized results

Active-set	Interior-point	SQP
l_1 = 36.3 in l_2 = 22.6 in	l_1 = 36.3 in l_2 = 22.6 in	l_1 = 36.3 in l_2 = 22.6 in

From the above table we can see that all the methods gave the same output. The graph below shows the final configuration of manipulator

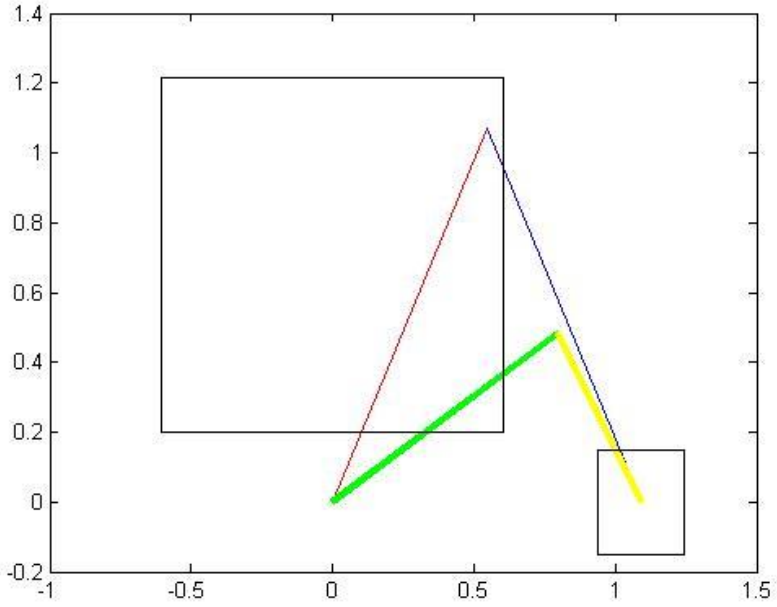

Figure 3.3:- Final Configuration

Step-2 Since this problem has only objective function which needs to be maximized with no constraints, used newton method and gradient to get the optimal length for L_c

$$L_c = 0.94$$

In order to verify that, this is a solution for given problem, KKT analysis is done. After running the optimizaton problem, we found that inequality constraints 1 and 3 were found to be active. This is because velcocity and acceleration parameter for joint 2 are high. This will drive the torque to joint-2 to reach its limit. From KKT analysis we found mu1 and mu3 will be greater than zero and mu2 will be equal to zero.

The lagrange function for this problem is L= $(c_1 * mu_1)$+ $(c_2 * mu_2)$+($(c_1 * \lambda_1)$)

So after solving the above equations we got mu_1 = 0.01, mu_2 = 1.43, mu_3 = 0. We can see that $mu_2 > mu_1$. This implies constraints 3 is dominant than 1. This is true because if we remove the constraints 3 we found that constraints 1 and 2 is active.

3.8 Parametric Study:

The solution for different sets of parameter values were obtained. Each parameter was independently varied to check how it will affect the manipulability. The parameters which are being varied are maximum torque of each motor, joint velocities and joint accelerations. The effect of parameter is studied by assuming a square cross section of link. So keeping height and width constant, mass is only dependent in link length.

3.8.1 Effect of Torque

The main parameter which effect the link length is torque. The figure-3.4 shows how link lengths are affected by joint torque. We can see that link lengths are not independently dependent on respective joint torques. That is when toque of motor-2 is reduced, length of link-2 changes significantly and length of link-3 changes by small amount. When torque of motor-3 is reduced length of link-3 changes significantly and length of link-2 changes by small amount.

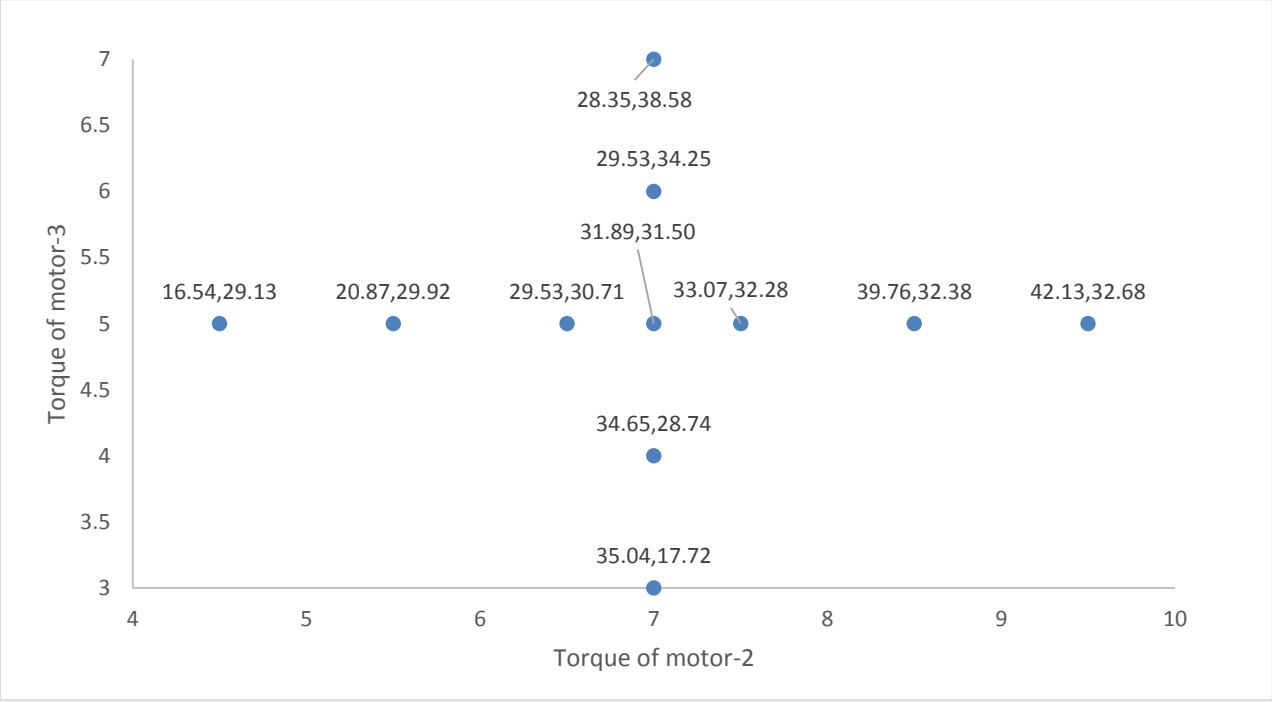

Figure 3.4:- Effect of torque on link lengths

3.8.2 Effect of angular velocity

This figure-3.5 shows how link lengths are affected by angular velocity of links. We can see that link length 2 and link length -3 are directly related to angular velocity of link-2. When we increase the angular velocity of lin-2 link lengths decreases. But it's different in the case where we change angular velocity of link-3. From graph we can see that link length-3 decreases with increases in angular velocity of link-3 but link length-2 increases. This is because of two reasons.

1. When we increase angular velocity of link-3, for the same maximum torque constraint link length has to decrease to satisfy the constraints. This decrease the mass of link-3, which decreases the load acting on the link-2, which increases the link length.
2. Second and important one is the torque equation. In the equation we see that the angular velocity is being subtracted from torque equation of link-2. So as you increase the velocity torque of joint decreases. This is only applicable for link-2.

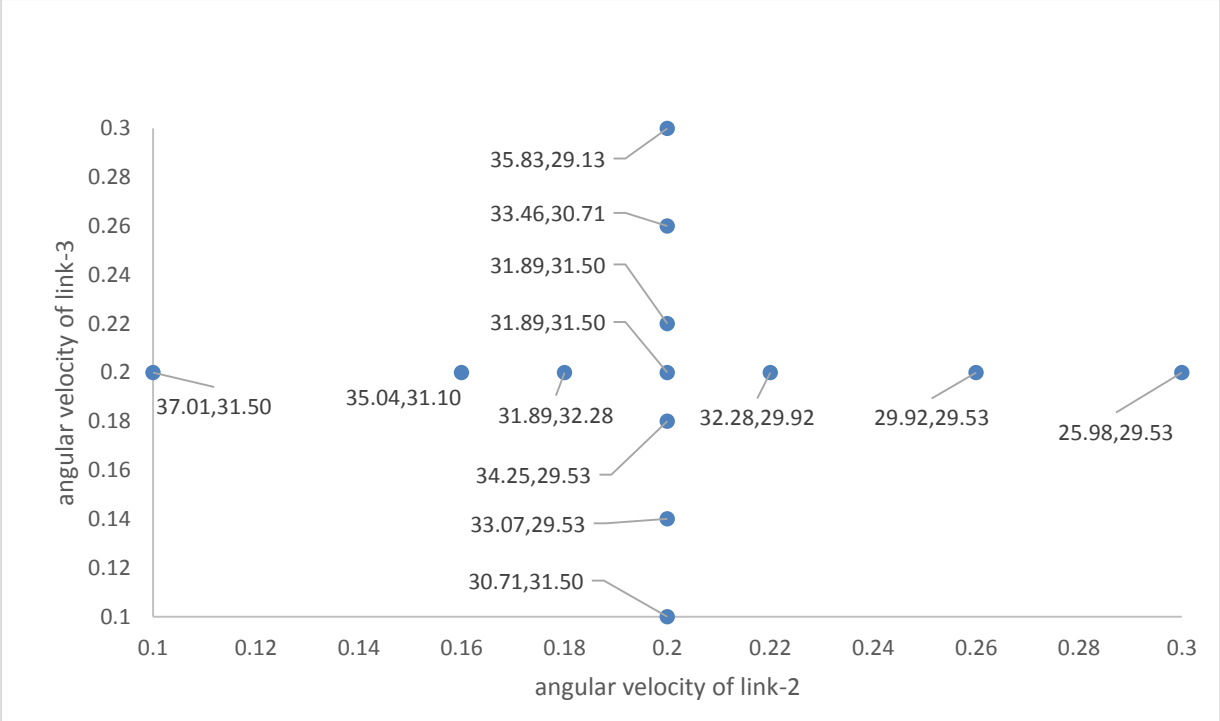

Figure 3.5:- Effect of angular velocity on link lengths

3.8.3 Effect of angular acceleration

This figure-3.6 shows how link lengths are affected by angular acceleration of links. When we increase the angular acceleration of link-2 link lengths decreases. But it's different in the case where we change angular acceleration of link-3. From graph we can see that link length-3 decreases with increases in angular acceleration of link-3 but link length-2 increases. This is because of mass. When we increase angular velocity of link-3, for the same maximum torque constraint link length has to decrease to satisfy the constraints. This decrease the mass of link-3, which decreases the load acting on the link-2, which increases the link length. We can see that effect of angular acceleration is similar of angular velocity. The only difference is the rate of change of length. It's more in case of angular acceleration than compared to angular velocity.

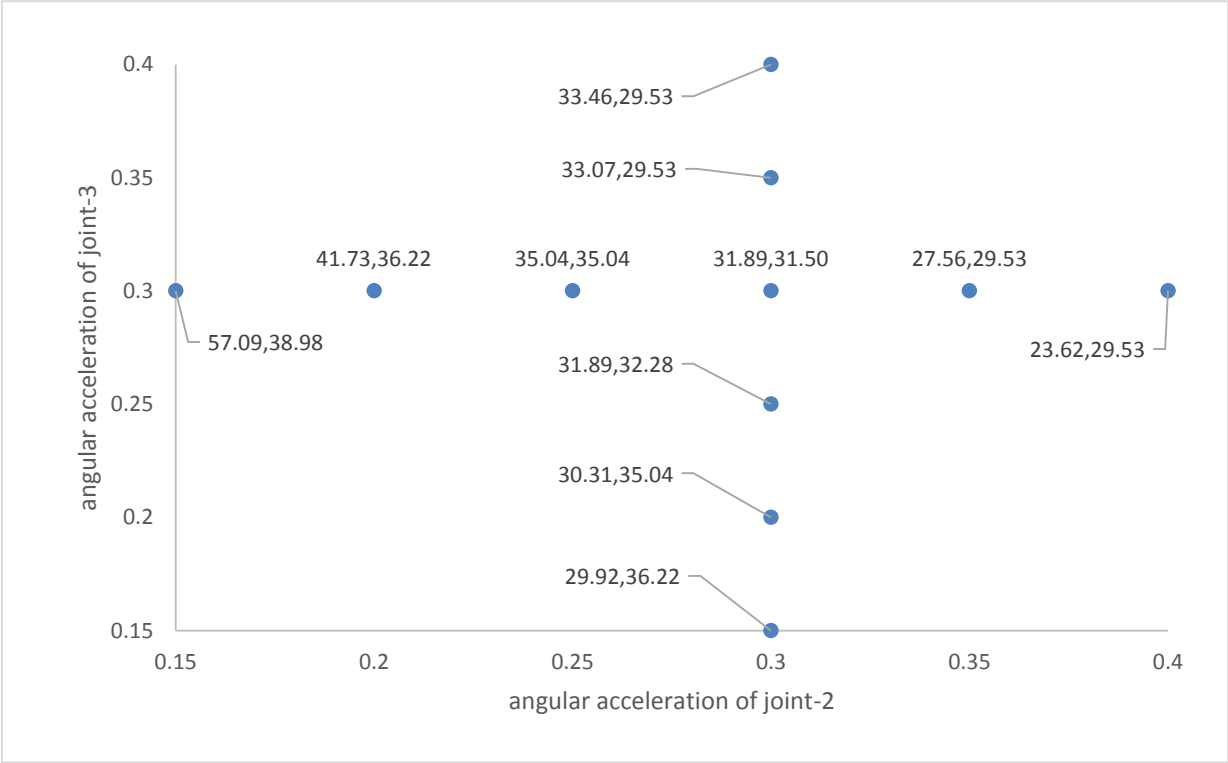

Figure 3.6:- Effect of angular acceleration on link lengths

3.9 Discussion of results

As shown above in the optimization and parametric study, the results obtained from the Matlab appears to be very reasonable because of their stability (for different starting points) and the satisfaction of the KKT conditions. As explained in optimization study constraints 3 is dominant. This happens because joint angular velocity and acceleration are high. When these values are less we can see that both torque constraints will be active for this problem.

We can increase the manipulability by decreasing the joint velocity and acceleration. This will increase the workspace but need more time to complete the task. The best solution is to decrease the acceleration, which will drastically change the link lengths. Another way to increase manipulability is by increasing link lengths which is achieved by relaxing the torque constraints. This will also increase the robot workspace, but increases the overall cost of the robot. If the objective is to lift heavy weight than we need to decrease acceleration and velocity. So depending on the problem suitable parameter is selected. So, it would be more practical to first determine which motor fits best for our application and taking maximum torque that motor for optimization problem.

We can see from the results the best working posture is always when the angle between link-2 and link-3 is 90 degree for any link lengths for Scara robot. This is true because, the objective function will have the maximum value when angle is 90 degree. The solution is very much dependent on how the robot arm structures were designed and what we consider for the joint torque, angular velocities and angular acceleration. The arm structure is the parameter in this system but it is a variable in sub-system-2, and joint angular velocity and acceleration were the variables in subsystem-4.

Since the joint torques is very much dependent on masses of the arm, used neural network model derived from sub-system-3 to calculate mass and integrated with this system. It is difficult to say how dependent our solution are on our worst case consideration. It would be interesting to do in future, by developing better Meta model to calculate mass which increase the accuracy of calculating mass of the link. We can also extend our work to find optimal configurations for some selected points on the trajectory and to other type of robot.

4 Structure & topology Optimization – Zarvan Damania (Subsystem 3):

4.1 Topology optimization

Topology optimization is a mathematical approach that optimizes material layout within a given design space, for a given set of design variables, such that the final design meets the prescribed goal. Topology optimization is used to find the best concept design that meets the design requirements.

The structure of the SCARA robot consists of two main parts: the robot base and the arm. The robot base is stationary. The robot arm consists of a number of moving links forming a kinematic chain connected to the fixed base. The base and link 1 are joined together by a revolute joint, link 1 and link 2 by prismatic joint and link 2and link 3 by a revolute joint. The SCARA robot is equipped with a vacuum cup, which is called the end-effector to perform the prescribed task.

The dimensions of the links were taken from Adept Cobra SCARA robot. The thickness of link 2 is 4 inches and that of link 3 is 3.5 inches and the width of link 2 is 8 inches and of link 3 is 6inches.

The most important characteristic of the robot is the ratio between the weights of the arms to maximum payload. This ratio can only be minimized by reducing the weight of the robot manipulator but this will also result in increased payload capacity. This should be achieved without compromising the static stiffness, the stress or the maximum allowable deflection of the individual linkages. Lighter the robot better its is as less material will be used for building it, it can carry out the work at a fast rate and there will be less stain on the servo motors.

