Optimizing motion paths with respect to the needs of the application

Jakob Pettersson Peeker

May 9th, 2014

1 Summary

A method for optimizing motion path definitions for servo unit motors of Tetra Pak's packaging machines using optimal control is chosen, described and tested.

Tetra Pak's packaging machines are characterized by the use of servo motors to control repetitive actions that are repeated in cycles. When developing new packaging machines the motion path of the servo motor is always optimized in some way, usually to complete a cycle of actions as fast as possible. While there is usually a need to maintain a specific position, velocity or acceleration at a few points along the cycle, there is often flexibility in where these points are located in time and in how the motion path is defined in-between these points. Finding an optimal path is time consuming with current methods employed by Tetra Pak and this thesis will focus on a way of using optimal control theory to find a motion path that is optimized according to the specific needs and limitations of the application. A common need is to perform a cycle as fast as possible, while adhering to a servo motor's limitations on torque.

The optimization method needs to optimize the path with respect to the needs of the application as well as the limitations of the mechanical system and the servo motor, and as such a significant part of the thesis concerns the problem definition and modelling of the mechanical system. As well as providing a method for optimizing motion paths, this thesis will in detail describe an example where a mechanical system will move a mass up and down as fast as possible, while adhering to a servo motor's limits on torque and speed. The optimization method is implemented for the specific example outlined in this thesis to show that this is a viable way for Tetra Pak to design motion paths.

The final result of this thesis is the definition of the optimization method used and the results of this method when optimizing the motion path for the described example. The goal of the method is to be able to be used in a wide variety of different scenarios within Tetra Pak's research and development, and as such it is constructed to be a generic method that can handle different mechanical systems as well as different aspects to be optimized. While the method is constructed to be generic, designing a workflow for using this method for other scenarios than the example outlined as well as integrating this method into Tetra Pak's current workflow is left for future research.

2 Introduction

In 1995 Björn Bachman and Joakim Lubeck did a master thesis on ”Curve Generation by variational optimization for controlling machine movement”, also led by my mentor Anders Sundberg at Tetra Pak and Gustaf Söderlind at LTH. They developed a method for defining motion paths using finite differences. The main difference between their their method and the one suggested in this thesis is that their method didn't optimize the time taken to reach a destination, but was rather used to find a smooth path given that we already know the time needed for the motion path to reach its final destination. It obtained a smooth path by minimizing the area that is defined by the integral over the speed, acceleration, and jerk(with a possibility to weigh them in accordance with their importance), while still maintaining the positions and speed and acceleration given for specific points in time. It was a very flexible method for doing this, as you could define the speed and acceleration for the path at specific points in time and you would usually get a very smooth curve with less user input need than Tetra Pak's currently used SCCA methods (see Section 4.4 for further details). This method has sadly not been developed further at Tetra Pak.

In comparison to both the current system of using SCCA methods and the method developed by Bachman and Lubeck, the method presented in this thesis takes into account the mechanics of the machine and the torque and speed limitations of the servo motor to obtain an optimized motion path.

3 Tetra Pak

Tetra Pak has a wide range of packaging machines that typically produce thousands of packages per hour. These machines shape a material into a container, fill the container and then seal it. This cycle is then repeated continuously. Improving existing packaging machines and developing new ones that produce packages with more functionality is an important part of Tetra Pak's research and there is always a need to produce packages as efficiently as possible, both in terms of production rate and package quality. A common goal is usually to produce as many packages an hour as possible, which means that the mechanical systems moving the packages around need to move as fast as possible.

One important aspect when developing a machine is defining the motion paths for the servo unit motors that are driving the machine. These paths will make sure that the different parts of the mechanical system are in the right place at the right time, and the definition of a motion path for a servo motor is crucial to the effectiveness of the machine. At Tetra Pak there is an opportunity to develop better ways to optimize this motion path based on the true needs of the application, as well as the limitations of the selected servo motor. Tetra Pak usually wants to minimize the time required to complete one cycle of an operation, the cycle time, while making sure that the machine operates within a servo motor's limitations on torque. Historically this has meant that the designers need to go through several iterations of a machine's design stages before they reach a satisfactory motion path. The goal of this thesis is to help develop a way to optimize these motion paths with respects to the needs and restrictions of the application. The needs will vary between different machines and different parts of each machine, but the most common optimization criteria is a need for as fast of a movement as possible.

The structure of the method to solve the optimization problem is defined as follows:

1. Defining the mechanical system
2. Defining the torque of the mechanical system
3. Defining the optimal control problem
4. Discretizing the optimal control problem
5. Solving the discretised problem

The definition of the mechanical system consists of defining how the servo motor can move and how it is connected to other moving parts of the system. Subsequently the next step is defining the torque at the servo motor, as a function of the current position, velocity and acceleration. The torque is calculated because one of the major limiting factors will be the servo motor's limitations on torque, and the optimization method needs to take this into account. The torque is dependant on the changes in kinetic and potential energy and as such it is written as a differential equation, which needs to be solved.

Steps 1 and 2 will be solved analytically for the specific example of a mechanical system that is described in this thesis. For other mechanical systems it might be more efficient to solve these steps numerically, but this should not impact the use of the optimization method.

After the torque is correctly defined this thesis goes on to define the optimal control problem. Here we define what it is that we want to optimize and what the constraints on the problem are. The optimal control problem is then discretised and transcribed into a Nonlinear Programming problem (NLP), and later as a Quadratic Programming problem. After this the problem can finally be solved in MATLAB using Sequential Quadratic Programming and the Quasi-Newton BFGS method.

4 Tetra Pak machines

Tetra Pak has had a lot of different packaging machines, but they all serve a similar purpose: to shape a material into a container, fill the container with a liquid and then seal the container. There is usually some additional treatment of the container as well, e.g. sterilization. The packaging machines are characterized by highly coordinated repetitive actions and typical cycle times are between 500 milliseconds and 1.5 seconds.

4.1 A6 machine

An example of a modern packaging system is the Tetra Pak machine (see Figure 1). The Tetra Pak A6 is their latest machine platform and it has approximately 50 servo axes, nearly all of them performing synchronized motion paths. The cycle time is 1.44 seconds and the production rate is 10000 aseptic packages per hour.

Figure 1: Tetra Pak A6 Machine

The A6 machine has been developed to be cost efficient and takes up 50% less space than its predecessor. It has a lower investment cost and uses half the electricity of other aseptic bottling lines. It uses injection moulding to fuse the top, carton sleeve and capped neck into a ready-to-fill package. Using ”gas phase technology”, it sterilizes the package in three steps: first the package is pre-heated, then it is treated with hydrogen gas, and finally it is ventilated with sterile air to eliminate all the gas in preparation for product filling. After it has been sterilized, the package is filled in four steps to minimize foaming.

As the description above hints, Tetra Pak packaging machines can be very complex and there is a need to optimize the functions involved in package production. It is not just the number of packages produced per hour that is important; Tetra Pak is also interested in lowering the costs to produce and run their machines. This is important to take into consideration when designing packaging machines and there is often a need for choosing more cost efficient drive solutions that can fulfil the specified needs.

4.2 Servo Motors

Cam wheel systems and indexing gears have previously been common solutions to generate repetitive motion patterns (see Figure 2).

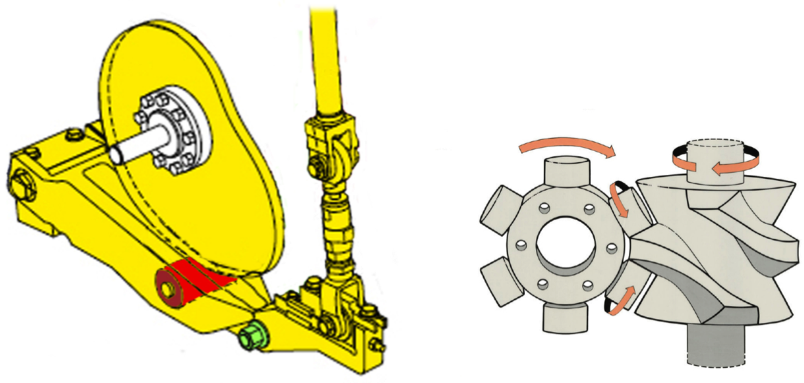

Figure 2: Cam wheel systems and indexing gears

The cam wheel systems and indexing gears were mechanically linked in order to achieve synchronized motion.