To meet these requirements, design study is taken place in SolidWorks to reduce the weight of the robot.

4.2 Nomenclature

l_2 = Length of Link 2

l_3 = Length of Link 3

T_2 =Thickness of Link 2

T_3 =Thickness of Link 3

IL_2 = Inner cut Length of Link 2

IL_3 = Inner cut Length of Link 3

IW_2 =Inner cut width of Link 2

IW_3 =Inner cut width of Link 3

W_2 =Width of Link 2

W_3 =Width of Link 3

D_2 = Outer Diameter of Link 2

D_3 = Outer Diameter of Link 3

ID_2 = Inner Diameter of Link 2

ID_{31} = Inner Diameter 1 of Link 3

ID_{32} = Inner Diameter 2 of Link 3

H1 =Height of link 1

ID_1 =Inner Diameter of Link 1

OD_1 =Outer Diameter of link 1

4.3 Objective function

The objective function of this subsystem is to minimize the mass of the robot. Lighter the robot better its is as less material will be used for building it, it can carry out the work at a fast rate and there will be less stain on the servo motors.

Problem statement is

$$\min : f = M \text{ (mass)}$$

4.4 Constraints

The main goal is to reduce the mass of the robot but it has to be solved without compromising the stress on the object. The maximum stress it can withstand is the yield strength of the material.

$$\text{S.T.:} \quad g = \text{Stress} < 1e{+}007 \text{ N/m}^2$$

4.5 Parameters and Variables

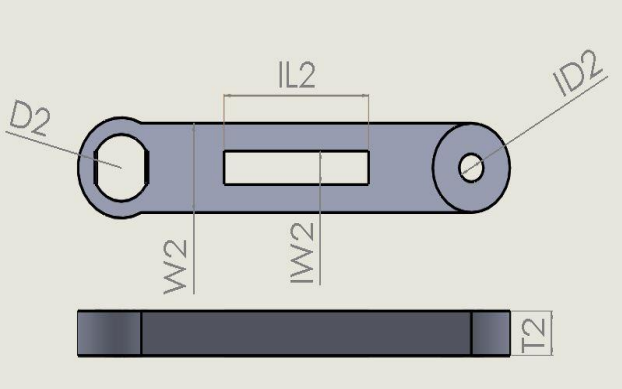

Figure 4.1:- Sketch of link 2

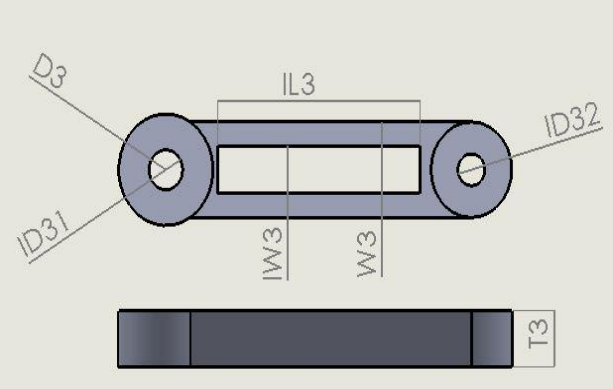

Figure 4.2:- Sketch of link 3

The parameters for this subsystem are:

1. The length(l_2) links 2 = 36.3 in
2. The length(l_2) link 3 = 22.6 in
3. The inner diameter (ID_2)of Link 2 which is given by the shaft of the motor on link 2. = 2.5 in
4. The inner diameter 1(ID_{31}) of Link 3 which is given by the shaft of the motor on link 2. = 2.5 in
5. The inner diameter 2(ID_{32}) of Link 3 which is given by the shaft of the motor on link 3. = 2 in
6. The weight of the Grabber. = 4.4 lbs.
7. The weight of the motor on Link 2. = 10 lbs.
8. The weight of the motor on Link 3. =7.5 lbs.
9. Height of the link 1. = 50in
10. The material of the Robot = Aluminum Alloy with density2699 2699 kg/ m^3 and yield strength 1e+007 N/m^2.

The Variables for this Subsystem are:

1. Thickness of Link 2
2. Thickness of Link 3
3. Inner cut Length of Link 2
4. Inner cut Length of Link 3

5. Inner cut width of Link 2
6. Inner cut width of Link 3
7. Width of Link 2
8. Width of Link 3
9. Outer Diameter of Link 2
10. Outer Diameter of Link 3
11. Inner Diameter of Link 2
12. Inner Diameter 1 of Link 3
13. Inner Diameter 2 of Link 3
14. Inner Diameter of Link1
15. Inner Diameter of Link 1

4.6 Model Analysis

The task of this subsystem is to get an optimized weight for robotic links. The links of the robot are a cantilever beam with a extrude cut in the structure to reduce the mass of the link while keeping the stress on the link minimum.

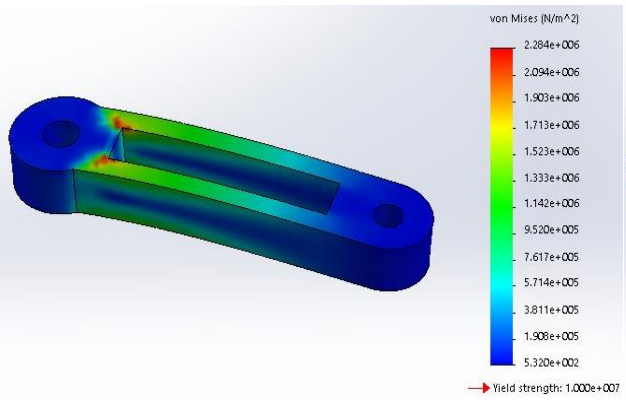

Figure 4.3:- Stress analysis on link 2 before optimization

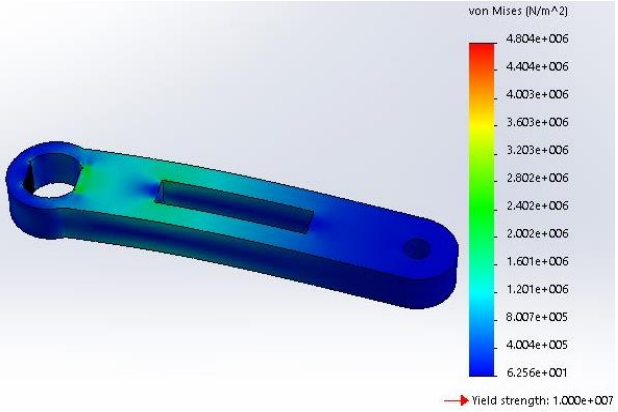

Figure 4.4:- Stress analysis on link 3 before optimization

For the link 3, we can see that the yield strength for the material is 1e+007 N/m2 and the maximum stress on the link is 2.284e+006 N/m2. Since the stress on the link is within the limit of the yield strength we can change the dimensions of the link to reduce the mass. There is very less or no stress acting at the diameter of the joint, so we can reduce the diameter and the thickness at the joint. The reduction in the diameter and the thickness will reduce the mass of the link 3. A

good amount of stress is acting at the edge of the hollow section but it is still within the limit of the yield strength. We can increase the length and the width of the hollow section by a small margin. This small increment will reduce the mass of the link.

For the link 2, we can see that the yield strength for the material is 1e+007 N/m2 and the maximum stress on the link is 4.804e+006 N/m2. Hardly any strain is acting on the link 2 therefore we can we can reduce the diameter and the thickness at the joint and the length and the width of the hollow section up till a good limit.

The change in the dimensions of the Thickness (T), the Width (W) and the Diameter of the joint (D) is proportional to the mass of the link. As these variables reduces the mass of the link will also reduce whereas as the length and width of the extrude cut section increases the mass reduces.

4.7 Neural Network

A neural network (NN) is a mathematical model or computational model that is inspired by the structure and functional aspects of biological neural networks. These models mimic the real life behavior of neurons and the electrical messages they produce between input processing by the brain and the final output from the brain. Neural network is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase. They are usually used to model complex relationships between inputs and outputs or to find patterns in data.

4.7.1 Mathematical Model of ANN

- Inputs –These act like synapses
- Weights- They determine strength of the respective signals.
- Computational Unit –This is responsible for processing all input signals to obtain output.
- Activation function –It controls the amplitude of the output of the neuron.

4.7.2 Training of Artificial Neural Networks

The method of setting the value of the weights enables the process of learning or training. The process of modifying the weights in the connections between network layers with the objective of achieving the expected output is called training and network. The internal process that takes

place when a network is trained is called learning. If the accuracy of the network declines, undo the change and make a different one. It takes time, but the trial and error method does produce results. The task is to mirror the status of the input row onto the output row.

4.7.3 Training algorithm

For the neural network the Levenberg–Marquardt algorithm (LMA) is used and is incorporated into the back-propagation algorithm. The Levenberg–Marquardt algorithm interpolates between the Gauss–Newton algorithm and the method of gradient descent. The authors provided the algorithm with examples of the inputs and outputs that they wanted the network to compute, and then the error is calculated. The training began with random weights, and the goal was to adjust them so that the error was minimal. As the error depends on the weights, the weights were adjusted in order to minimize the error.

The authors discuss Neural Networks implementation for determining the masses of the links. The lengths of the links were taken from the sub-system 2 and their optimal masses were found using design study in SolidWorks. Some values were excluded from training, to be used as fast data. The network was programmed in MATLAB and it is a feed forward network. The Levenberg–Marquardt algorithm was used to train the network.

Two sets of data were trained

1) To find the mass of link 3 when the length of link 3 was given as input. For the preparation of the learning data, 14 sets of length for link 3 were introduced. These 14 sets of data were used to train the neural network. The training started with one hidden layer and 3 neurons. As there was high margin of error the number of neurons were increased to 10.

Table 4.1:- Data used for training the Neural Network for link 3

Sr. No	Input Data	Output Data
	Length of link 3 (in)	Mass of link 3 (lb)
1	30.472	10.79954
2	30.484	10.80563

3	30.657	10.88981
4	30.79528	10.96348
5	31.00787	11.0603
6	31.24803	11.17914
7	31.50787	11.30772
8	31.80315	11.45383
9	32.417	11.75758
10	32.64961	11.87268
11	34.67717	12.87598
12	36.84252	13.94746
13	39.2126	15.12025
14	41.87795	15.77662

2) To find the mass of link 2 when the length of link 2 and the mass of link 3 was given as input. For the preparation of the learning data, The 14 sets of data were used to train the neural network. The length of link 2 and the mass of link 3 are used to train the neural network. The training started with one hidden layer and 3 neurons. As there was high margin of error the number of neurons were increased to 10.

Table 4.2:- Data used for training the Neural Network for link 2

Sr. No	Input Data		Output Data
	Length of link 2 (in)	Mass of link 3 (lb)	Mass of link 2 (lb)
1	32.468	10.79954	30.0089
2	32.539	10.80563	30.2236
3	33.6535	10.88981	30.8852
4	33.80315	10.96348	31.0428

5	34.72835	11.06030	31.84933
6	35.32283	11.17914	34.73243
7	35.618	11.30772	33.8921
8	35.72047	11.45383	32.47753
9	36.7874	11.75758	32.78945
10	37.45669	11.87268	35.1095
11	37.94094	12.87598	32.8977
12	39.33858	13.94746	36.2561
13	41.01969	15.12025	38.1442
14	42.5315	15.77662	38.8571

4.7.4 Training graphs of the neural network

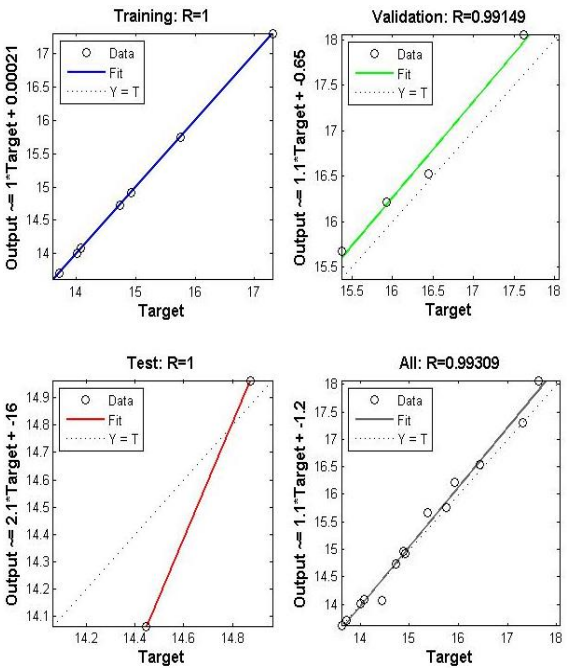

Figure 4.5:- Training graphs for Link-2

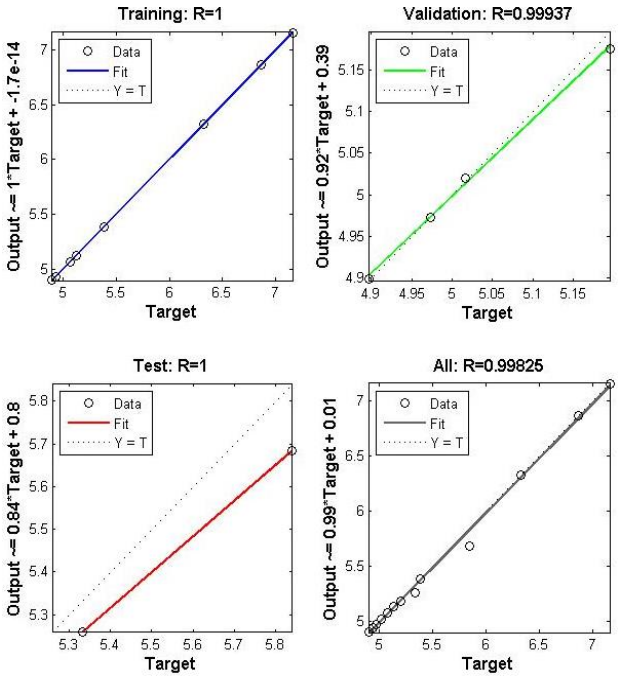

Figure 4.6:- Training graphs for Link 3

From the above graphs it can be seen that the Neural Network has been trained up to 99%. For training of the Neural Network 8 values are taken for Training, 4 for Validation and 2 for Testing.

4.8 Optimization Study

Optimization follows a very simple workflow. First it required a starting point or an initial study to set up loads, fixtures, etc.

The link 3 had the grabber, the motor on link 3 and the pay load as the loads and the shaft of the motor on link 2 was used as the fixture. Whereas for link 2 the motor on link 2, the link 3 along with the grabber the motor on link 3 and the payload was used as the load and the gear joints between the link 1 and link 2 was used as the fixtures.

The next step was to define the Optimization study. The optimization study is defined by goals or objective functions, as well as design variables, and constraints. In this case, the mass the author is minimizing is the objective function, the dimensions are the design variables, and the stress limit is the constraint. After completing the study, optimization visualizes how each change affects the model's performance through SolidWorks like configurations and results plots.

Design optimization in SolidWorks uses a method based on the Design of Experiments. The program offers two different qualities in the properties of the design study. The software runs a number of trials based on the quality level and the number of variables. For each trial, the program runs all the associated simulation studies with a strategically determined set of variable values. The program used in the design study is the Box-Behnken quadratic plan. Depending upon the number of variables it undertakes different number of iterations. In this design study the number of design variables were 5, therefore the number of iteration the study underwent was 41.

4.8.1 Initial study

Initial studies represent the basis for the optimization. During each iteration, the program runs these studies with modified variables. The required initial studies depend on the stress constraint and the goal (minimum mass). The constraints and goal studies belong to the same configuration. After the model was created, we undertook initial studies; the material, load and properties were defined.

4.8.1.1 Initial study of link-3

For the calculation of initial stresses and deformation in the Link 3, the forces acting were calculated.

The load for analysis are:

1) The end effecter
2) The link 2 motor
3) The payload
4) Gravity acting downwards

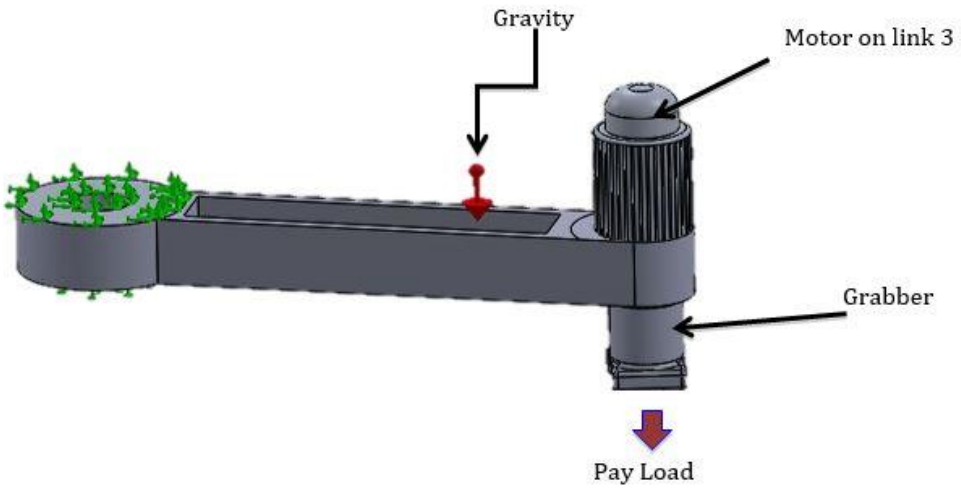

Figure 4.7:- Forces on link-3

The stress analysis was carried out on the link with the end load and the weight of the motor. The results from the stress analysis, was used as a constraint for the optimization of the design. From the analysis we cam to know that the material can be reduced more since the stresses are below the maximum limit.

4.8.1.2 Initial Study of link-2

For the calculation of initial stresses and deformation in the Link 3, the forces acting were calculated.

The load for analysis are:

1) The end effecter
2) The load of Link 2
3) The link 2 motor

4) The link 1 motor
5) The payload
6) Gravity acting downwards

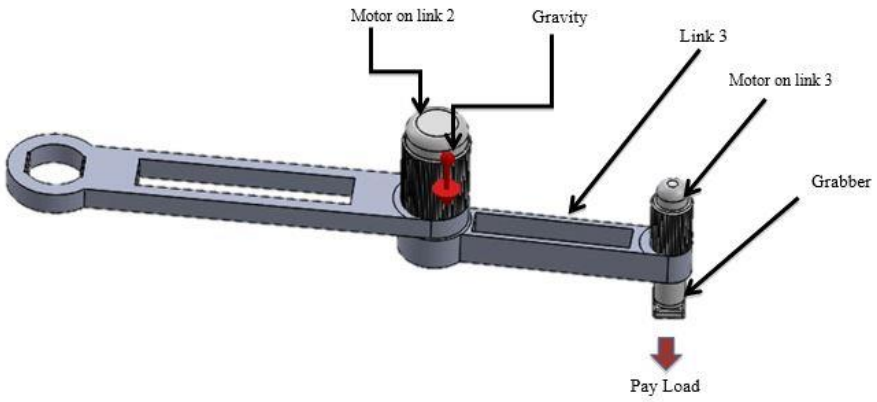

Figure 4.8:- Forces on link-2

The stress analysis was carried out on the link with the end load, the link 3, the weight of the motor 2 and the weight of the motor 1. Since more load is acting on the link it will have lot of stress on acting on it. So eventually the mass of the link 2 will be more than the link 3. The results from the stress analysis, was used as a constraint for the optimization of the design. From the analysis we cam to know that the material can be reduced more since the stresses are below the maximum limit.

4.8.1.3 Initial Study of link-1

The link 1 is the circular beam which holds the link 2 which holds the link 1 and at the end the pay load.

The link 2 is coupled with the link 1 with the help of gears which assists the link to go up or down depending upon the end load requirement. The maximum height of the link one is set to 50 in which is taken from subsystem 1. Because of gravity and the load on the end of the link 2 the link 2 will apply a non-uniform force distribution on the gears of the link 1.

Stress analysis was done on the link 1 to see that it can sustain the forces and the loads by link 2 and link 3, and it does not deform.

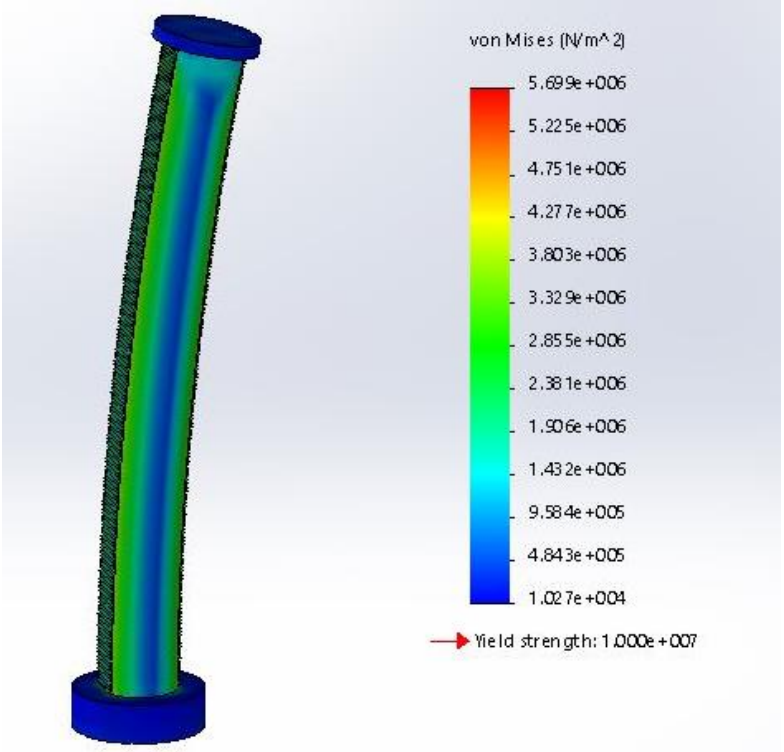

Figure 4.9:- Stress analysis on link 1

4.8.2 Steps for design study

1. Selecting the variables

   The variables are selected from a list of predefined parameters. Continuous variables are defined to perform the optimization. A continuous variable is one, which has any value between the minimum and the maximum values. These values will be exact values that may not be possible to produce due to manufacturing limitations. The initial values of the variables was taken arbitrary and the range was also selected arbitrary.

Table 4.3:- Design variables for link-3

Name	Type	Value	Units
Inner cut length (IL)	**Range**	Min:5 Max:17	in
Thickness (T)	**Range**	Min:0.5 Max:6	in
Width (W)	**Range**	Min:4 Max:8	in
Diameter of the joint (D)	**Range**	Min:5 Max:10	in
Inner cut width (IW)	**Range**	Min:0.5 Max:4	in

Table 4.4:- Design variables for link-2

Name	Type	Value	Units
Inner cut length (IL)	**Range**	Min:5 Max:22	in
Thickness (T)	**Range**	Min:2 Max:7	in
Width (W)	**Range**	Min:6.5 Max:10	in
Diameter of the joint (D)	**Range**	Min:7 Max:12	in
Inner cut width (IW)	**Range**	Min:0.5 Max:4	in

2. Selecting the constraints

Constraints are defined to specify the conditions that the design satisfies. Simulation data sensors are used as the constraint in the Design Study. It monitors simulation results such as stress, strain, and displacement. Stress constraint is defined by defining static study of the links.

Table 4.5:- Constraint for link-3

Sensor name		Bounds	Units	Study name
Stress 3	**is less than**	Max:1e+007	N/m^2	Static 3

Table 4.6:- Constraints for link-2

Sensor name		Bounds	Units	Study name
Stress 2	**is less than**	Max:1e+007	N/m^2	Static 2

3. Selecting the Objective Function

The goal, which is to minimize the mass, is defined to specify the objective function for the Optimization Design Study.

Table 4.7:- Goal for link-3

Name	Goal	Properties	Weight	Study name
Mass 3	Minimize	Mass	14.8 lbs	-

Table 4.8:- Goal for link-2

Name	Goal	Properties	Weight	Study name
Mass 2	Minimize	Mass	39.3 lbs	-

4.8.3 Design optimization of links

For design optimization a CAD model of the SCARA robot arms were made in SolidWorks using the dimensions taken from Adept Cobra SCARA robot. The lengths of the links were given from sub-system 2 for the purpose of optimizing its weight.

CURRENT SPECIFICATIONS OF ROBOT:
Manipulator Weights –
Link 1 - 49.8 lbs.
Link 2 - 39.3 lbs.
Link 3 - 14.8 lbs.
Total link weight - 140 lbs

The material for the optimization was set for Aluminum alloy.
Young's Modulus E = 1e+007 N/m^2
Poisson's Ratio = 0.33
Density = 2699 kg/m^2

The analysis is based on the finite element method and consists of completion of the design model using the dimensional data as design variables. Optimized design for the structures of the SCARA robot have to meet certain criteria regarding dimensional design, shape and material consumption. To improve the static and dynamic behavior of the SCARA robot structure, the following requirements must be accomplished:

- Minimum weight
- Maximum static stiffness
- Minimum displacement

4.9 Parametric Study

4.9.1 Effect of link length-3 on mass

The graph shows the changes in the mass of the link 3 with changes in the design variables for 41 iterations

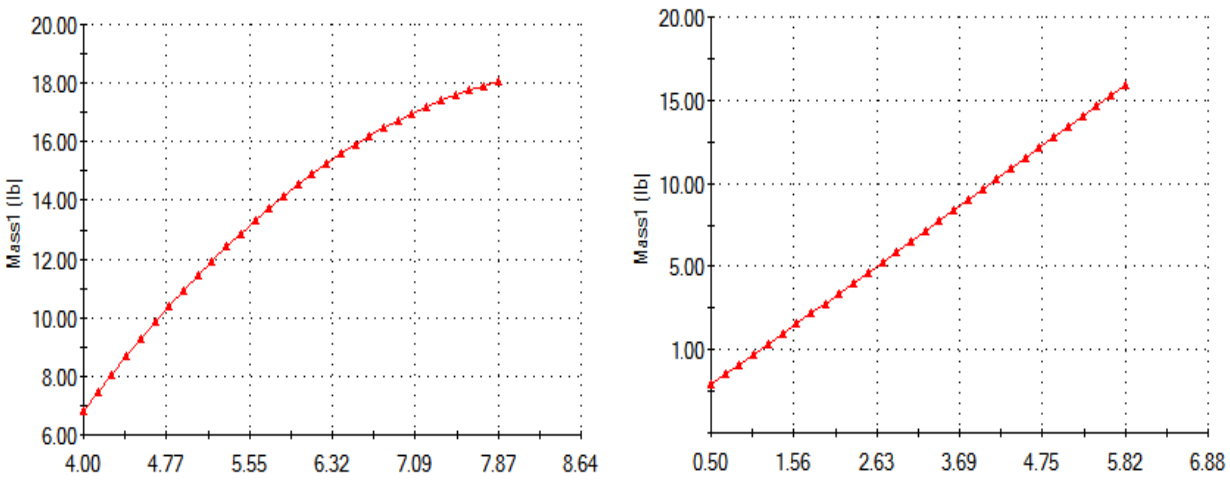

Figure 4.10:- Change in mass as the design variables increases or decreases for link 3

From the above graphs we can see that as the width and the thickness of the link increases the mass of the link also increases.

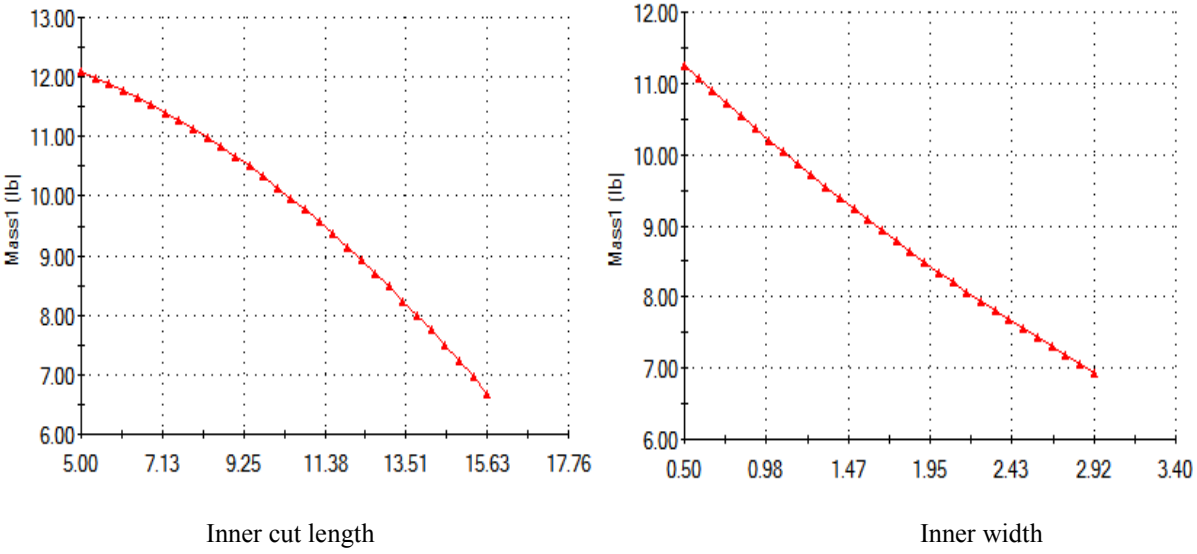

Figure 4.11:- Change in mass as the design variables increases or decreases for link 3

From the above graphs we can see that as the width and the inner width and the inner cut length of the link decreases the mass of the link also decreases.

4.9.2 *Effect of link length-2 on mass*

The graph shows the changes in the mass of the link 2 with changes in the design variables for 41 iterations.

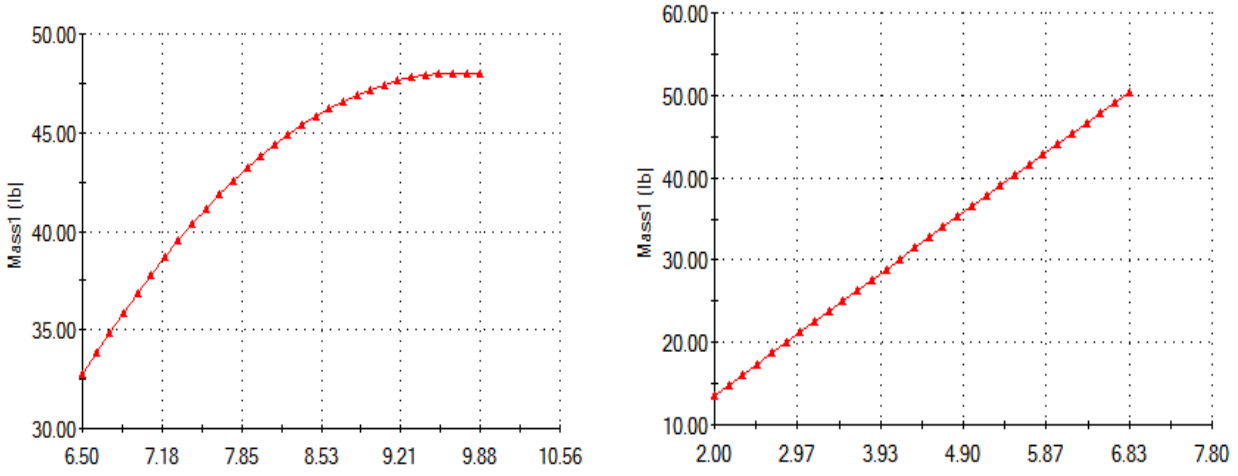

Figure 4.12:- Change in mass as the design variables increases or decreases for link 2

From the above graphs we can see that as the width and the thickness of the link increases the mass of the link also increases.

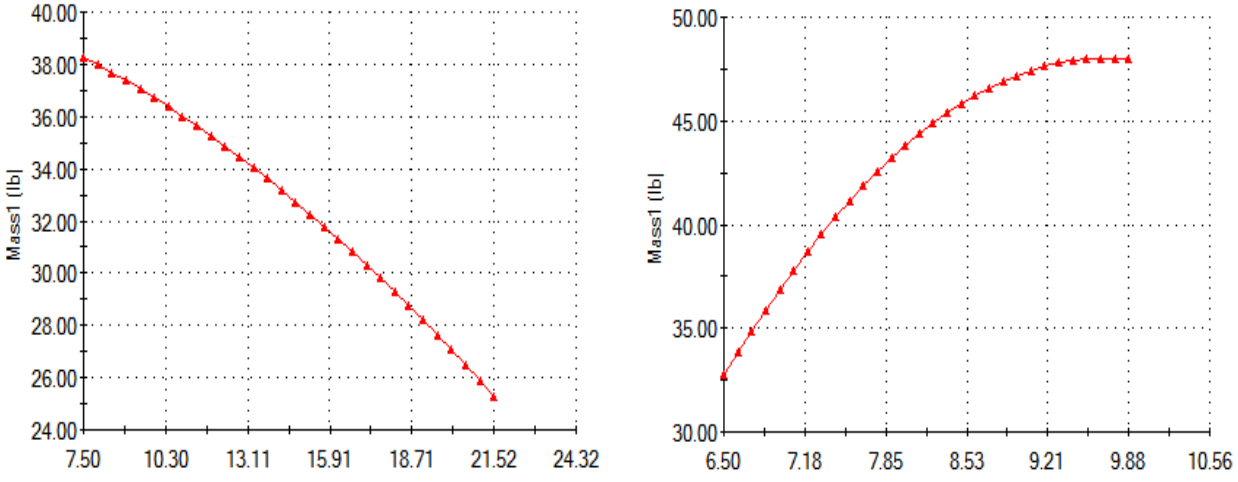

Figure 4.13:- Change in mass as the design variables increases or decreases for link 2

From the above graphs we can see that as the width and the inner width and the inner cut length of the link decreases the mass of the link also decreases.