In modern machine designs they use electric servo motors instead of mechanical solutions for motion control. Each electric servo motor unit follows a motion path that has been defined during the development of the actual packaging machine. This motion path is then repeated every machine cycle. Each servo unit thus has a motion path defined that has been in some way optimized during development of the machine.

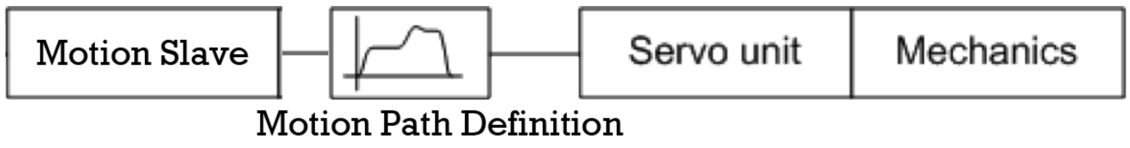

Figure 3: One servo axis gets its motion path from a Motion Slave

The motion paths for the individual servo axes are coordinated by a Motion Master, thus creating a synchronized system (see Figure 4).

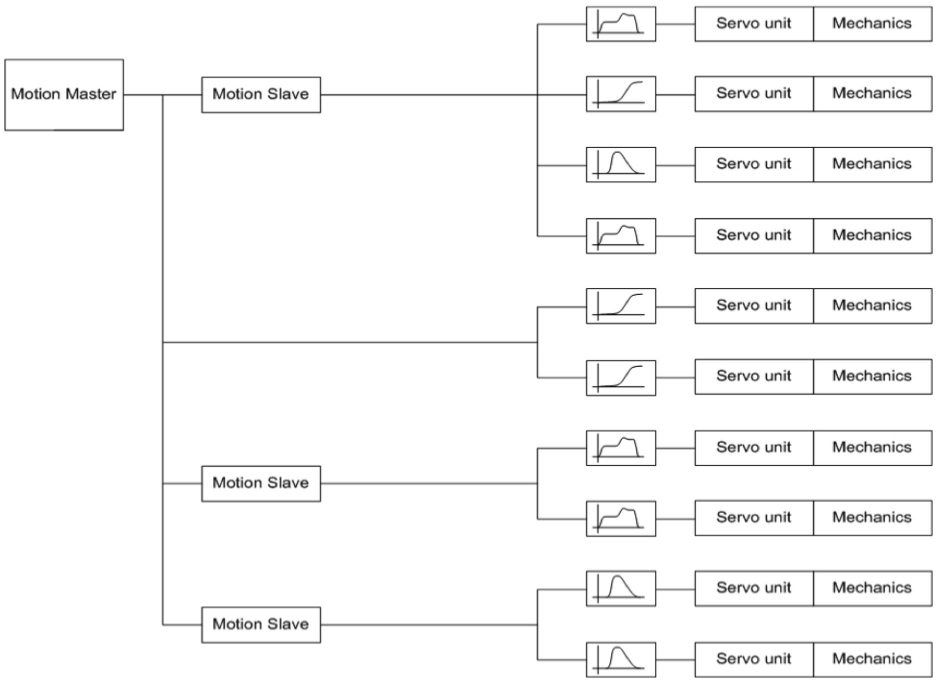

Figure 4: The individual Servo Axes are synchronized by the Motion Master

Choosing which servo motors to use and defining their motion paths is a complex process and there is a need for developing a method which further optimizes the motion paths of a specific servo motor while also being able to easily check how a different servo motor would affect the design. When finding the fastest motion path the torque-speed envelope of the servo motor is usually the main limiting factor. This envelope defines the speed and torque with which a servo motor may operate without degradation of performance.

A typical envelope limit as well as the speed and torque of an actually used motion path can look like Figure 5.

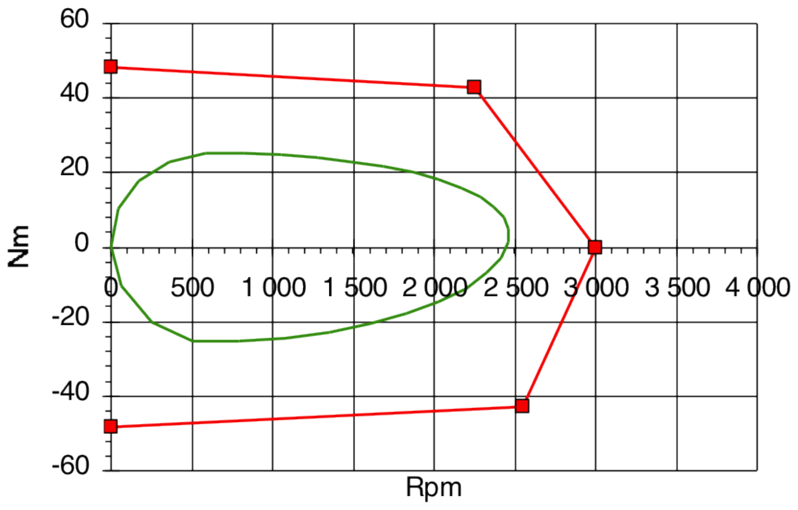

Figure 5: Red curve = Envelope limit, Green curve = Actual utilization

The shown motion path is not utilizing the servo motor to full capacity and could most likely be better defined to make a faster motion; however, designing an optimal motion "by hand" quickly gets very complex and thus better optimization tools are needed.

In many cases it is not an option to select a larger servo motor since this will introduce a larger moving mass into the system. This increase of moving mass often consumes the larger torque capacity of the motor.

4.3 Motion Path Background

While the needs and restrictions that constrain the optimization of a motion path are unique to each application, there is almost always the need for the fastest possible motion in order to increase machine productivity. The task of optimizing a servo motor application for short cycle times involves many interacting variables and component choices, where some of them like the motion path are by themselves complex.

In many cases it is only necessary from a process point of view to maintain a specific position, velocity or acceleration at a few points along a motion profile. The definition of the motion profile is left open and it is this part that can be subjected to optimization by the method described in this thesis. This optimization is very difficult to manage "by hand" and a tool that encourages iterations and presents results in a clear and relevant format is essential.

4.4 Current Motion Path development

To improve the design and optimization of Tetra Pak's machines it is necessary to understand how the procedure for defining their motion paths currently works. Basically, the current design procedure looks like this:

1. Definition of coordinated multi axis motion at load side (the final motion of, for example, a package being moved)

2. Definition of mechanical system
3. Mechanical load calculation for each motor separately
4. Motion path definition for each servo unit motor
5. Load and control simulation of single axis servo unit with mechanical system
6. Simulation of multi axis motion with mechanical system
7. Physical test and verification of actual performance

However, the process is not as structured as this list suggests. Steps $4 - 5$ are repeated until a satisfactory motion path definition is obtained, which can take many iterations. Many applications have high demands which means these calculations quickly become very complex.

The main improvement on the design process that this project will focus on is having the motion path definition take into account the simulation of the whole mechanical system, and optimize the motion path based on the needs of the application and the restriction of the motor. This will make item 4 on the list, Motion path definition for each servo unit motor, take the simulations of item 5 in account as well.

Currently Tetra Pak designs their motion paths with custom-built programs using SCCA functions. SCCA stands for Sine-Constant-Cosine-Acceleration. The motion paths are usually first defined using a combination of these SCCA functions; this requires a large amount of manual input and "trial and error" before a satisfactory motion path is obtained. Usually Tetra Pak knows the start and end conditions of a motion, and maybe the acceleration and speed at some points along the way, and then try and piece together a combination of SCCA segments that fulfil these conditions. When a motion path is obtained it is converted into polynomials before it can be used in the actual applications. The main issue with this way of working is that to optimize the motion path you need to go through many different iterations of this process, and it is very hard to evaluate how much you can improve your motion path.

4.5 Improving Development

The previous sections have outlined the basic background information and process needed for the development of motion paths for Tetra Pak's packaging machines. There are many aspects that need to be taken into consideration when designing a motion path and Tetra Pak wants to develop a new way of optimizing motion paths.