4.9.3 *Parametric study for the payload against the mass of the links*

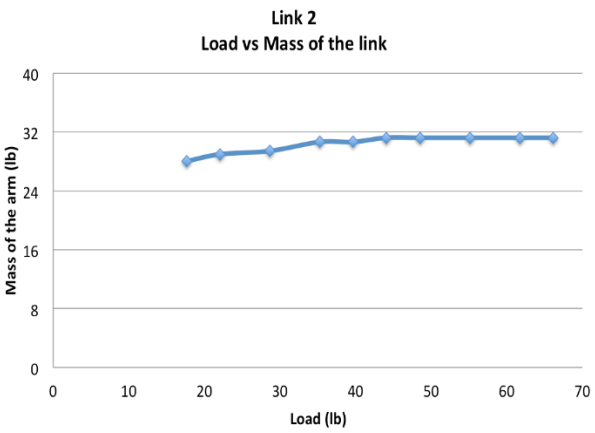

Figure 4.14: Payload vs the mass of the link 2

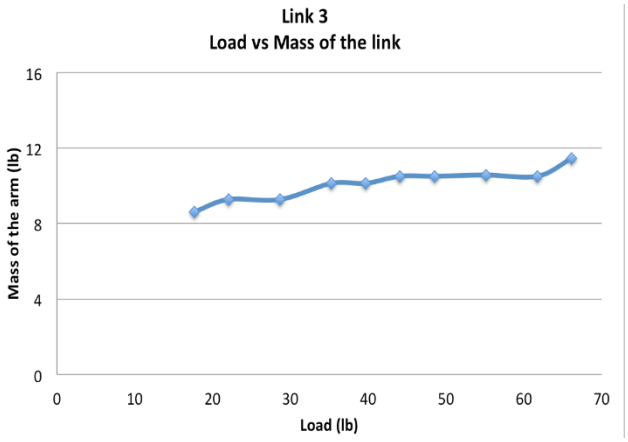

Figure 4.5: Payload vs the mass of the link 3

From the Graphs we can see that as the weight of the payload increases the mass of the link also increases. This is due to stress acting on the link. The links of the robot are a cantilever beam with one end fixed and the load at the other end. As the payload on the link 3 increases the link will start bending and the stress on the link will begin to increase. The stress on the link cannot exceed the material's yield strength. To compensate that, the thickness and the width of the link will increase correspondingly increasing the mass of the link 3.

For the case of link 2, the end which is not fixed has to sustain the weight of the payload as well as the weight of the link 3. There for to sustain the stress due to the above weights the mass of the link 2 is higher than the mass of link 3. We can also see from the above graphs as the end load on the link 2 increases its mass increases.

4.10 Discussion of Results

During the optimization of the links out of 41 iterations 8 iterations failed, as the stress constraint was greater than the yield strength of the material, so these iterations were discarded. Out of the 33 iterations only one iteration met the requirements with minimum weight and required stress.

From the results you can see that the optimal mass is less than the initial mass of the links. In this study we used a solid bar with an extrude cut in the middle of the link. In the future we can use truss geometry to further decrease the mass of the arms by keeping the stress minimum.

The lengths of the links were the variables of subsystem 2 but the parameters for this subsystem. The length of the links are dependent on the mass of the link and vice-versa. To find the length of the links in subsystem 2 arbitrary lengths were taken and given to subsystem 3 and their optimized masse was found in Solidworks. Then a neural network was trained to find out lengths for any given mass of the link. This data was then given to subsystem 2 to find out the optimal length of the link 2 and link3. These optimal lengths were given to the subsystem 3 to optimize the mass of the links.

Optimized Design for Link 3

Table 4.9:- Result for the optimized mass of link 3

Component name	Units	Initial	Optimal
Inner cut length (IL)	in	12	15
Thickness (T)	in	3.5	3.2
Width (W)	in	6	4
Diameter of the joint (D)	in	7	6.8
Inner cut width (IW)	in	3	2.8
Stress	N/m^2	2.284e+006	4.8042e+006
Mass	lb	14.8	8.3

Optimized Design for Link 2

Table 4.10:- Result for the optimized mass of link 2

Component name	Units	Initial	Optimal
Inner cut length (IL)	in	7.5	14.75
Thickness (T)	in	7	4.5
Width (W)	in	8.25	6.5
Diameter of the joint (D)	in	10	9.5
Inner cut width (IW)	in	2.25	4
Stress	N/m^2	3.7508e+006	3.3082e+006
Mass	lb	39.3	33.2

5 Trajectory/Control Optimization - Erica Neuperger (Subsystem 4)

5.1 Problem

The last subsystem of this project is optimizing the trajectory of the robot. In industrial applications it is extremely important for a robot to perform its task as quickly as possible, in order to maximize the amount of money earned in a given time. However, it is also important for a company to conserve as much energy as they can. There is obviously a trade-off between these two properties. The faster the robot moves, the more boxes it can stack, but the more energy it will use. We are interested in optimizing the trajectory of the robot by maximizing the profit of this stacking process.

For this portion of optimization, the general configuration of the robot will be assumed. In other words, the link lengths, masses, and their strengths are known. Because this study does not include motor selection optimization, we will assume that motors with appropriate maximum torques and 100% efficient are implemented in the design. Also, to simplify the problem, only the two rotational joints will be considered. The trajectory of the third rotational joint will not be studied, since it is only responsible for the box's orientation and it does not affect the system's dynamics. Also, the prismatic joint's dynamics and trajectory are not considered here in this study, as adding a 3rd dimension greatly complicates the problem.

5.2 Nomenclature

Joint angles: θ_2, θ_3

Joint angular velocities: $\dot{\theta_2}, \dot{\theta_3}$

Joint angular accelerations: $\ddot{\theta_2}, \ddot{\theta_3}$

Joint torques: τ_2, τ_3

Link lengths: L_2, L_3

Masses of links: m_2, m_3

Time: t

Time to complete movement: t_f

Total time the manipulator is able to stack boxes: t_{total}

Mass of end effector: m_e

Moment of inertia of the links: I_2, I_3

Center of mass of the links: L_{g2}, L_{g3}

Max torque allowed for joint: τ_{max_i}

Position of the end effector: (x_e, y_e, z_e)

Velocity of the end effector: $(\dot{x_e}, \dot{y}, \dot{z_e})$

Acceleration of the end effector: $(\ddot{x_e}, \ddot{y_e}, \ddot{z_e})$

Mass of box: m_b

Energy: E

Amount of money made per box stacked: P

Power's cost per joule of energy: p_w

Work: W

5.3 Mathematical Model

5.3.1 Objective Function:

The objective of this study is to maximize profit. To do this, a cost function is obtained, which is dependent on the time it takes to stack a box (t_f) to as well as the energy required to stack the box (E).

The total amount of money made by stacking all of the boxes in the allotted amount of time is equated by:

$$\frac{P * t_{total}}{t_f * 2}$$

which assumes that the path that the robot takes to return to its starting position is the exact opposite as the one it took to get there. In other words this equation assumes that the manipulator takes the same amount of time and energy to return back to its initial position as it did to get there.

The amount of energy required to stack each box should then be calculated. This can be done using the Lagrangian dynamics equations. The figure below shows the manipulator's setup and variables.

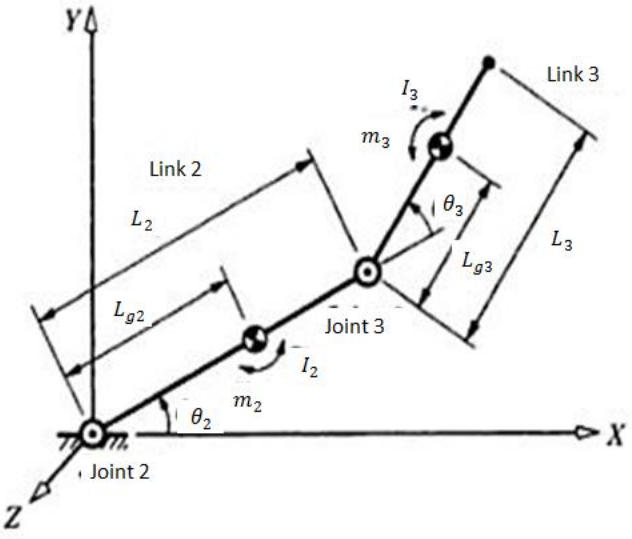

Figure 5.1:- Setup of the manipulator for deriving the Lagrangian dynamics equations

Here, L_{gi} represents the distance from the joint to the center of mass of the arm and I_i represents the moment of inertia of the link about its center of mass. These values can be determined by the following equations, where the weight of the links is assumed to be a point mass at the center of the links and the weight of the motor, box, and end effector is assumed to be a point mass at the end of the appropriate link:

$$L_{g2} = \frac{\left(\frac{m_2 * L_2}{2} + mm2 * L_2\right)}{(mm2 + m_2)}$$

$$L_{g3} = \frac{\left(m_3 * \frac{L_3}{2} + (m_b + m_e)L_3\right)}{(m_b + m_e + m_3)}$$

$$I_2 = \left(L_2 - L_{g2}\right)^2 * mm2 + \frac{1}{12}(m_2)(L_2)^2 + m_2 * \left(\frac{1}{2}L_2 - L_{g2}\right)^2$$

$$I_3 = \left(L_3 - L_{g3}\right)^2 * (m_e + m_b) + \frac{1}{12}m_3 L_3^2 + m_3\left(\frac{1}{2}L_3 - L_{g3}\right)^2$$

Then, the kinetic energy of link 2 and link 3 can be obtained using the following formulas.

$$K_2 = \frac{1}{2}m_2 L_{g2}^2 \dot{\theta_2}^2 + \frac{1}{2}I_2 \dot{\theta_2}^2$$

$$K_3 = \frac{1}{2}m_3\left(L_2^2\dot{\theta_2}^2 + L_{g3}^2\left(\dot{\theta_2} + \dot{\theta_3}\right)^2 + 2L_2 L_{g3} C_3(\dot{\theta_2}^2 + \dot{\theta_2}\dot{\theta_3})\right) + \frac{1}{2}I_3\left(\dot{\theta_2} + \dot{\theta_3}\right)^2$$

Because the potential energy for both links is zero, L can be calculated by $L = K_2 + K_3$ and substituted into the Lagrangian equation for Q_{ib},

$$Q_{ib} = \frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} + \frac{\partial P}{\partial q_i}$$

thus resulting in equations for the torques:

$$\tau_2 = \left[m_2 L_{g2}^2 + I_2 + m_3\left(L_2^2 + L_{g3}^2 + 2L_2 L_{g3} C_3\right) + I_3\right]\ddot{\theta}_2 + \left[m_3\left(L_{g3}^2 + L_2 L_{g3} C_3\right) + I_3\right]\ddot{\theta}_3 - m_2 l_1 l_{g2} S_2 (2\dot{\theta}_1\dot{\theta}_2 + \dot{\theta}_2^{\,2})$$

$$\tau_3 = \left[m_3\left(L_{g3}^2 + L_2 L_{g3} C_3\right) + I_3\right]\ddot{\theta}_2 + \left(m_3 L_{g3}^2 + I_3\right)\ddot{\theta}_3 + m_3 L_2 L_{g3} S_3 \dot{\theta}_2^2$$

Finally, an equation representing the work done to rotate the two joints to the desired position is obtained by:

$$W = \int_0^{t_f} \sum_{i=1}^{2} \tau_i\,(t)\dot{\theta}_i(t)\ dt$$

In the case that $\tau_i(t) * w_i(t)$ is negative, the value will be taken to be zero. This accounts for when the motors are breaking.

This can all be combined to obtain a final equation for profit in terms of θ_i and its derivatives, where the first term represents the amount of money made by stacking all the boxes and the second term represents the cost of energy of stacking those boxes.

$$maximize\ F_{\theta_2(t),\theta_3(t),t_f} = \frac{P * t_{total}}{t_f * 2} - \frac{t_{total}}{t_f} * p_w \int_0^{t_f} \sum_{i=1}^{2} \max(\tau_i(t)\,\dot{\theta}_i(t),0)\ dt$$

Note that the objective is to maximize profit by manipulating the joint angles with time as well as the time it takes to place a box on the pallet. The trajectories $\theta_2(t)$ and $\theta_3(t)$will be defined by an n^{th} order polynomial, given as:

$$\theta_2(t) = a_1 + a_2 t + a_3 t^2 + \cdots + a_n t^{n-1}$$

$$\theta_3(t) = b_1 + b_2 t + b_3 t^2 + \cdots + b_n t^{n-1}$$

Thus the objective function can be re written as:

$$maximize\ F_{a_1,a_2 \ldots a_n,b_1,b_2 \ldots b_n,t_f} = \frac{P * t_{total}}{t_f * 2} - \frac{t_{total}}{t_f} * p_w \int_0^{t_f} \sum_{i=1}^{2} \max(\tau_i(t)\,\dot{\theta}_i(t),0)\ dt$$

5.3.2 *Constraints:*

The arm starts at a specified location with a velocity of 0.

$$h_1 = \theta_2(0) = 31.96°$$
$$h_2 = \theta_3(0) = -90.03°$$
$$h_3 = \dot{\theta}_2(0) = 0°/sec$$
$$h_4 = \dot{\theta}_3(0) = 0°/sec$$

The arm ends at a specified location with a velocity of 0. These values correspond to an end effector final position of (0in, 49in), obtained from the pallet space optimization.

$$h_5 = \theta_2(t_f) = 130.7°$$
$$h_6 = \theta_3(t_f) = -69.0°$$
$$h_7 = \dot{\theta}_2(t_f) = 0°/sec$$
$$h_8 = \dot{\theta}_3(t_f) = 0°/sec$$

The arm's joint torques should be limited in order to prevent structural failure and meet the motor specifications.

$$g_1 = \tau_2(t) \leq 80Nm$$
$$g_2 = \tau_3(t) \leq 10Nm$$

The box's acceleration should be limited to prevent the items in the box from breaking.

$$g_3 = \sqrt{\ddot{x}_e(t)^2 + \ddot{y}_e(t)^2} \leq 4m/s^2$$

5.3.3 *Design Variables and Parameters:*

Variables:

Joint angles: $\theta_2(a_1, a_2, a_3, \dots n), \theta_3(b_1, b_2, b_3 \dots b_n)$

Time to complete movement: t_f

Parameters:

Link lengths: $L_2 = 0.9227m,\ L_3 = 0.5760m$

Masses of links: $m_2 = 14.414kg,\ m_3 = 4.887kg$

Mass of end effector: $m_e = 2kg$

Max torque allowed for joint: $\tau max_2 = 80Nm\ \ \tau max_3 = 10Nm$

Mass of box: $m_b = 20kg$

Total time the manipulator is able to stack boxes: $t_{total} = 1\ hour$

Position of box on the conveyer: (X_0, Y_0) = (1.15,0)

Desired location of box: $(X_f, Y_f) = (-0.35, 1.15)$

Amount of money made per box stacked: $P = \$0.005$

Price of energy per Joule: $p_w = \$3.33 * 10^{-8}$

Order of the polynomial: n = 15

5.3.4 Summary Model:

The objective will be to minimize:

$$F_{a_1,a_2 \ldots a_n,b_1,b_2 \ldots b_n,t_f} = -\frac{P * t_{total}}{t_f * 2} + \frac{t_{total}}{t_f} p_w \int_0^{t_f} \sum_{i=1}^{2} \max(\tau_i(t)\, w_i(t), 0)\ dt$$

With the constraints:

$g_1 = \tau_2(t) - 80 \leq 0$

$g_2 = \tau_3(t) - 10 \leq 0$

$g_3 = \sqrt{\ddot{x}_e(t)^2 + \ddot{y}_e(t)^2} - 20 \leq 0$

$h_1 = \theta_2(0) = 31.97°$

$h_2 = \theta_3(0) = -90.03°$

$h_3 = \dot{\theta}_2(0) = 0°/sec$

$h_4 = \dot{\theta}_3(0) = 0°/sec$

$h_5 = \theta_2(t_f) = a_1 + a_2 t_f + \cdots a_n {t_f}^{n-1} = 130.7°$

$h_6 = \theta_3(t_f) = b_1 + b_2 t_f + \cdots b_n {t_f}^{n-1} = -69°$

$h_7 = \dot{\theta}_2(t_f) = a_2 + 2a_3 t_f + \cdots (n-1) a_n t_f^{n-2} = 0°/sec$

$h_8 = \dot{\theta}_3(t_f) = b_2 + 2b_3 t_f + \cdots (n-1) b_n t_f^{n-2} = 0°/sec$

5.4 Model Analysis

Due to the nonlinear nature of the problem, monotonicity analysis cannot be applied. However, 4 of the variables can immediately be solved for from the equality constraints h_1 through h_4.

$$h_1 => a_1 = 31.96 * \frac{\pi}{180} = 0.5578$$

$$h_2 => b_1 = -90.03 * \frac{\pi}{180} = -1.571$$

$$h_3 => a_2 = 0$$

$$h_4 => b_2 = 0$$

5.5 Optimization Study

The trajectories are calculated at 100 different points in time. Note that in reality, the motor commands sent would be updated more frequently at around 400Hz, which would correspond to about 800 points in time. This would make the optimization problem more expensive computationally and is not necessary for calculating the polynomial.