The main goal of this thesis is to define one method to simulate and optimize the motion paths that are typically applied for functions in Tetra Pak machines for packaging. There is a need for a method that defines such motion paths based on the true needs and limitations of the application. Typically the goal is to maximise production rate while still adhering to the limits on torque and speed imposed by the motor used to perform the motion. The method needs to be flexible enough to deal with many different configurations and requirements, to be as automated as possible to keep the manual testing and design iterations to a minimum and to consistently give good, optimal results based on the requirements and needs of the specific application. This will be done by simulating the torque as a function of the motion path and then using optimal control theory to optimize the time taken for the motion, with respect to the limitations. The input that should be needed for this method should ideally just be for the user to define the mechanical system that is being controlled and then define the different start and end conditions.

In this thesis I present a suggestion for an optimization method that could help Tetra Pak fulfil these requirements. The method will be designed around a specific example model of one Tetra Pak's actual machine parts, but it will be flexible enough that it could work with any machine configuration as long as it is accurately described.

5 Model

5.1 Definition

The model we study is based on one of Tetra Pak's current machine parts and consists of a mechanical system that will move a mass up and down, typically to perform a function when the mass is at the highest position. The execution time for moving the mass up and down shall be minimized since in many cases the packaging function can not start before the motion path has reached the end; this thesis is about finding a method for defining a motion path that does this. A schematic of model is depicted below in Figure 6, as well as my simplification of the model to the right of the schematic.

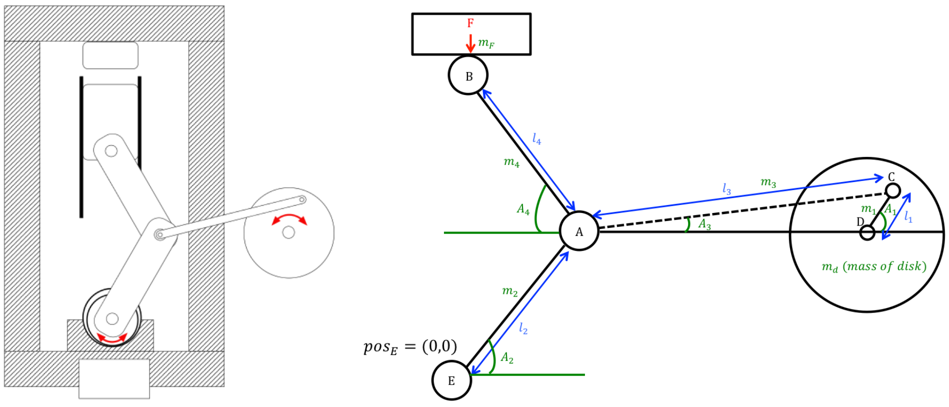

Figure 6: Schematic of the mechanical system to the left, simplification of the model to the right

A servo motor at point D will apply a torque to point D, this will affect the system in a way that moves the mass attached to point B in a vertical direction. Point B is constrained to vertical motion and point E is fixed to the machine frame.

The points A, B, C, D, are interconnected by mechanical links. Point C is rotated around D by the motor.

5.2 Problem Definition

This theses will attempt to find a method to optimize the motion path for the servo motor at point D in Figure 4 so that point C rotates 180 degrees around point D as fast as possible, given the limitations of the servo motor at point D. The method should be flexible enough that it could theoretically be applied to mechanical systems with different configurations as well.

5.3 Calculations of Torque

The first step to solving this problem is to calculate the torque at point D. Point D is situated at the known points x_0 and y_0 on an horizontal and vertical axis respectively, with Point E having the position $(0, 0)$. We can find the exact angles A_2, A_3 and A_4 using geometry (the x- and y-positions of a specific point are defined with the indexes x and y in this section):

$$\begin{aligned}
&Point\ B: \\
&\qquad B_x = 0 \\
&\qquad B_y = 2 \cdot \mathrm{l}_2 \cdot \sin(\mathrm{A}_2) = 2 \cdot y_2
\end{aligned}$$

$$\begin{aligned}
&Point\ C: && Point\mathrm{A}: \\
&\qquad C_x = \mathrm{l}_1 \cdot \cos(\mathrm{A}_1) + x_0 && \qquad \mathrm{A}_x = \mathrm{l}_2 \cdot \cos(\mathrm{A}_2) \\
&\qquad C_y = \mathrm{l}_1 \cdot \sin(\mathrm{A}_1) + y_0 && \qquad \mathrm{A}_y = \mathrm{l}_2 \cdot \sin(\mathrm{A}_2)
\end{aligned}$$

$$\begin{aligned}
&Point\ C: \\
&\qquad C_x = \mathrm{l}_2 \cdot \cos(\mathrm{A}_2) + \mathrm{l}_3 \cdot \cos(\mathrm{A}_3) \\
&\qquad C_x = \mathrm{l}_2 \cdot \sin(\mathrm{A}_2) + \mathrm{l}_2 \cdot \sin(\mathrm{A}_3)
\end{aligned}$$

These conditions lead to derivations of the angles A_3 and A_2:

$$\begin{aligned}&Angles:\\&\mathrm{l}_2\cdot\cos(A_2)+\mathrm{l}_3\cdot\cos(A3)=x_0+\mathrm{l}_1\cdot\cos(A_1)\\&\mathrm{l}_2\cdot\sin(A_2)+\mathrm{l}_3\cdot\sin(A3)=y_0+\mathrm{l}_1\cdot\sin(A_1)\end{aligned}$$

$$\Rightarrow$$

$$A_2 = tan^{-1}(\frac{y_0+\mathrm{l}_1\cdot\sin(A_1)}{x_0+\mathrm{l}_1\cdot\cos(A_1)})+$$
$$cos^{-1}(\frac{\mathrm{l}_2^2+(x_0+l_1\cdot\cos(A_1))^2+(y_0+l_1\cdot\sin(A_1))^2-l_3^2}{(2\cdot\mathrm{l}_2\cdot\sqrt{(x_0+l_1\cdot\cos(A1))^2+(y_0+l_1\cdot sin(A1))^2}}$$

$$A_3 = tan^{-1}(\frac{y_0+\mathrm{l}_1\cdot sin(A_1)-l_2\cdot\sin(A_2)}{x_0+\mathrm{l}_1\cdot\cos(A_1)-l_2\cdot\cos(A_2)})$$

The derivatives of these expressions quickly become cumbersome, which is why instead we're going to focus on the relationships between the derivatives of A_2 and A_3 and the derivatives of A_1:

$$\begin{aligned}&\mathrm{l}_2\cdot\sin(A_2)\cdot\dot{A}_2+\mathrm{l}_3\cdot\sin(A3)\cdot\dot{A}_3=\mathrm{l}_1\cdot\sin(A_1)\cdot\dot{A}_1\\&\mathrm{l}_2\cdot\cos(A_2)\cdot\dot{A}_2+\mathrm{l}_3\cdot\cos(A3)\cdot\dot{A}_3=\mathrm{l}_1\cdot\cos(A_1)\cdot\dot{A}_1\end{aligned}$$

$$\Rightarrow \quad \begin{pmatrix}\mathrm{l}_2\cdot\sin(A_2) & \mathrm{l}_3\cdot\sin(A_3)\\ \mathrm{l}_3\cdot\cos(A_2) & \mathrm{l}_3\cdot\cos(A_3)\end{pmatrix}\cdot\begin{pmatrix}\dot{A}_2\\ \dot{A}_3\end{pmatrix}=\begin{pmatrix}\mathrm{l}_1\cdot\sin(A_1)\\ \mathrm{l}_1\cdot\cos(A_1)\end{pmatrix}\cdot\dot{A}_1$$

$$\begin{pmatrix}\dot{A}_2\\ \dot{A}_3\end{pmatrix}=\begin{pmatrix}\frac{\mathrm{l}_1\sin(A_1-A_3)}{\mathrm{l}_2\sin(A_2-A_3)}\\ \frac{\mathrm{l}_1\sin(A_1-A_2)}{\mathrm{l}_3\sin(A_2-A_3)}\end{pmatrix}\cdot\dot{A}_1=\begin{pmatrix}S_1\\ S_2\end{pmatrix}\cdot\dot{A}_1$$

These expressions for the derivatives will be used when calculating the actual torque.