Because the nature of the problem is both large scale and nonlinear, AMPL was used in order to find a solution. AMPL is able to perform automatic differentiation to obtain gradient and hessian information. The KNITRO solver was then utilized, as it is able to deal with nonlinear problems that have a large number of inequality constraints. The code to do this is included in the appendix and the results are given below.

Initially a 5^{th} order polynomial was used to calculate joint angles over time. This resulted in the following graphs for the trajectories.

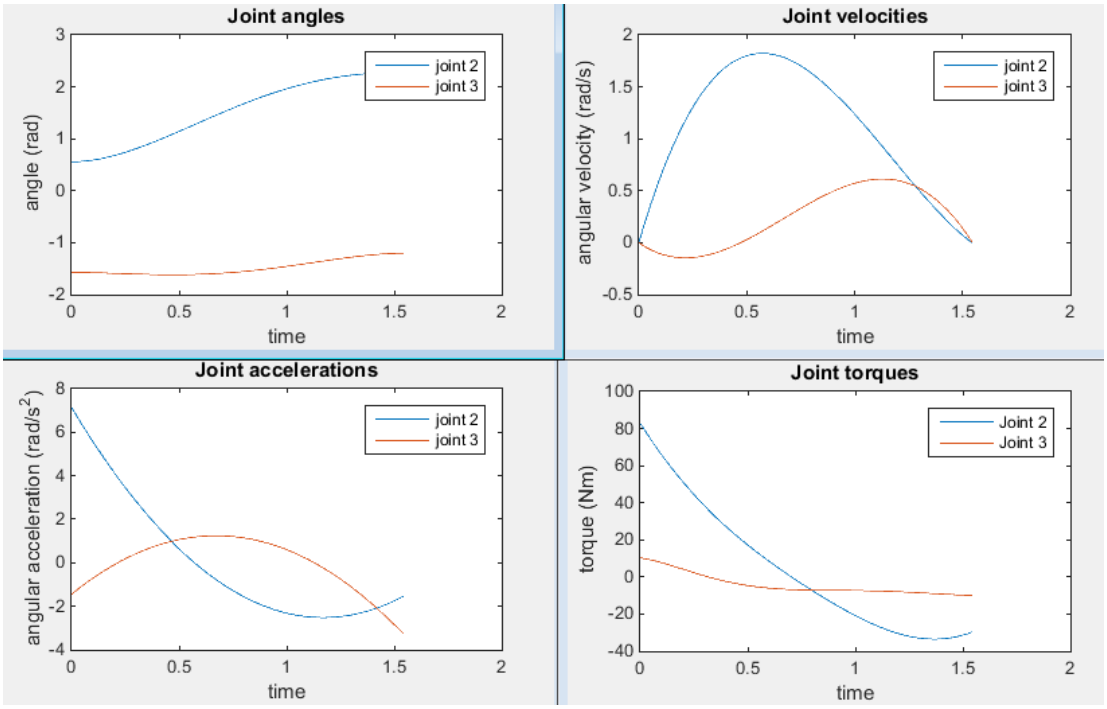

Figure 5.2:- Optimized 5th order trajectory

However, the figure below shows that using a polynomial of higher order significantly increases the profit. Using a 15th order polynomial as opposed to a 5th order polynomial increases the total profit by $1.04.

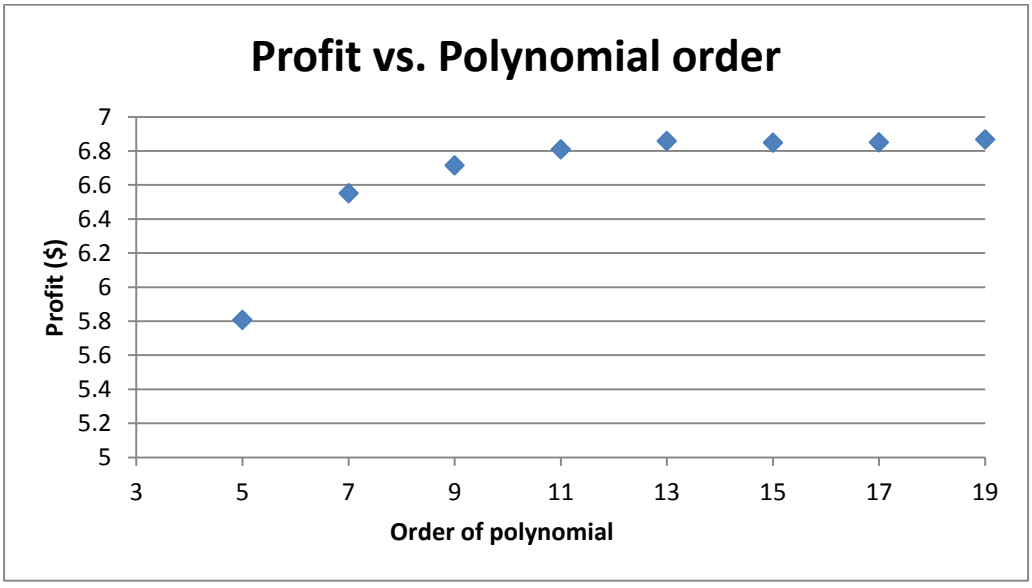

Figure 5.3:- Profit vs polynomial order

Also, note that as the polynomial order is increased, the total amount of time to pick and place a box decreases. This is seen in the figure below.

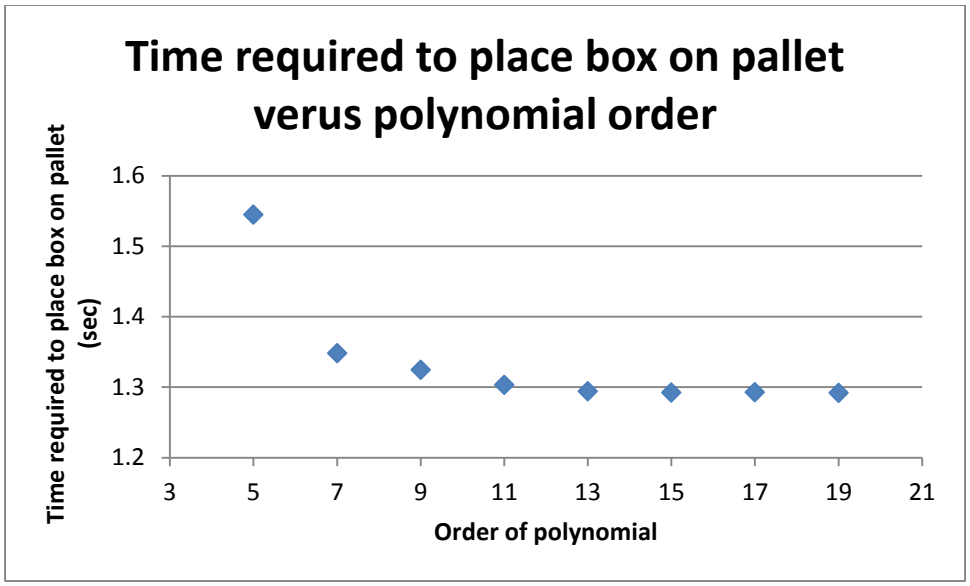

Figure 5.4: -Time required to place box on the pallet vs polynomial order

Because of the large increase in profit, a 15th order polynomial is used for the rest of the optimization study. A higher order polynomial is not considered, because although the profit will slightly increase, it is much more computationally expensive to find.

The trajectory results for this 15th order polynomial can be seen in the figure below.

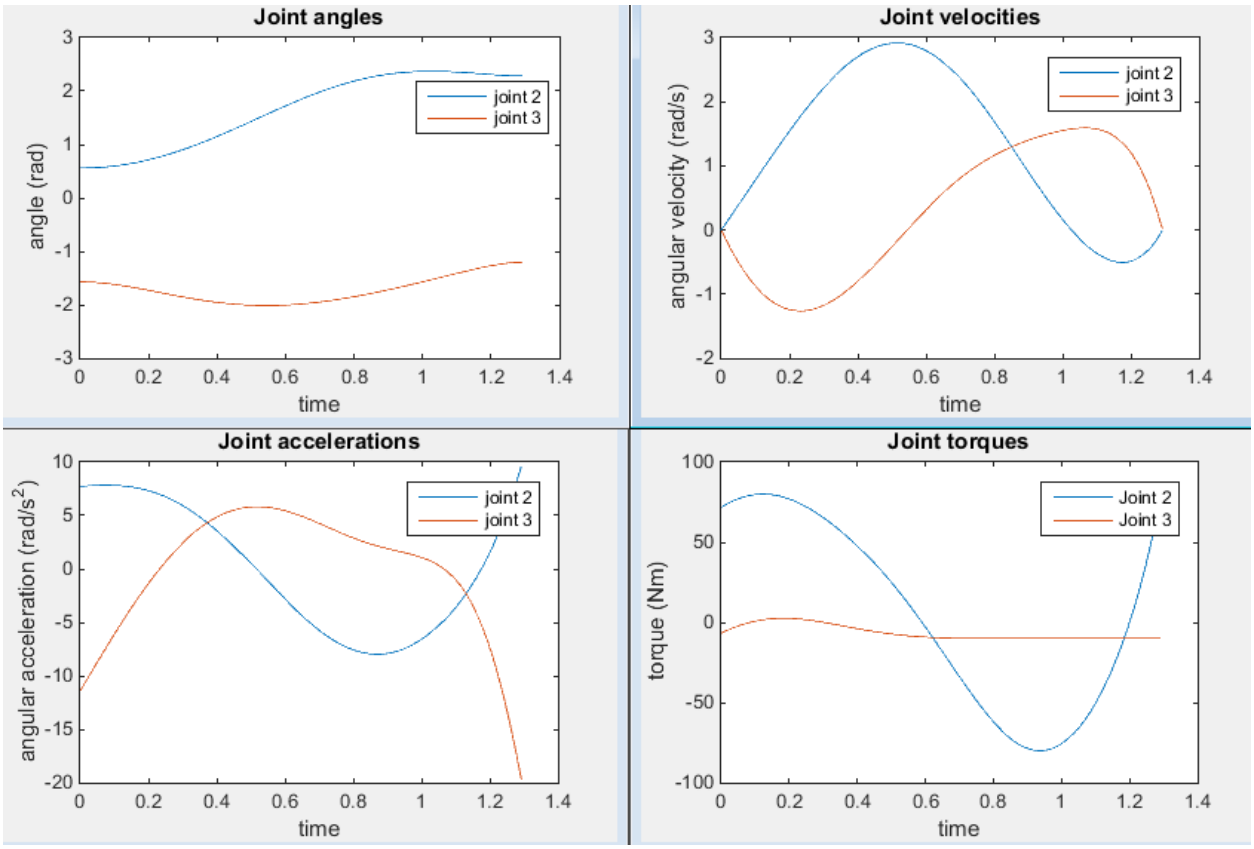

Figure 5.5:- Optimized 15th order trajectory

Note that the inequality constraints for both joints are active, as the magnitude of joint 2 is limited by 80Nm and the magnitude of joint 3 is limited by 10Nm. Also, the figure below shows the end effector acceleration over time. This constraint is inactive, as it does not come close to the $20\frac{m}{s^2}$ limit.

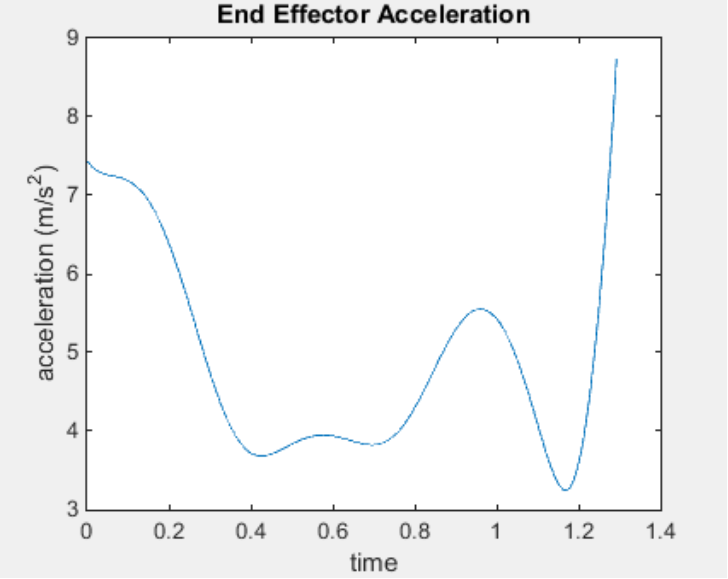

Figure 5.6: - End effector acceleration

The inactive and active constraints can also be determined by examining the Lagrange multipliers. The following table lists the largest Lagrange multiplier associated with each inequality constraint.

Table 5.1:- Lagrange multiplier values

Associated Inequality Constraint	Max Lagrange multiplier
$\boldsymbol{g_1}$: max torque joint 2 (80Nm)	0.168
$\boldsymbol{g_2}$: max torque joint 3 (10Nm)	1.34
$\boldsymbol{g_3}$: max end effector acceleration	0

The Lagrange multipliers indicate that increasing the allowable torque of joint 3 would lead the greatest increase in profit.

A sensitivity analysis is performed, which examines the effect of increasing the torque limits of both joints (see the figure below). Here a higher slope is seen along axis associated with the 3rd joint's maximum torque. Also, it should be noted that even when the maximum torques of joints 2 and 3 are increased to 14 and 84Nm respectively, the inequality constraint for the maximum end effector acceleration still remains inactive. Thus, it is not necessary to perform a sensitivity analysis on the maximum allowable end effector acceleration.