The total kinetic energy is defined

$$T=\sum_{j=1}^{n} m_j\frac{\dot{c}_{x,j}^2+\dot{c}y,j^2}{2}+\sum_{j=1}^{n}\mathrm{I}_j\frac{\dot{A}_j^2}{2}, \tag{1}$$

where $c_{x,j}$ and $c_{x,j}$ are the centre of masses for a specific bar l_j in the x- and y-axis respectively.

In this example we are most interested in the torque at point D, which needs to be within the servo motor's limitations. This torque depends on the change rate of the kinetic energy of the different parts of the system. The total kinetic energy is composed of the sum of the rotational kinetic energy and the translational kinetic energy. To get the translational kinetic energy we need the moment of inertia of rotation around the center of the bars for the different parts of the system:

$$\mathrm{I} = \frac{m \cdot l^2}{12} \tag{2}$$
$$\mathrm{I}_2 = \frac{m_2 \cdot \mathrm{l}_2^2}{12} \tag{3}$$
$$\mathrm{I}_3 = \frac{m_3 \cdot \mathrm{l}_3^2}{12} \tag{4}$$
$$\mathrm{I}_4 = \frac{m_4 \cdot \mathrm{l}_4^2}{12} \tag{5}$$

With these we can calculate the rotational kinetic energies:

$$T_{kr_1} = \frac{1}{2} \cdot \mathrm{I}_1 \cdot \dot{\acute{A}}_1^2 \tag{6}$$
$$T_{kr_2} = \frac{1}{2} \cdot \mathrm{I}_2 \cdot \dot{\acute{A}}_2^2 \tag{7}$$
$$T_{kr_3} = \frac{1}{2} \cdot \mathrm{I}_3 \cdot \dot{\acute{A}}_3^2 \tag{8}$$
$$T_{kr_4} = \frac{1}{2} \cdot \mathrm{I}_4 \cdot \dot{\acute{A}}_4^2 \tag{9}$$
$$\tag{10}$$

Bar 1 and bar 2 are fixed and don't have any translational kinetic energies. However, bar 3 and 4 are moving and we need to calculate their translational kinetic energies as well:

$$T_{kv_3} = \frac{1}{2} \cdot m_3 \cdot v_3^2 \tag{11}$$
$$T_{kv_4} = \frac{1}{2} \cdot m_4 \cdot v_4^2 \tag{12}$$
$$\tag{13}$$

When we have the kinetic energies we need to calculate the torque around our point D. To do this we need to write the kinetic energy T as a function of the angular rotation and angular acceleration around D. Then we can use the Lagrange equations to calculate the kinetic energy T:

$$T = \sum_{j=1}^{n} m_j \frac{\dot{x}_j^2 + \dot{y}_j^2}{2} + \sum_{j=1}^{n} I_j \frac{\dot{\acute{A}}_j^2}{2} \tag{14}$$

And the potential energy U:

$$U = \sum_{j=1}^{4} a \cdot \mathbf{y}_{j,m} \tag{15}$$

Where $\mathbf{y}_{j,m}$ is the vertical position of the centre of mass of bar j and a is the gravity of earth. Now we define the Lagrangian

$$L = T - U \tag{16}$$

$$\tag{17}$$

Finally we can get the torque from

$$\mathrm{M}_{\theta} = \frac{d}{dt}(\frac{dL}{d\dot{A}_1}) - \frac{dL}{dA_1} \tag{18}$$

$$\tag{19}$$

See the last Section of this paper, "Notes on calculations", for more specific calculations.

6 Optimization

6.1 Optimization criteria

The optimization criteria for our specific example are as follows:

1. Making sure the torque and speed of the motor are in accordance with the motor's characteristics
2. Moving as fast as possible from our desired start point to our desired end point

7 Optimal Control Problem

The optimal control problem is a free final time state constrained problem. We first define the state variables of the optimal control problem as $\mathbf{y}$. We then want to find the control functions $u(t)$ and the final time $\mathrm{t_F}$ to minimize

$$\phi[\mathbf{y}(\mathrm{t_F}), \mathrm{t_F}] \tag{20}$$

subject to the state equations:

$$\dot{\mathbf{y}}(\mathrm{t}) = \mathrm{f}[\mathbf{y}(\mathrm{t}), \mathrm{u}(\mathrm{t})] \tag{21}$$

the boundary conditions:

$$\mathrm{g}_i[\mathbf{y}(\mathrm{t_F}), \mathrm{u}(\mathrm{t_F}), \mathrm{t_F}] = 0, i = 1, ..., m_e \tag{22}$$

and the inequality constraints

$$\mathrm{g}_i[\mathbf{y}(\mathrm{t}), \mathrm{u}(\mathrm{t}), \mathrm{t}] \leq 0, i = m_e + 1, ..., m \tag{23}$$

where the initial conditions $\mathbf{y}(t_I) = \mathbf{y}_I$ are given at the fixed initial time t_I and the final time t_F is free. We will also call the inequality constraints $\mathrm{g}_{\mathrm{ineq}}$.

With our problem we define $\mathbf{y} = (\mathrm{x}_1, \mathrm{x}_2)$, where x_1 is the angle A_1 and x_2 is the angular velocity of A_1. The control variable u is defined as the angular acceleration of A_1. We define $n = 2$ as the number of state variables. The state equations define the derivatives of state variables as a function of the state variables and the control variable; in our case the derivative of x_1 is the second state variable x_2, while the derivative of the second state variable x_2 is just our control variable u. The state and control variables are set up this way because we are currently not interested in the the jerk of A_1, which we would need to define in a state equation if we set up the acceleration of A_1 as a state variable. Our optimal control problem is now defined as:

Initial definitions:

$$\phi(\mathrm{t_F}) = \mathrm{t_F}$$
$$\mathbf{y} = (x_1, x_2)$$
$$\mathrm{x}_1(\mathrm{t}) = \mathrm{A}_1(\mathrm{t})$$
$$\mathrm{x}_2(\mathrm{t}) = \dot{\mathrm{A}}_1(\mathrm{t})$$
$$\mathrm{u}(\mathrm{t}) = \ddot{\mathrm{A}}_1(\mathrm{t})$$

x_1 definitions:

$$\dot{\mathrm{x}}_1(t) = \mathrm{f}_1(\mathrm{t}, \mathrm{x}(t), \mathrm{u}(\mathrm{t})) = \mathrm{x}_2(\mathrm{t})$$
$$\mathrm{x}_1(0) = 0$$
$$\mathrm{x}_1(\mathrm{t_F}) = \mathrm{x}_F$$

x_2 definitions:

$$\dot{\mathrm{x}}_2(t) = \mathrm{f}_2(\mathrm{t}, \mathrm{x}(t), \mathrm{u}(\mathrm{t})) = \mathrm{u}(\mathrm{t})$$
$$\mathrm{x}_2(0) = 0$$
$$\mathrm{x}_2(\mathrm{t_F}) = 0$$

Control variable constraints:

$$\mathrm{u}_L \leq \mathrm{u}(\mathrm{t}) \leq \mathrm{u}_U$$

Inequality constraints:

$$\mathrm{torque}(\mathrm{t}, \mathrm{x}_1, \mathrm{x}_2, \mathrm{u}) - \mathrm{q}(\mathrm{t}, \mathrm{x}_1, \mathrm{x}_2, \mathrm{u}) \leq 0$$

This optimal control problem will later be transcribed as a Non-linear Programming (NLP) problem in section 8.2 when we try to find a solution to it, but it is important to first define the optimal control problem.

In our case, the inequality constraints $\mathrm{g}_{\mathrm{ineq}}$ (see equation (23)) becomes

$$\mathrm{torque}(t, x_1, x_2, u) \text{ - } q(t, x_1, x_2, u) \leq 0$$

where $q(x_2)$ is the limit of the torque as a function mainly of the angular velocity, but also of the sign of the torque (which is why it is a function of $\mathrm{t}, \mathrm{x}_1, \mathrm{x}_2, \mathrm{u}$ as well). An example of this limit is illustrated in Figure 7 below.

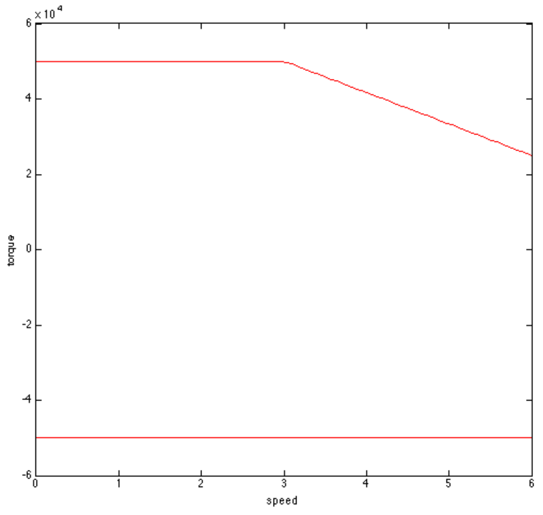

Figure 7: Speed / Torque envelope limit of servo motor

8 Optimization Method

8.1 Introduction

The general procedure in constrained optimization is to transform the problem into easier sub problems that can then be solved in an iterative process. One method for doing this is to replace the constraints with penalty functions and then solving this unconstrained problem. The initial constrained problem is then solved using multiple unconstrained optimizations that are solved in an iterative sequence, which then converge to the solution of the constrained problem. For larger-scale problem these methods can be inefficient and one alternative is to look at methods that solve the Karush-Kuhn-Tucker (KKT) equations, which are necessary for optimality for a constrained optimization problem. This is the method we choose to use and to do this we will first transcribe our optimal control problem into a nonlinear programming problem.

8.2 Transcription to Non-linear Programming problem

We want to transcribe the Optimal Control Problem (OCP) to a Non-linear Programming Problem (NLP). We start by discretizing the time according to

$$\mathrm{t}_\mathrm{I} = \mathrm{t}_1 < \mathrm{t}_2 < ... < \mathrm{t}_i < ... < \mathrm{t}_N = \mathrm{t}_\mathrm{F}$$

Where t_I is the initial time and t_F is the final time. The individual time points are called grid points, the state value at a grid point t_k is $\mathrm{x}_{1,k} = \mathrm{x}_1(\mathrm{t}_k)$

and $\mathrm{x}_{2,k} = \mathrm{x}_2(\mathrm{t}_k)$ and the control vector at t_k is $\mathrm{u}_k = \mathrm{u}(\mathrm{t}_k)$.

We treat the state values and the controls at the nodes, as well as the final time $\mathrm{t_F}$, as a set of NLP variables. We denote these as the parameters $\mathbf{Y}$:

$$\mathbf{Y} = (\mathrm{u}(\mathrm{t}_1), ..., \mathrm{u}(\mathrm{t}_N), \mathrm{x}_1(\mathrm{t}_1), ..., \mathrm{x}_1(\mathrm{t}_N), \mathrm{x}_2(\mathrm{t}_1), ..., \mathrm{x}_2(\mathrm{t}_N), \mathrm{t}_F) \in R^{N(3)+1}$$

We use an equidistant grid and can get t_n using

$$\mathrm{h}(\mathrm{t_F}) = \frac{\mathrm{t_F} - \mathrm{t_I}}{\mathrm{N}}$$
$$\mathrm{t}_n = \mathrm{t_I} + \mathrm{h} \cdot (n-1)$$

Where N is the number of grid points. This number is supplied by the user and will have a large impact on the time needed for the calculations.

Between two grid points we approximate the controls as piecewise linear interpolating functions between $\mathrm{u}(\mathrm{t}_j)$ and $\mathrm{u}(\mathrm{t}_{j+1})$ for $\mathrm{t}_j \leq \mathrm{t} < \mathrm{t}_{j+1}$:

$$\mathrm{u_{app}}(\mathrm{t}) = \mathrm{u}(\mathrm{t}_j) + \frac{\mathrm{t} - \mathrm{t}_j}{\mathrm{t}_{j+1} - \mathrm{t}_j}(\mathrm{u}(\mathrm{t}_{j+1} - \mathrm{u}(\mathrm{t}_j)) \text{ for } \mathrm{t}_j \leq \mathrm{t} < \mathrm{t}_{j+1}$$

The state values are chosen as continuously differentiable functions. They are piecewise defined as cubic polynomials between $x(t_j)$ and $x(t_j+1)$, with $\dot{x}_{app}(s) := f(x(s), u(s), s)$ at $s = t_j, t_{j+1}$ according to the following formula:

$$x_{app}(t) = \sum_{k=0}^{3} c_k^j (\frac{t - t_j}{h_j})^k \text{ for } t_j \leq t < t_{j+1},\ j = 1, ..., N-1 \tag{24}$$

$$c_0^j = x(t_j), \tag{25}$$
$$c_1^j = h_j f_j, \tag{26}$$
$$c_2^j = -3x(t_j) - 2h_j f_j + 3x(t_{j+1}) - h_j f_{j+1} \tag{27}$$
$$c_3^j = 2x(t_j) + h_j f_j - 2x(t_{j+1}) + h_j f_{j+1} \tag{28}$$

where $f_j := f(x(t_j), u(t_j), t_j),\ hj := t_{j+1} - t_j$

The approximating functions of the states have to satisfy the differential equations (21) at the grid points t_j, $j = 1, ..., N$, as well as at the centres $t_{c,j} := t_{j+1/2} := (t_j + t_{j+1})/2$, $j = 1, ..., N$ of the discretization intervals. This method is known as a cubic collocation at Lobatto points. Since the chosen approximation (24) already fulfils these constraints at the grid points t_j, we end up with the following remaining constraints (the index "app" stands for our chosen approximation):

- the collocation constraints at $t_{c,j}$

$$f(x_{app}(t_{c,j}), x_{app}(t_{c,j}), t_{c,j}) - \dot{x}_{app}(t_{c,j}) = 0,\ j = 1, ..., N-1 \tag{29}$$

- the inequality constraints at the grid points t_j

$$g_{ineq}(x_{app}(t_j), u_{app}(t_j), t_j) \geq 0,\ j = 1, ..., N$$

- and we combine the end point constraints with with the initial conditions $\mathbf{y}(t_I) = \mathbf{y}_I$ to form

$$\mathrm{r}_i(x_{app}(t_1), x_{app}(t_N), t_N) = 0,\ i = 1, ..., 2 \cdot n$$

The objective function we want to minimize is the same as in the optimal control definition

$$\phi(\mathrm{t_F}) = \mathrm{t_F} \tag{30}$$

8.3 Necessary optimality conditions

To form the necessary conditions for the discretised NLP problem we first need to state the Lagrangian, our cost function:

$$L(\mathbf{Y}, \lambda) = \Phi(x_N, t_n) + \sum_{j=1}^{m} \lambda_j g(x_j, u_j, t_j)$$

The necessary conditions for optimality are then

$$\nabla\Phi(x_N, t_n) + \sum_{j=1}^{m} \lambda_j \nabla g(x_j, u_j, t_j) = 0$$
$$\lambda_i g_i(x^*) = 0,\ \mathrm{i} = 1, ..., \mathrm{m_e}$$
$$\lambda_i \geq 0,\ \mathrm{i} = \mathrm{m_e} + 1, ..., \mathrm{m}$$

These are called the Karush–Kuhn–Tucker (KKT) conditions. λ_i are Lagrange multipliers and are needed to balance the difference in magnitude between the gradients of the objective function and the constraint gradients. One result of the KKT equations is that the constraints that are not active have their Lagrange multipliers set to zero. To solve our optimization problem we will solve these KKT equations using Sequential Quadratic Programming (SQP).