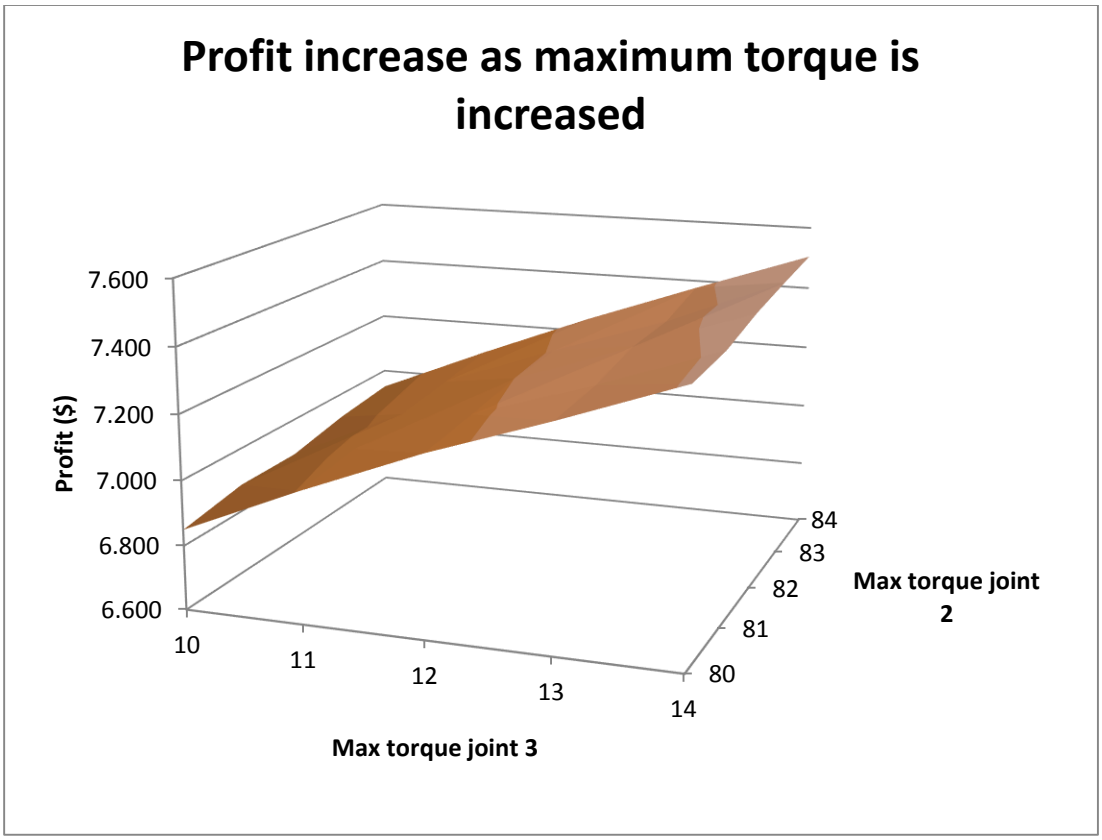

Figure 5.7:- Profit increase as maximum torque is increased

In order to check whether this 15th trajectory was hitting a local solution, different initial conditions were used in order to see if the solution remained the same. KNITRO allows the user to specify a desired number of initial conditions. Then, a built in tool randomly generates the desired number of initial conditions. These initial conditions are used to calculate an optimal solution and the "best" result is reported. This option was implemented, using 100 different initial conditions, and the same result was obtained. Thus, the problem is considered to be optimized.

5.6 Parametric Study

Considering the fact that the cost of energy constantly varies, a parametric study is performed to see how profit is affected by energy costs. The results are shown below.

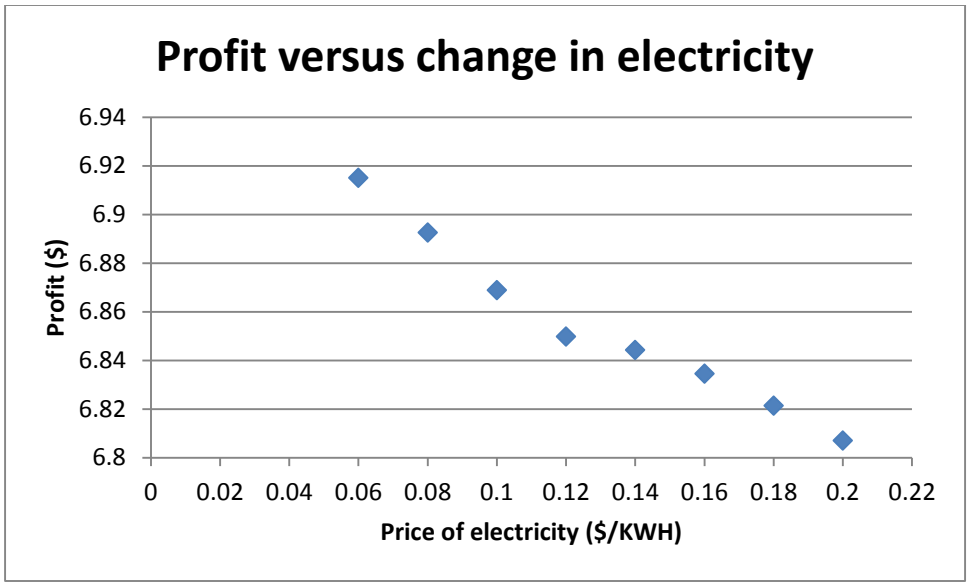

Figure 5.8:- Profit vs change in electricity

As expected, as the price of energy goes up, the amount of profit decreases. The general shape of the trajectory graphs remains similar for all of the prices by inspection; however, the graph below demonstrates that the trajectory graphs do in fact change. The total time required to place a box on the pallet increases with the increasing cost of electricity.

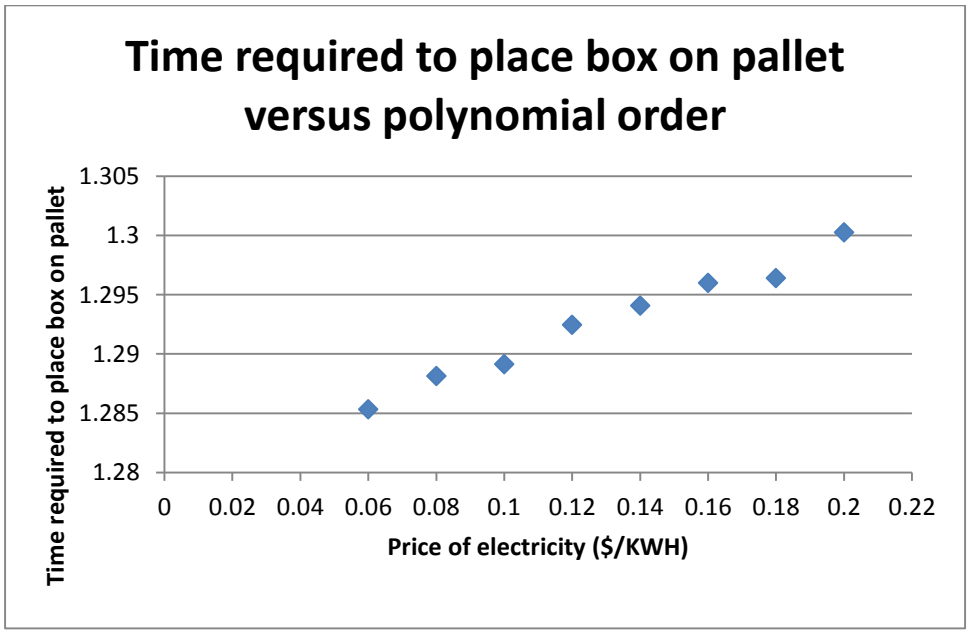

Figure 5.9:- Time required to place the box on pallet vs polynomial order

A similar analysis is performed on how the amount of money made per package affects the profit and time required to place a box on the pallet. These results can be seen below.

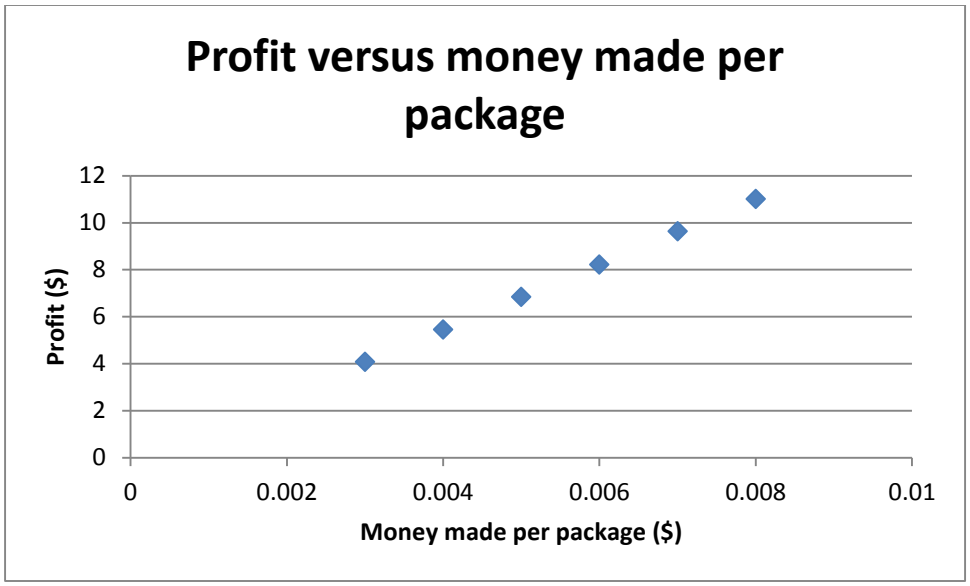

Figure 5.10:- Profit vs money made per package

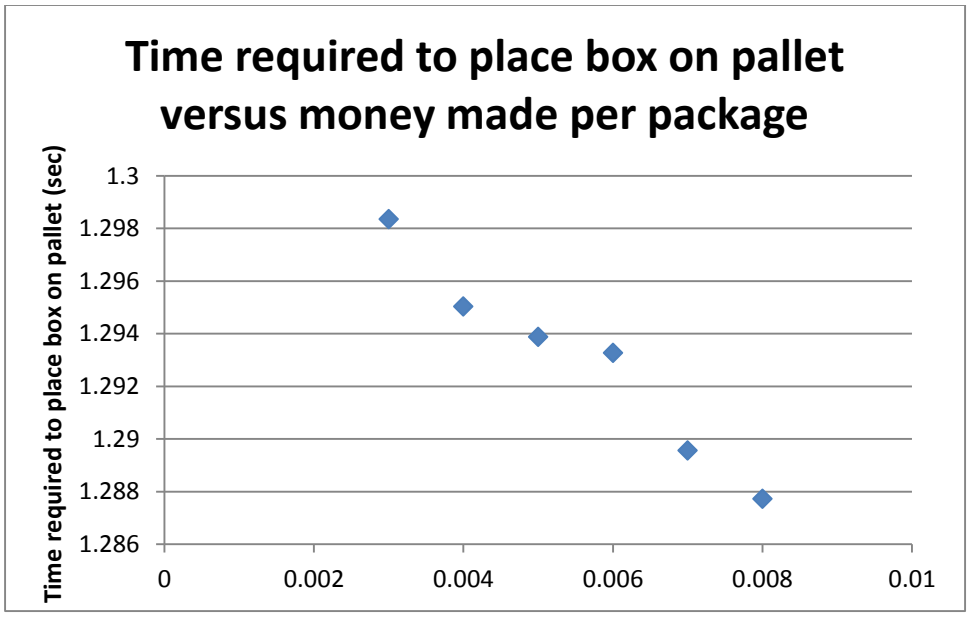

Figure 5.11:- Time required to place the box on pallet vs money made per package

As the amount of money made per package increases, the profit linearly increases and the time required to place a box on the pallet decreases. These two parameter studies are useful because they show how important it is for each parameter value to be accurate and up to date. In application it would be beneficial to have these values continuous update throughout the work in order to obtain the best trajectories.

5.7 Discussion of Results

First off, the figures above show that there is a significant difference between the 5^{th} order and 15^{th} order trajectory. Increasing the polynomial order led to an increase in profit of $1.04 per hour. Annually this adds up to just over $3000. This is significant since most robotic trajectories only use a fifth order polynomial in order to complete tasks. A fifth order polynomial allows the user to specify initial and final position and velocity, which is good enough to prevent manipulator damage, but does it not allow the user to take advantage of the of the potential amount of money that could be saved. The results here show that a higher order trajectory model increases profits dramatically in an industrial setting, without have to make any changes to the existing robot hardware. This by far is the most important parameter here, as its only "cost" is the computational one.

Also, the results show that the optimal joint trajectory does not move directly from the initial joint position to the final joint position. Rather, during some parts of the trajectory, the position of the arm moves away from its final position before heading towards it. This phenomenon is present in both of the joint trajectories, but it is clearer for the 3^{rd} joint. In the graph of the 15^{th} order trajectory, the position of the 3^{rd} arm initially moves away from its final position instead of directly to it. This actually reduces the both the torque and the inertia term, thereby reducing the amount of work done and the total cost of energy. Although moving away from the final position costs that particular joint more energy, the other joint is able to save more energy than the extra energy that was used. This is also be seen near the end of the trajectory of the 2^{nd} joint. The optimal trajectory result was very different from what was expected, as one would expect the joint to move directly to its final position. This was the most surprising result of this study.

Finally, the sensitivity analysis and lagrange multiplier study showed that increasing the maximum allowable torque of joint 3 would yield in the greatest amount of profit increase.

In the future, more work should be done to implement what was learned here into code that would give optimal trajectories for each and every box. A model predictive controller may prove of use here.

6 System integration

Once the sub-system optimization is done next step is to integrate each and every subsystem. This is done in multiple steps.

1. We used the values of pallet dimension from sub-system-1(packaging space) to find the optimal configuration of the robotic arm in sub-system-2 (configuration).
2. Since the subsystem-3(topology) is integrated with sub-system-2 (configuration), mass is calculated for each iteration. But once we got the optimal values for the link lengths, Solidworks was used to check the output of mass from Matlab.
3. Then the link lengths, masses and X and Y coordinate from subsystem-1 were used in sub-system-4(trajectory) to find the optimal path for placing the boxes.
4. These three processes were repeated until the output of sub-system-4 (trajectory) satisfied the values assumed in sub-system-2 (configuration).

After performing the procedure above, the following results were obtained.

Table of results

Table 6.1:- Solution matrix for Box set 1

Width	Height	X-coordinate	Y-coordinate
30	1	0	0
10	29	30	0
30	29	0	1
5	30	29	28
29	20	0	29
29	10	0	49
5	14	35	28

Table 6.2:- Solution matrix for Box set 1

Width	Height	X-coordinate	Y-coordinate
30	15	0	0
10	18	30	0
29	5	0	18
11	14	29	18
28	5	0	20
28	24	0	25
5	20	29	32
6	17	34	32

Table 6.3:- Solution matrix for Box set 1

Width	Height	X-coordinate	Y-coordinate
30	12	0	0
4	25	30	0
6	24	34	0
30	11	0	12
28	20	0	24
1	6	28	24
1	4	29	24
6	25	34	25

Table 6.4:- Specification of manipulator

Link length l2	36.3 in
link length l3	22.6 in
mass of link 2	33.2 lb
mass of link 3	8.3 lb
max angular velocity of link-2	2.8rad/s
max angular velocity of link -3	1.5rad/s
max angular acceleration of link-2	9rad/s^2
max angular acceleration of link-3	19.5rad/s^2

Results and discussion

As discussed in the introduction each sub-system is dependent on the output of the other sub-systems. With the pallet dimension from the sub-system-1 (packaging space) and integration of sub-system-3 (topology) with sub-system-2 (configuration), we did a study on the remaining sub-system (trajectory). Once the maximum accelerations and velocities were obtained from the trajectory analysis, they needed to be compared to the maximum values that were assumed in the configuration optimization. If the acceleration and velocity corresponding to maximum torque from the trajectory optimization exceeded the assumed values, the configuration optimization was updated with the new maximum values. New lengths and masses were then calculated with these values to use in the trajectory optimization. The process was repeated until the acceleration and velocity corresponding to maximum torque that were obtained in the trajectory optimization did

not exceed the assumed values in the configuration optimization. Once the numbers converged, the problem was considered to be optimal. This final result is what has been reported in the previous subsections and summarized in the table above.

However, it should be noted that in reality there are multiple boxes that should be placed on the pallet. Depending on the final location of the boxes the optimized path for sub-system-4 changes. Thus, the maximum acceleration and velocity values will change. This will effect the parameter value for subsystem-2 which is output of subsystem-4. For our problem we only consider the farthest box for optimization. A future study should be performed that optimizes this for all of the final box locations.

Since all the subsystems are interdependent, the results shown in the respective sub-systems are the final results, after following iterative process described above. So, we have used All-in-One approach from the beginning of the project.