8.4 Quadratic Programming problem

To solve the NLP problem we will be setting up a Quadratic Programming (QP) problem which will be solved using an active set method. With the Lagrangian defined we obtain the Quadratic Programming (QP) sub problem by linearising the nonlinear constraints to get

$$\min_{d \in R^n} \frac{1}{2} d^T H_k d + \nabla\Phi(\mathbf{Y}_k) d$$
$$\nabla \mathrm{g}_i(\mathbf{Y}_k)^T d + \mathrm{g}_i(\mathbf{Y}_k) = 0, \ \mathrm{i} = 1, ..., \mathrm{m_e}$$
$$\nabla \mathrm{g}_i(\mathbf{Y}_k)^T d + \mathrm{g}_i(\mathbf{Y}_k) = 0, \ \mathrm{i} = \mathrm{m_e} + 1, ..., \mathrm{m}$$

where $H = \nabla^2 \mathrm{L}$

The solution to this subproblem is then used to get a new iteration

$$\mathbf{Y}_{k+1} = \mathbf{Y}_k + a_k d_k$$

This sequence is repeated until we find a solution which is deemed satisfactory, which in our case means that the the constraints are fulfilled and objective function has changed less than our function tolerance. The matrix H is a positive definite approximation of the Hessian matrix of the Lagrangian function; this approximation will be updated using the quasi-Newton BFGS method after every iteration, according to the formula

$$\mathrm{H}_{k+1} = \mathrm{H}_k + \frac{\mathrm{q}_k \mathrm{q}_k^T}{\mathrm{q}_k^T \mathrm{s}_k} - \frac{\mathrm{H}_k \mathrm{s}_k \mathrm{s}_k^T \mathrm{H}_k^T}{\mathrm{s}_k^T \mathrm{H}_k \mathrm{s}_k}$$

where

$$\mathrm{s}_k = \mathrm{x}_k,$$
$$\mathrm{q}_k = \nabla\phi(\mathrm{x}_{k+1})$$

This method needs a starting guess which the user will need to provide.

9 Implementation

9.1 Collocation constraints

To solve the optimal control problem we first need to define the collocation constraints (see (29)) at the points between the grid points, $t_{c,j}$. This means we also need to $\mathrm{x}_{1,j+1}$ and $\mathrm{x}_{2,j+1}$ given that we know $\mathrm{x}_{1,j}$ and $\mathrm{x}_{2,j}$ as well as u_j and u_{j+1}. We do this by using the collocation constraints at $\mathrm{t}_{c,j}$:

$$f_1(\mathrm{x}_{1,app}(t_{c,j}), \mathrm{x}_{2,app}(t_{c,j}), t_{c,j}) - \dot{x}_{1,app}(t_{c,j}) = 0,\ j = 1, ..., N-1$$
$$f_2(\mathrm{x}_{1,app}(t_{c,j}), \mathrm{x}_{2,app}(t_{c,j}), t_{c,j}) - \dot{x}_{2,app}(t_{c,j}) = 0,\ j = 1, ..., N-1$$

So, first we use the definition of $\mathrm{x}_{app}(t)$ from (24):

$$\mathrm{x}_{2,\mathrm{app}}(t) = \mathrm{x}_2, j + h_j u_j(\frac{t-t_j}{h_j})$$
$$+ (-3x_{2,j} - 2h_j f_j + 3x_{2,j+1} - h_j u_{j+1})(\frac{t-t_j}{h_j})^2$$
$$+ (2x_{2,j} + h_j u_j - 2x_{2,j+1} + h_j u_{j+1})(\frac{t-t_j}{h_j})^3$$

where $h_j := t_{j+1} - t_j = \frac{\mathrm{t_F} - \mathrm{t_I}}{\mathrm{N}}$

Which leads to

$$\mathrm{x}_{1,\mathrm{app}}(t) = \sum_{k=0}^{3} c_k^j (\frac{\mathrm{t} - \mathrm{t}_j}{h_j})^k,\ \mathrm{t}_j \leq \mathrm{t} < \mathrm{t}_{j+1},\ j = 1, ..., N-1$$
$$\Rightarrow \mathrm{x}_{1,\mathrm{app}}(t) = c_0^j + c_1^j(\frac{\mathrm{t} - \mathrm{t}_j}{h_j})^1 + c_2^j(\frac{\mathrm{t} - \mathrm{t}_j}{h_j})^2 + c_3^j(\frac{\mathrm{t} - \mathrm{t}_j}{h_j})^3$$
$$\Rightarrow \dot{\mathrm{x}}_{1,\mathrm{app}}(t) = \frac{1}{h_j}c_1^j + \frac{2}{h_j}c_k^j(\frac{\mathrm{t} - \mathrm{t}_j}{h_j})^1 + \frac{3}{h_j}c_k^j(\frac{\mathrm{t} - \mathrm{t}_j}{h_j})^2$$

- the collocation constraints at $t_{c,j}$ then lead to

$$f_2(\mathrm{x}_{1,\mathrm{app}}(\mathrm{t}_{c,j}), \mathrm{u}\mathrm{app}(\mathrm{t}_{c,j}), \mathrm{t}_{c,j}) - \dot{\mathrm{x}}_{2app}(\mathrm{t}_{c,j}) = 0,\ j = 1, ..., N-1$$
$$\Rightarrow u\mathrm{app}(\mathrm{t}_{c,j}) = \dot{\mathrm{x}}_{2,\mathrm{app}}(\mathrm{t}_{c,j}),\ j = 1, ..., N-1$$

For x_2 this can be written

$$f_2(u\mathrm{app}(t_{c,j}), t_{c,j}) = \dot{x}_{2,\mathrm{app}}(t_{c,j}) \tag{31}$$

This leads to:

$$\frac{u_{j+1}+u_j}{2} = u_j + 2\frac{1}{2}(\frac{-3\mathrm{x}_{2,j}}{h} - 2u_j + \frac{3\mathrm{x}_{2,j+1}}{h} - u_{j+1})$$
$$+3\frac{1}{4}(\frac{2\mathrm{x}_{2,j}}{h} + u_j - \frac{2\mathrm{x}_{2,j+1}}{h} + u_{j+1})$$

$$= u_j + (\frac{-3\mathrm{x}_{2,j}}{h} - 2u_j + \frac{3\mathrm{x}_{2,j+1}}{h} - u_{j+1})$$
$$+\frac{3}{4}(\frac{2\mathrm{x}_{2,j}}{h} + u_j - \frac{2\mathrm{x}_{2,j+1}}{h} + u_{j+1})$$

$$= u_j + \frac{1}{4}(\frac{-12\mathrm{x}_{2,j}}{h} - 8u_j + \frac{12\mathrm{x}_{2,j+1}}{h} - 4u_{j+1})$$
$$+\frac{1}{4}(\frac{6\mathrm{x}_{2,j}}{h} + 3u_j - \frac{6\mathrm{x}_{2,j+1}}{h} + 3u_{j+1})$$

$$= u_j + \frac{1}{4}(\frac{-6\mathrm{x}_{2,j}}{h} - 5u_j + \frac{6\mathrm{x}_{2,j+1}}{h} - 1u_{j+1})$$

$$= \frac{u_{j+1}+u_j}{2} + \frac{u_j}{2} - \frac{u_{j+1}}{2} + \frac{1}{4}(\frac{-6\mathrm{x}_{2,j}}{h} - 5u_j + \frac{6\mathrm{x}_{2,j+1}}{h} - 1u_{j+1})$$
$$= \frac{u_{j+1}+u_j}{2} + \frac{2u_j}{4} - \frac{2u_{j+1}}{4} + \frac{1}{4}(\frac{-6\mathrm{x}_{2,j}}{h} - 5u_j + \frac{6\mathrm{x}_{2,j+1}}{h} - 1u_{j+1})$$
$$= \frac{u_{j+1}+u_j}{2} + \frac{1}{4}(\frac{-6\mathrm{x}_{2,j}}{h} - 3u_j + \frac{6\mathrm{x}_{2,j+1}}{h} - 3u_{j+1})$$
$$\Rightarrow \mathrm{x}_{2,j+1} = \frac{1}{6}(6\mathrm{x}_{2,j} + 3hu_j + 3hu_{j+1})$$
$$\Rightarrow \mathrm{x}_{2,j+1} = \mathrm{x}_{2,j} + h\frac{u_j + u_{j+1}}{2}$$

And thus we have obtained and expression for $\mathrm{x}_{2,j+1}$ given that we know h, u_j and u_{j+1}. For x_1 the collocation constraints lead to

$$f_1(\mathrm{x}_{2,\mathrm{app}}(t_{c,j}), \mathrm{uapp}(\mathrm{t}_{c,j}), \mathrm{t}_{c,j}) = \dot{\mathrm{x}}_{1,\mathrm{app}}(\mathrm{t}_{c,j}) \tag{32}$$

$$\mathrm{x}_{2,\mathrm{app}}(t_c, j) = \mathrm{x}_{2,j} + \frac{1}{4}(\frac{-6\mathrm{x}_{1,j}}{h} - 5\mathrm{x}_{2,j} + \frac{6\mathrm{x}_{1,j+1}}{h} - 1\mathrm{x}_{2,j+1}) \tag{33}$$

$$\Rightarrow \mathrm{x}_{1,j+1} = \frac{2h}{3}(\mathrm{x}_{2,\mathrm{app}}(t_c) - \mathrm{x}_{2,j}) + \mathrm{x}_{1,j} + h(\frac{5}{6}\mathrm{x}_{2,j} - \frac{1}{6}\mathrm{x}_{2,j+1})$$

Since we already have an expression for $\mathrm{x}_{2,j+1}$, we also know $x_{2,\mathrm{app}}(t_c)$.

9.2 MATLAB implementation

This section will detail the MATLAB implementation of the optimization method. First we define our variables as

$$y = (u_1, x_{1,1}, x_{2,1}, \quad \mathrm{u}_1, \mathrm{x}_{1,2}, \mathrm{x}_{2,2}, \quad ..., \quad \mathrm{u}_N, x_{1,N}, \mathrm{x}_{2,N}, \mathrm{t}_\mathrm{F}) \tag{34}$$

This leads to a total of $3 \cdot N + 1$ variables. In MATLAB I use the function fmincon to implement the optimization method outlined in section 8 . It is defined to

$$\min_x f(x) \text{ such that}$$

$$c(x) \leq 0 \tag{35}$$

$$ceq(x) = 0 \tag{36}$$

$$A \cdot x \leq b \tag{37}$$

$$Aeq \cdot x \leq beq \tag{38}$$

$$lb \leq x \leq ub \tag{39}$$

In the following subsections I go through these constraints one by one to show how they relate to our problem. Note that we do not use equations (37) and (38) in our formulation.

Minimization $f(x)$

We want to minimize

$$f(x) = t_f = y(3 \cdot N + 1) \tag{40}$$

Equation (35): $c(x) \leq 0$

Here, $c(x)$ is the function limiting the torque and speed that is defined by which motor we use. An example is shown as follows:

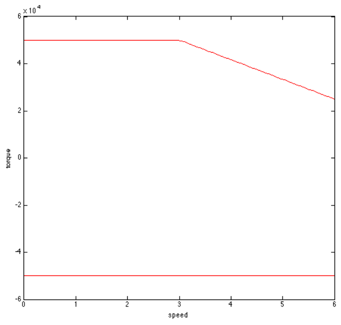

Figure 8: Maximum operational limits for servo motor

We can define this graph by choosing "turning points" in the MATLAB program.

If needed, this function can also be used to limit the jerk by limiting

$$\text{abs}(\frac{u_{i+1} - u_i}{h}) \leq \text{Jerk Limit} \tag{41}$$

Equation (36): $ceq(x) = 0$

The most obvious constraints here are the beginning and end constraints

$$x_{i,0} = x_{i,\text{start}} \quad , \; i = 1...2 \tag{42}$$

$$x_{i,N} = x_{i,\text{end}} \quad , \; i = 1...2 \tag{43}$$

$$\tag{44}$$

i.e. a total of 4 equations. However, we also use the collocation constraints (see further calculations in section 9.1):

$$f_2(i, \text{app}(t_{c,j}), t_{c,j}) = \dot{x}_{i,\text{app}}(t_{c,j}) \quad , \; i = 1...2 \tag{45}$$

which leads to

$$x_{1,j+1} = \frac{2h}{3}(x_{2,\text{app}}(t_c) - x_{2,j}) + x_{1,j} + h(\frac{5}{6}x_{2,j} - \frac{1}{6}x_{2,j+1}), \; j = 1, ..., N-1 \tag{46}$$

$$x_{2,j+1} = x_2, j + h\frac{u_j + u_{j+1}}{2} \; j = 1, ..., N-1 \tag{47}$$

These equations can be written as part of the $ceq(x) = 0$ term as

$$x_{1,j+1} - (\frac{2h}{3}(x_{2,\text{app}}(t_c) - x_{2,j}) + x_{1,j} + h(\frac{5}{6}x_{2,j} - \frac{1}{6}x_{2,j+1})) = 0, \; j = 1, ..., N-1 \tag{48}$$

$$x_{2,j+1} - (x_2, j + h\frac{u_j + u_{j+1}}{2}) = 0, \; j = 1, ..., N-1 \tag{49}$$

which gives us $2 \cdot (N-1) = 2N-2$ more equations, for a total of $2N+2$ equations.

Equation (39): $lb \leq x \leq ub$

These are the lower and upper bounds for our problem. If we have constant limitations on position, speed or acceleration, this is where we put them.

Choice of search algorithm

To get Matlab's fmincon function to use the active-set method described in this paper we use the options:

'Algorithm' = 'active-set'
'Hessian' = 'bfgs'

We also need to set an initial guess which the search algorithm will start from.

10 Numerical Experiments

10.1 Introduction

The main purpose of the experiments is to test the method of optimization on the model previously described in section 5 to show that motion path definition by way of optimization is a viable path for Tetra Pak to explore. The experiment will simulate the torque on the described model and then find a motion path optimized to move as fast as possible from the start position to the end position within the restrictions of the selected motor.

10.2 Experiment Definition

The model that is simulated is the one described in section 4 and pictured in Figure 6 and with the specifications outlined below in Figure 9.

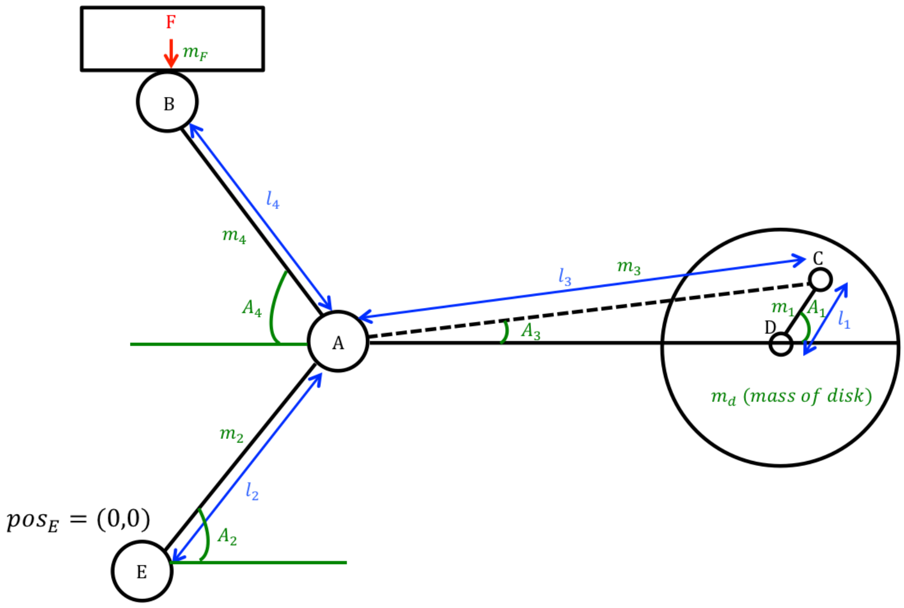

Figure 9: Experiment model

The geometry of the model was used to define a function that takes as input the angle A_1 and in return gives us angles A_2 and A_4 (angle A_2 is the same as angle A_4). With the help of the derivations of the torque from section 5.3 and this function I built a function that simulates the torque at point D from an input consisting of the current angle A_1, angular speed $\dot{A}_1$ and angular acceleration $\ddot{A}_1$. Our method of optimization is then used to obtain a motion path that moves as fast as possible from our start conditions to our end conditions, given the limitations of the servo motor. The angle A_1 will start at $A_{1,\text{start}} = 0$ and end at $A_{1,\text{end}} = \pi$, with the angular velocities $\dot{A}_{1,\text{start}} = 0$ and $\dot{A}_{1,\text{end}} = 0$.