Reference

1. http://www.met.reading.ac.uk/~vb904302/browne_phd.pdf - 30th January
2. Adept - http://www.adept.com/products/robots/scara/cobra-s600/general - 14th February
3. SolidWorks - http://www.solidworks.com/sw/docs/Optimization_2010_ENG_FINAL.pdf - 14th February.
4. SolidWorks - http://help.solidworks.com/2012/English/SolidWorks/sldworks/c_Design_Studies_in_SolidWork s.htm? id =3620793c93f64c52819ab07126f61048 - 14th February
5. http://glendale.directrouter.com/~modernte/components/com_wordpress/wp/wp-content/uploads/2013/05/Using-FEA-Optimization-to-Guide-Innovation.pdf - 14th February
6. Design Optimization of Robotic Arms by Prof. L. S Utpat, Prof. Chavan Dattatraya K,Yeolekar N., Sahasrabudhe A, Mandke S. - 4th April
7. Crack Location and Size Prediction in a Cantilever Beam Using Artificial Neural Network by Aniket Deo, Apratim Mishra - 10th April
8. Yoshikawa, T. (1990). *Foundations of robotics: analysis and control*. MIT Press.
9. http://www.optimaldesign.org/papers/1988/566-88-07.pdf
10. Foundation of Robotics-Analysis and control – Tsuneo Yokhikawa
11. Robot modeling and control first edition Mark W. Spong, Seth Hutchinson, and M. Vidyasagar
12. Robotics-modelling, planning and control by Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, Giuseppe Oriolo
13. A New Placement Heuristic for the Orthogonal Stock-Cutting Problem by E. K. Burke, G. Kendall, G. Whitwell.
14. A hybrid placement strategy for the three-dimensional strip packing problem by S.D. Allen, E.K. Burke, G. Kendall

Appendix

Trajectory Previous Attempt

Initially, when the entire problem was implemented as described in the report, it caused many errors in the solver. For this reason, a simpler objective function was determined in order to study the feasibility of this project. This objective function treats the arms of the manipulator as if they have a large amount of friction (or are the electrical equivalent of large resistors), in order to obtain an equation for energy. Thus the power required to operate the manipulator is related to velocity squared. An equation was obtained as follows:

$$minimize\ F_{a_1,a_2\ldots a_n,b_1,b_2\ldots b_n,t_f} = -\frac{P * t_{total}}{t_f * 2} + \frac{t_{total}}{t_f} p_w \int_0^{t_f} (R_1\omega_1^2 + R_2\omega_2^2)\ dt$$

where the values for R_1 and R_2 represent the "resistances of the arms". These values were set to 5 and 3 respectively.

Unfortunately, this change caused the solver to time out. The inequality constraint, g_3, limiting the end effector's maximum acceleration was then eliminated. Finally, a solution for a ninth order polynomial determined (using the MINOS solver for AMPL) as

a1	0.523599
a2	0
a3	2.331575
a4	1.175014
a5	0.490648
a6	0.046129
a7	-0.26431
a8	-0.49293
a9	-0.66813

b1	0
b2	0
b3	1.865261
b4	0.940011
b5	0.392517
b6	0.036903
b7	-0.21145
b8	-0.39434
b9	-0.53451

tf	5.26306

where the "Cost", or money made is $456.01.

These trajectories were then plotted in MATLAB and can be seen below.

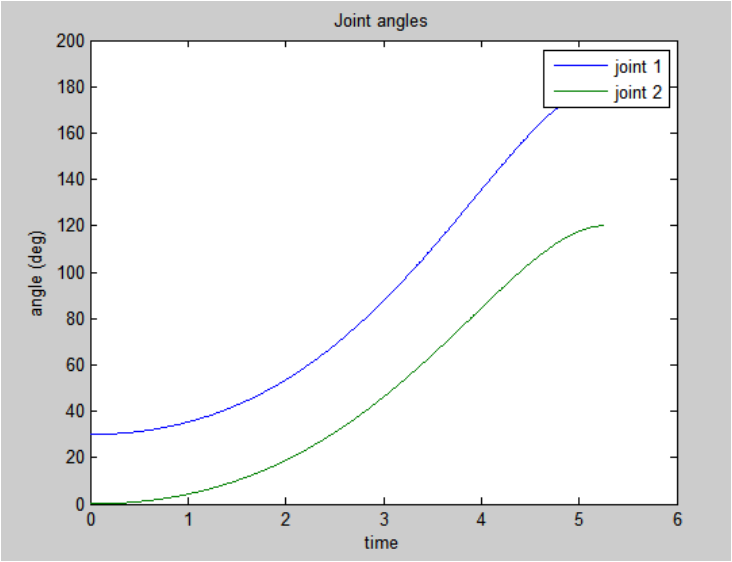

Figure: Joint angles versus time

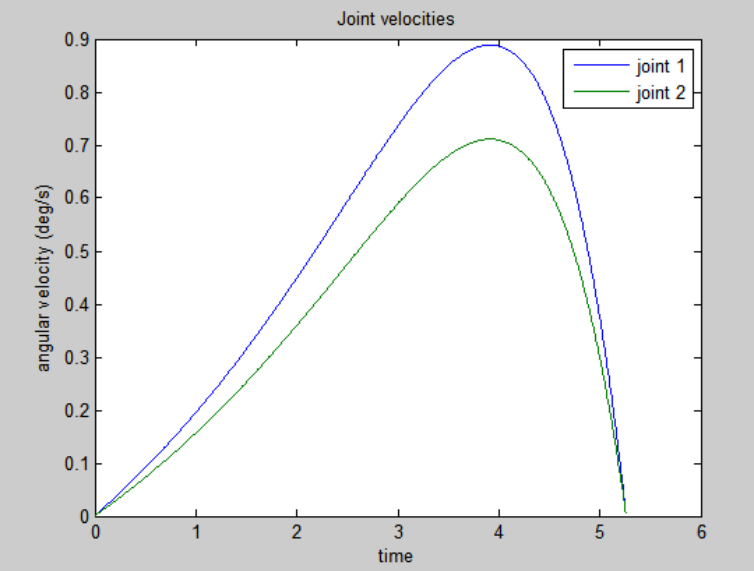

Figure: Joint angular velocities versus time

Note that these results use parameter values that are different than the ones in the report. This includes values for the link lengths, link masses, and even parameters such as the cost of electricity and the amount of money made per box stacked. However, that being said, the nature of the solution is still very different than the optimal solution found in the report. The figure above shows that both trajectories go directly to the desired final position. Adjusting the parameter values does not change the general shape of these trajectories.

Design optimization of link-1

The Design study was also undertaken for the link 1 but it did not matter much since the goal was to reduce the masses of the manipulators (Link 1 and Link 3).

Table 6.:- Design variables for link-1

Name	Type	Value		Units
Inner Diameter ID	**Range**	Min:1	Max:5	in
Outer Diameter OD	**Range**	Min:6	Max:9	in

Table 0:- Constraint for link-1

Sensor name		Bounds	Units	Study name
Stress 3	**is less than**	Max:1e+007	N/m^2	Static 3

Table 8:- Goal for link-2

Name	Goal	Properties	Weight	Study name
Mass 3	Minimize	Mass	10 lbs	-

Table 9:- Result for the link 1

Component name	Units	Initial	Optimal
id	in	4.99994	5
od	in	6.00005	6
Stress1	N/m^2	5.6983e+006	5.7386e+006
Mass1	lb	24.35375	24.35208

Code

Sub-system-1-code

```
clc
clear
close all
tic;
rng(0);
ws=40; % width of stock sheet
hs=60; % height of stock sheet
ws_a=zeros(1,ws);
hs_a=zeros(1,hs);
solution_final=zeros(1,4);
d=size(box);
a1=[];
b1=[];

for i=1:d(1)
    if box(i,1)<box(i,2)
    temp=box(i,1);
    temp1=box(i,2);
    box(i,1)=temp1;
    box(i,2)=temp;
    end
end

box=sortrows(box,[-1,-1]);

CA=0;
CB=0;
r=1;
P=isempty(box);

while (ws_a<=60 & P~=1)
   fprintf('\n finishes %d boxes',size(solution_final,1)-1);
   solution_check=solution_final;

   sc=size(solution_check);
   sf=size(solution_final);

   if r==1||g==1
        r=r+1;
   end

    [k1,k2]=min(ws_a);

    gap_location = min(k2);
    cc=size(find(ws_a==k1));
    gap_width=cc(2);
    [k3,k4]=min(hs_a);
    height_location = min(k4);
    cc1=size(find(ws_a==k3));
    height_width=cc1(2);

    CA=0;
```

```
    CB=0;



for j=1:size(box,1)

    diff=[(gap_width-box(:,1)),(gap_width-box(:,2))];
end

[min_diff1,Idiff1]=min(diff(:,1));
[min_diff2,Idiff2]=min(diff(:,2));

if min_diff1<0 || min_diff2<0
    a1 = find(~diff(:,1));
    b1 = find(~diff(:,2));
    b1=min(b1);
    a1=min(a1);
    tf=isempty(b1);
    tf1=isempty(a1);

if tf==1 && tf==1
   x1=diff(:,1);
   x2=diff(:,2);
   a1=find(x1>0);
   b1=find(x2>0);
   a1=min(a1);
   b1=min(b1);
   tx=isempty(a1);
   tx1=isempty(b1);

   if tx==1 && tx1==1
     wss=sort(ws_a);
     [z1,II]=min(ws_a);

     z=find(wss>z1);

     ws_a(II)=ws_a(z(1));
   else
       tf=isempty(b1);
    tf1=isempty(a1);
   end

end

if b1<a1
    solution_1= box(b1,:);
    solution_1 = rot90(solution_1,2);
    location=[gap_location-1,ws_a(gap_location)];
    solution=horzcat(solution_1,location);
    solution_final=vertcat(solution_final,solution);


rectangle('Position',[solution_final(r,3),solution_final(r,4),solution_final(
r,1),solution_final(r,2)]);
    axis([0 40 0 60]);
```

```
    hold on
    for j1=0:solution(1)-1
            ws_a(gap_location+j1)=ws_a(gap_location+j1)+solution_1(2);
    end
    box(b1,:)=[];
end

if (tf~=1 && tf1==1)
        solution_1 = box(b1,:);
        solution_1 = rot90(solution_1,2);
         location=[gap_location-1,ws_a(gap_location)];
    solution=horzcat(solution_1,location);
    solution_final=vertcat(solution_final,solution);


rectangle('Position',[solution_final(r,3),solution_final(r,4),solution_final(
r,1),solution_final(r,2)]);
    axis([0 40 0 60]);
    hold on
        for j1=0:solution(1)-1
                ws_a(gap_location+j1)=ws_a(gap_location+j1)+solution(2);
        end
    box(b1,:)=[];



    end

if a1<b1
    solution_1 = box(a1,:);
     location=[gap_location-1,ws_a(gap_location)];
    solution=horzcat(solution_1,location);
    solution_final=vertcat(solution_final,solution);


rectangle('Position',[solution_final(r,3),solution_final(r,4),solution_final(
r,1),solution_final(r,2)]);
    axis([0 40 0 60]);
    hold on

    for j5=0:solution(1)-1
            ws_a(gap_location+j5)=ws_a(gap_location+j5)+solution_1(2);
            hs_a(height_location+j5)=hs_a(height_location+j5)+solution_1(1);
    end
    box(a1,:)=[];
end

if a1==b1
    solution_1 = box(a1,:);
     location=[gap_location-1,ws_a(gap_location)];
    solution=horzcat(solution_1,location);
    solution_final=vertcat(solution_final,solution);


rectangle('Position',[solution_final(r,3),solution_final(r,4),solution_final(
r,1),solution_final(r,2)]);
    axis([0 40 0 60]);
```

```
    hold on



    for j5=0:solution(1)-1
            ws_a(gap_location+j5)=ws_a(gap_location+j5)+solution_1(2);
            hs_a(height_location+j5)=hs_a(height_location+j5)+solution_1(1);
    end
    box(a1,:)=[];

end

if (tf1~=1 && tf==1)
    solution_1 = box(a1,:);
    location=[gap_location-1,ws_a(gap_location)];
    solution=horzcat(solution_1,location);
    solution_final=vertcat(solution_final,solution);


rectangle('Position',[solution_final(r,3),solution_final(r,4),solution_final(
r,1),solution_final(r,2)]);
    axis([0 40 0 60]);
    hold on
    for j5=0:solution(1)-1
            ws_a(gap_location+j5)=ws_a(gap_location+j5)+solution(2);
    end
    box(a1,:)=[];
end
end
clear a1;
clear b1;

if (min_diff1<min_diff2 && min_diff1>0 &&
min_diff2>0)||min_diff1==0||min_diff1==min_diff2
    solution_1 = box(Idiff1,:);
    location=[gap_location-1,ws_a(gap_location)];
    solution=horzcat(solution_1,location);
    solution_final=vertcat(solution_final,solution);

rectangle('Position',[solution_final(r,3),solution_final(r,4),solution_final(
r,1),solution_final(r,2)]);
    axis([0 40 0 60]);
    hold on
    for j2=0:solution_1(1)-1
            ws_a(gap_location+j2)=ws_a(gap_location+j2)+solution_1(2);
    end
      box(Idiff1,:)=[];
end

if (min_diff1>min_diff2 && min_diff1>0 && min_diff2>0)||min_diff2==0
    solution_1 = box(Idiff2,:);
    solution_1 = rot90(solution_1,2);
    location=[gap_location,ws_a(gap_location)];
    solution=horzcat(solution_1,location);
    solution_final=vertcat(solution_final,solution);
```

```
rectangle('Position',[solution_final(r,3),solution_final(r,4),solution_final(
r,1),solution_final(r,2)]);
    axis([0 40 0 60]);
    hold on
    for j3=0:solution_1(1)-1
            ws_a(gap_location+j3)= ws_a(gap_location+j3)+solution_1(2);
    end
      box(Idiff2,:)=[];
end
solution_check1=solution_final;
sc=size(solution_check);
sf=size(solution_check1);

 if sc(1)~=sf(1)
       g=1;
   else
       g=0;
   end
   P=isempty(box);
end

xx=find(ws_a>60);
txx=isempty(xx);

if txx~=1
    pq=size(solution_final);
    solution_final(pq(1),:)=[];
    pq1=size(solution_final);
    figure(2);
    for q=1:pq1(1)

rectangle('Position',[solution_final(q,3),solution_final(q,4),solution_final(
q,1),solution_final(q,2)]);
    axis([0 40 0 60]);
    hold on
    end
end


toc;
```

Sub-system-2 code

```
% main %
close all
clc

% call optimization
A=[-1 0;0 -1]; % condition for link length to be greater than on%
B=[0;0];
Aeq=[];
Beq=[];
Lc = 0.94;
x0=[0.8;0.6]; %initial condition%
```

```
lb=[0,0];
ub=[10,10];
options = optimoptions(@fmincon,'Display','iter','Algorithm','sqp');%active-set,interior-point,sqp%
[xsol,FVAL,EXITFLAG,OUTPUT,LAMBDA,GRAD,HESSIAN] =
fmincon(@objproject,x0,A,B,Aeq,Beq,lb,ub,@(x)conproject(x,net_m2,net_m1_2),options)

len = xsol;
f_len = @(x)rosenbrock(x,len);
g_len = @(x)rosenbrockg(x,len);
H_len = @(x)rosenbrockH(x,len);
opt.alg = 'newton'; % gradient or newton
opt.linesearch = false;
opt.eps = 0.0001;
x0 = Lc;
if strcmp(opt.alg,'gradient')
   solution = gradient(f_len,g_len,H_len,len,x0,opt);
elseif strcmp(opt.alg,'newton')
   solution = newton(f_len,g_len,H_len,len,x0,opt);
end
n_iter = length(solution);
L_out = solution(n_iter);
b=0.3;            %m%
L=1.21;B_1=1.016;         %m%
Lp=0.2;
theeta2=(pi)+acos((xsol(1)^2+xsol(2)^2-(L_out+0.5*b)^2)/(2*xsol(1)*xsol(2)));
theeta1=asin(sqrt(((xsol(2)*sin(theeta2))^2)/((xsol(1)+(xsol(2)*cos(theeta2)))^2+(xsol(2)*sin(theeta2))^2)));
f_final=-(xsol(1)*xsol(2)*sin(abs(theeta2)));
output=[len(1);len(2);L_out;f_final;theeta1;theeta2];
disp(output)
%plotting inital assumption and final solution%
xc=0;
yc=0;
xi=[1.2;1.2];
theeta2=(-pi)+acos((xi(1)^2+xi(2)^2-(L_out+0.5*b)^2)/(2*xi(1)*xi(2)));
theeta1=asin(sqrt(((xi(2)*sin(theeta2))^2)/((xi(1)+(xi(2)*cos(theeta2)))^2+(xi(2)*sin(theeta2))^2)));
x1i=(xi(1)*cos(theeta1));
y1i=(xi(1)*sin(theeta1));
x2i=(xi(1)*cos(theeta1))+(xi(2)*cos(theeta1+theeta2));
y2i=(xi(1)*sin(theeta1))+(xi(2)*sin(theeta1+theeta2));
plot([xc x1i],[yc y1i],'r')
hold on
plot([x1i x2i],[y1i y2i],'b')
hold on
theeta2=(-pi)+acos((xsol(1)^2+xsol(2)^2-(L_out+0.5*b)^2)/(2*xsol(1)*xsol(2)));
theeta1=asin(sqrt(((xsol(2)*sin(theeta2))^2)/((xsol(1)+(xsol(2)*cos(theeta2)))^2+(xsol(2)*sin(theeta2))^2)));
x1=(xsol(1)*cos(theeta1));
y1=(xsol(1)*sin(theeta1));
x2=(xsol(1)*cos(theeta1))+(xsol(2)*cos(theeta1+theeta2));
y2=(xsol(1)*sin(theeta1))+(xsol(2)*sin(theeta1+theeta2));
hold on
plot ([xc x1],[yc y1],'g','lineWidth',3)
hold on
plot ([x1 x2],[y1 y2],'y','lineWidth',3)
rectangle('Position',[-(L/2),Lp,L,B_1])
rectangle('Position',[L_out,-(b/2),b,b])
function f = objproject(x)
```

```
Lc=0.94;
b=0.3;
theeta2=(pi)+acos((x(1)^2+x(2)^2-(Lc+0.5*b)^2)/(2*x(1)*x(2)));
f=-(x(1)*x(2)*sin(theeta2));
 end


function [c, ceq] = conproject(x,net_m2,net_m1_2)
Lc =0.94;
% Nonlinear inequality constraints
%constants%
mm2 =4.5;              %Kg%
mm3 = 3.5;             %Kg%
%parameter
T2max = 80;       %N.m%
T3max = 10;     %N.m%
M = 22;              %Kg%
%this is dependant on the pallet design%
b = 0.3;             %m%
L = 1.21;
B_1 = 1.016;         %m%
Lp = 0.2;
% This is dependant on path planning%
%angular velocities and acceleration%
v2dot = 1;
v2dotdot = 8;

v3dot = 1.5;
v3dotdot = 20;
%This is dependant on sturctural design%

ml3 = net_m2(x(2));
ml2 = net_m1_2([x(1);ml3]);

theeta2 = (pi)+acos((x(1)^2+x(2)^2-(Lc+0.5*b)^2)/(2*x(1)*x(2)));
%theeta1=acos((x(1)^2-x(2)^2+(Lc+0.5*b)^2)/(2*x(1)*(Lc+0.5*b)));

Lg2 = (ml2*x(1)/2+mm3*x(1))/(mm3+ml2);
Lg3 = (ml3*x(2)/2+(M)*x(2))/(ml3+M);

I2 = (x(1)-Lg2)^2*mm3+1/12*ml2*x(1)^2+ml2*(1/2*x(1)-Lg2)^2;
I3 = (x(2)-Lg3)^2*(M)+1/12*ml3*x(2)^2+ml3*(1/2*x(2)-Lg3)^2;

tau1 =
(ml2.*Lg2^2+I2+ml3.*(x(1)^2+Lg3^2+2.*x(1).*Lg3.*cos(theeta2))+I3).*(v2dotdot)+((ml3.*(Lg3^2+x(1).*Lg3.*c
os(theeta2))+I3).*v3dotdot)-(ml3.*x(1).*Lg3.*sin(theeta2).*(2.*v2dot.*v3dot+v2dot^2));
tau2
=((ml3.*(Lg3^2+x(1).*Lg3.*cos(theeta2))+I3).*v2dotdot)+((ml3.*Lg3^2+I3).*v3dotdot)+ml3.*x(1).*Lg3.*sin(thee
ta2).*v2dot^2;
c=[tau1-T2max;tau2-T3max;(sqrt((L+Lp)^2+(0.5*B_1)^2))-x(1)-x(2)]

ceq = [];
end
function solution = newton(f,g,H,len,x0,opt)
x = x0;
iter = 1;
```

```
solution(iter,1) =x;
gnorm = 0.1;
while gnorm>opt.eps
   iter = iter + 1;
   if opt.linesearch
      a = lineSearch(f,g,H,x,opt);
   else
      a = 0.01;
   end
   g1=rosenbrockg(x,len);
   H =rosenbrockH(x,len);
   gnorm = norm(g1);
   x = x-(a*inv(H)*g1);
   solution(iter,1) =x ;
end
end


function f = rosenbrock(x,len)
l1=len(1);
l2=len(2);
b=0.3;
theeta2=pi-acos((l1^2+l2^2-(x(1)+0.5*b)^2)/(2*l1*l2));
f=-(l1*l2*sin(theeta2));
end


function g = rosenbrockg(x,len)
l1=len(1);
l2=len(2);
b=0.3;
g=-((2*x(1) + b)*(l1^2 + l2^2 - (x(1) + b/2)^2))/(4*l1^2*(1 - (l1^2 + l2^2 - (x(1) + b/2)^2)^2/(4*l1^2*l2^2))^(1/2));
end


function H = rosenbrockH(x,len)
l1=len(1);
l2=len(2);
b=0.3;
H =(2*x(1) + b)^2/(4*l1^2*(1 - (l1^2 + l2^2 - (x(1) + b/2)^2)^2/(4*l1^2*l2^2))^(1/2)) - (l1^2 + l2^2 - (x(1) +
b/2)^2)/(2*l1^2*(1 - (l1^2 + l2^2 - (x(1) + b/2)^2)^2/(4*l1^2*l2^2))^(1/2)) + ((2*x(1) + b)^2*(l1^2 + l2^2 - (x(1) +
b/2)^2)^2)/(16*l1^4*l2^2*(1 - (l1^2 + l2^2 - (x(1) + b/2)^2)^2/(4*l1^2*l2^2))^(3/2));
End
```

Sub-system-4

Trajectory code

```
#MAIN CODE#
reset;
# Parameters #
# link lengths etc
param pi := 4*atan(1);
param L1 := .9227;
param L2 := .5760;
param m1 := 14.4142;
```

```
param m2 := 4.8869;
param m3 := 0;

param mb := 20;
param me := 2;

param mm1 := 4.5;
param mm2 := 3.5;

param Lg1 := (m1*L1/2+mm2*L1)/(mm2+m1);
param Lg2 := (m2*L2/2+(mb+me+m3)*L2)/(m2+mb+me);
param I1 := (L1-Lg1)^2*mm2+1/12*m1*L1^2+m1*(1/2*L1-Lg1)^2;
param I2 := (L2-Lg2)^2*(mb+me+m3)+1/12*m2*L2^2+m2*(1/2*L2-Lg2)^2;

param initAng1 := 31.966*pi/180;
param initAng2 := -90.03*pi/180;
param initVel1 := 0;
param initVel2 := 0;

param tau1max:= 84;
param tau2max := 14;

param finAng1 :=   130.7*pi/180;
param finAng2 :=   -69*pi/180;
param finVel1 := 0;
param finVel2 := 0;

param n := 15;
param T :=100; #number of time steps

param tf :=1;
param ttot := 3600; #1 hour
param P := .005; # $ made per package
param pw:= 3.33*10^-8; # $ per Joule #originally 3.33e-8

param amax:= 20; # max accel m/s^2

#variables
var a{1..n};
var b{1..n};

var tf2 >=0.01, <= 60;

var th1{t in 1..T} = sum{m in 1..n} a[m]*(t*tf/T)^(m-1);
var th1d{t in 1..T} = 1/tf2*sum{m in 2..n}  (m-1)*a[m]*(t*tf/T)^(m-2);
var th1dd{t in 1..T} = 1/(tf2^2)*sum{m in 3..n} (m-1)*(m-2)*a[m]*(t*tf/T)^(m-3);
```

```
var th2{t in 1..T} =  sum{m in 1..n} b[m]*(t*tf/T)^(m-1);
var th2d{t in 1..T} = 1/tf2*sum{m in 2..n}  (m-1)*b[m]*(t*tf/T)^(m-2);
var th2dd{t in 1..T} = 1/(tf2^2)*sum{m in 3..n} (m-1)*(m-2)*b[m]*(t*tf/T)^(m-3);

var tau1{t in 1..T} =
(m1*Lg1^2+I1+m2*(L1^2+Lg2^2+2*L1*Lg2*cos(th2[t]))+I2)*(th1dd[t])+(m2*(Lg2^2+L1*Lg
2*cos(th2[t]))+I2)*th2dd[t]-m2*L1*Lg2*sin(th2[t])*(2*th1d[t]*th2d[t]+th1d[t]^2);
var tau2{t in 1..T} =
(m2*(Lg2^2+L1*Lg2*cos(th2[t]))+I2)*th1dd[t]+(m2*Lg2^2+I2)*th2dd[t]+m2*L1*Lg2*sin(th2
[t])*th1d[t]^2;

## Objective
maximize cost:
(ttot*P)/(tf2*2)-pw*ttot/(tf2)*(sum{t in 1..T} sum{m in 1..n} tf/T*(max(tau1[t]*(((m-
1)*a[m]*(t/T*tf)^(m-2))/tf2),0)+max(tau2[t]*(((m-1)*b[m]*(t/T*tf)^(m-2))/tf2),0)));

subject to A1:
a[1] = initAng1; #initial position joint 1
subject to A2:
a[2] = initVel1; # initial velocity joint 1

subject to B1:
b[1] = initAng2; # initial position joint 2
subject to B2:
b[2] = initVel2; # initial velocity joint 2

subject to FA1:
sum{m in 1..n} a[m]*tf^(m-1)= finAng1; #final position joint 1

subject to FV1:
sum{m in 2..n} (m-1)*a[m]*tf^(m-2)= finVel1; #final ang vel joint 1

subject to FA2:
sum{m in 1..n} b[m]*tf^(m-1)= finAng2; #final position joint 2
subject to FV2:
sum{m in 2..n} (m-1)*b[m]*tf^(m-2)= finVel2; #final ang vel joint 2

subject to Consttau1 {t in 1..T}:
abs(tau1[t])<= tau1max;

subject to Consttau2 {t in 1..T}:
abs(tau2[t])<=tau2max;

subject to ae{t in 1..T}:
```

((th1dd[t]*(-L1*sin(th1[t])-L2*sin(th1[t]+th2[t]))-L2*sin(th1[t]+th2[t])*th2dd[t]-
(L1*cos(th1[t])+L2*cos(th1[t]+th2[t]))*th1d[t]^2-L2*cos(th1[t]+th2[t])*th2d[t]^2-
2*L2*cos(th1[t]+th2[t])*th1d[t]*th2d[t])^2+(L1*cos(th1[t])*th1dd[t]-
L1*sin(th1[t])*th1d[t]^2+L2*cos(th1[t]+th2[t])*th1dd[t]-L2*sin(th1[t]+th2[t])*th1d[t]*th2d[t]-
L2*sin(th1[t]+th2[t])*th1d[t]^2+L2*cos(th1[t]+th2[t])*th2dd[t]-
L2*sin(th1[t]+th2[t])*th1d[t]*th2d[t]-L2*sin(th1[t]+th2[t])*th2d[t]^2)^2) <= amax^2;

#Run File code

#option knitro_options "outlev=6";

solve;
print {m in 1..n} a[m];
print {m in 1..n} b[m];
display cost;
display tf2;

%MATLAB CODE TO CREATE TRAJECTORY GRAPHS

```
close all; clc; clear all;

L1 = .9227;
L2 = .5760;

 m1 = 14.4142;
 m2 = 4.889;
 m3 = 0;

 mb = 20;
 me = 2;

 mm1 = 4.5;
 mm2= 3.5;

 Lg1 = (m1*L1/2+mm2*L1)/(mm2+m1);
 Lg2 = (m2*L2/2+(mb+me+m3)*L2)/(m2+mb+me);
 I1 = (L1-Lg1)^2*mm2+1/12*m1*L1^2+m1*(1/2*L1-Lg1)^2;
 I2 = (L2-Lg2)^2*(mb+me+m3)+1/12*m2*L2^2+m2*(1/2*L2-Lg2)^2;

a = [0.5579119486925074 0 6.453465664103448 0.9548264147620277 -
2.1676278075134774 -13.229979666539501 8.199129453289426e-11
14.511416816246394 1.1211668490546573e-08 -0.25867432780176736 -
6.976959371790416 -7.89567333210103e-07 4.2542508214986585e-08
4.5479775418005435 -2.1112111438720036];
b = zeros(1,length(a));
b(1) = -1.571;
b(2) = (1.571-1.204)/1;
%a = zeros(1,length(a));
%a(1) = .5579;
%a(2) = (2.281-.5579)/1;
b = [-1.571319925570495 0 -9.549297710282321 19.558995007108475 -
0.7693630620929905 -14.884240574933148 -2.1243225729413836e-09
```

```
1.8995985625493614 7.43645327449504 -7.742106770525467e-10 -
0.0017870360609998147 -2.495143752373424 -2.095159528527718
1.0518714133907198e-05 1.2669770459965308];
tcorrect = 1.29283;
%%
 T =100; %number of time steps
 tf2 = tcorrect;

 tf=1;
 ttot = 3600; %1 hour
 P = .005; % $ made per package
 pw= 3.33*10^-8; % $ per Joule #originally 3.33e-8

tf = 1;
%tf = 1.48392;
[t,th1,th2,th1d,th2d,th1dd,th2dd]=genTraj5(a,b,tf);
cost = 0;

tpretty = t*tcorrect;

th1d = th1d/tcorrect;
th2d = th2d/tcorrect;

th1dd = th1dd/(tcorrect^2);
th2dd = th2dd/(tcorrect^2);

figure()
plot(tpretty,th1,tpretty,th2)
title('Joint angles');
legend('joint 2','joint 3');
xlabel('time'); ylabel('angle (rad)');

figure()
plot(tpretty,th1d,tpretty,th2d)
title('Joint velocities');
legend('joint 2','joint 3');
xlabel('time'); ylabel('angular velocity (rad/s)');

figure()
plot(tpretty,th1dd,tpretty,th2dd)
title('Joint accelerations');
legend('joint 2','joint 3');
xlabel('time'); ylabel('angular acceleration (rad/s^2)');

for i = 1:length(th1)
    ae(i) = (((th1dd(i).*(-L1.*sin(th1(i))-L2.*sin(th1(i)+th2(i)))-
L2.*sin(th1(i)+th2(i)).*th2dd(i)-
(L1.*cos(th1(i))+L2.*cos(th1(i)+th2(i))).*th1d(i)^2-
L2.*cos(th1(i)+th2(i)).*th2d(i)^2-
2.*L2.*cos(th1(i)+th2(i)).*th1d(i).*th2d(i))^2+(L1.*cos(th1(i)).*th1dd(i)-
L1.*sin(th1(i)).*th1d(i)^2+L2.*cos(th1(i)+th2(i)).*th1dd(i)-
L2.*sin(th1(i)+th2(i)).*th1d(i).*th2d(i)-
L2.*sin(th1(i)+th2(i)).*th1d(i)^2+L2.*cos(th1(i)+th2(i)).*th2dd(i)-
L2.*sin(th1(i)+th2(i)).*th1d(i).*th2d(i)-
L2.*sin(th1(i)+th2(i)).*th2d(i)^2)^2))^.5;
```

```
    tau1(i) =
(m1.*Lg1^2+I1+m2.*(L1^2+Lg2^2+2.*L1.*Lg2.*cos(th2(i)))+I2).*(th1dd(i))+(m2.*(
Lg2^2+L1.*Lg2.*cos(th2(i)))+I2).*th2dd(i)-
m2.*L1.*Lg2.*sin(th2(i)).*(2.*th1d(i).*th2d(i)+th1d(i)^2);

tau2(i)=(m2.*(Lg2^2+L1.*Lg2.*cos(th2(i)))+I2).*th1dd(i)+(m2.*Lg2^2+I2).*th2dd
(i)+m2.*L1.*Lg2.*sin(th2(i)).*th1d(i)^2;
    power1(i) = tau1(i)*th1d(i);
    power2(i) = tau2(i)*th2d(i);
end

for i = 1:length(t)
    cost(i) = -
pw*ttot/(tf2)*(tf/T*(max(tau1(i)*th1d(i),0)+max(tau2(i)*th2d(i),0)));
    costlink1(i) = -pw*ttot/(tf2)*(tf/T*(max(tau1(i)*th1d(i),0)));
    costlink2(i) = -pw*ttot/(tf2)*(tf/T*(max(tau2(i)*th2d(i),0)));
end
COST = sum(cost);
COSTlink1 = sum(costlink1)
COSTlink2 = sum(costlink2)

figure()
plot(tpretty,ae)
title('End Effector Acceleration');
xlabel('time'); ylabel('acceleration (m/s^2)');

figure()
plot(tpretty,tau1,tpretty,tau2)
title('Joint torques')
legend('Joint 2','Joint 3')
xlabel('time'); ylabel('torque (Nm)');

figure()
plot(tpretty,power1,tpretty,power2)
legend('joint 2','joint 3');
```