The main goal for the project is to develop a method where the user just specifies the definitions of the model, the start and end conditions and the maximum operational limits for the servo motor as described by the model, and then the method should give us a path that is optimized to go as fast as possible from the start position to the end position.

10.3 Experiment Set-up

We use the following values for the properties seen in Figure 9

$$l_1 = 0.2m \tag{50}$$
$$l_2 = 0.8m \tag{51}$$
$$l_3 = 1.2m \tag{52}$$
$$l_4 = 0.8m \tag{53}$$
$$m_1 = 8.1kg \tag{54}$$
$$m_2 = 32.4kg \tag{55}$$
$$m_3 = 48kg \tag{56}$$
$$m_4 = 32.4kg \tag{57}$$

This was the initial guess provided for the method:

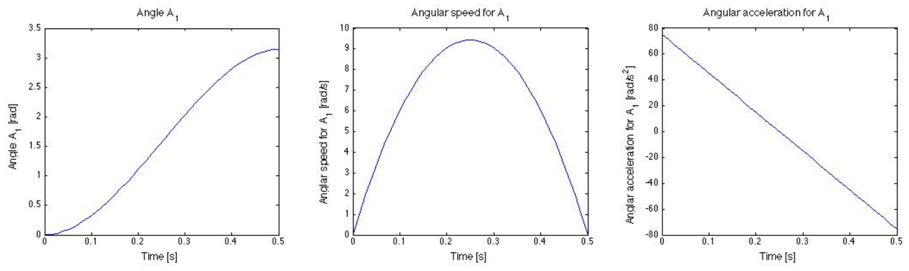

Figure 10: Initial guess

10.4 Verification of Model

Since we did not have any real-life data of the performance of our model we opted to test the model by simulating it in the Multibody Dynamics Simulation Software Adams and comparing the torque on the servo motor obtained for a specific motion path with the one obtained with the Matlab program. The motion path was defined as A_1 having a constant speed of π rad/s. The results were as follows:

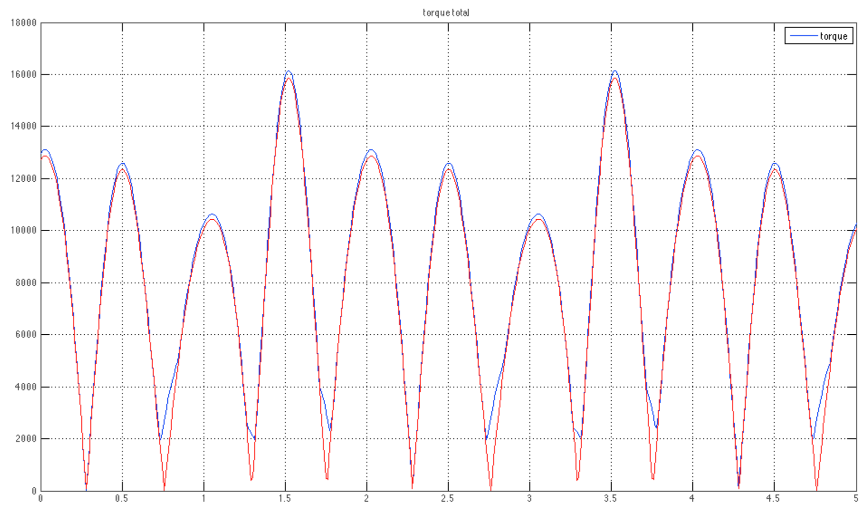

Figure 11: Red curve: Adams, Blue curve: Matlab

There is a slight difference, most visible at the low and high peaks of the curve. These can be explained by rounding errors in our method; when some of the derivatives get very large or very small then small then a tiny rounding error will impact the torque greatly. These results were deemed as satisfactory proof that our function for obtaining the torque was behaving properly.

11 Results

11.1 Experiment Results

The model described in Section 7.2 and the optimization parameters defined in Run A in Section 7.3 are pictured below in Figure 12.

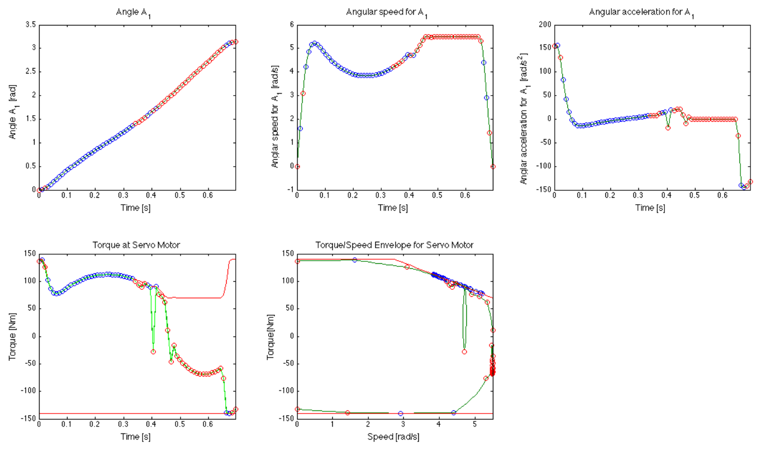

Figure 12: Results

11.2 Analysis

The current method however still needs a fair number of user input in regards to how the optimization method should be used, for example the number of grid points and the different tolerances of the model (see 8.2). The appropriate values for these will differ depending on a number of different factors like the definition of the model that we want to optimize, how long we want the optimization to take, etc. This all means that there is still a fair amount of "trial and error" when finding an optimal motion path.

If we first analyse the motion path for the angle A_1 (see Figure 13 below) we find a motion path that looks pretty smooth and about what you'd expect from an optimized path. There's bit of acceleration in the beginning as the motion finds its max speed, then it is moving forward at a fairly constant speed that is limited by the restrictions on torque defined by the servo motor before it slows down at the end to reach the end position at the specified zero velocity.

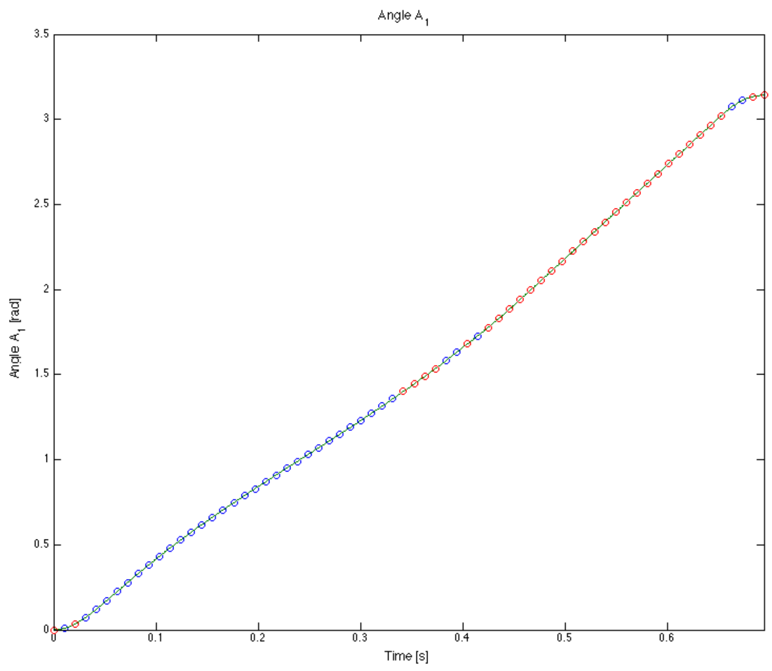

Figure 13: Angle of A˙1

However, just "looking good" is not enough to show that a path is optimal. We need to analyse other aspects of the results as well, like what the speed and acceleration of the path looks like and how the torque and speed fits in the envelope limit defined by the servo motor.

First, let us look at the angular speed of the A_1 angle, i.e. $\dot{A}_1$ (see Figure 14 below).

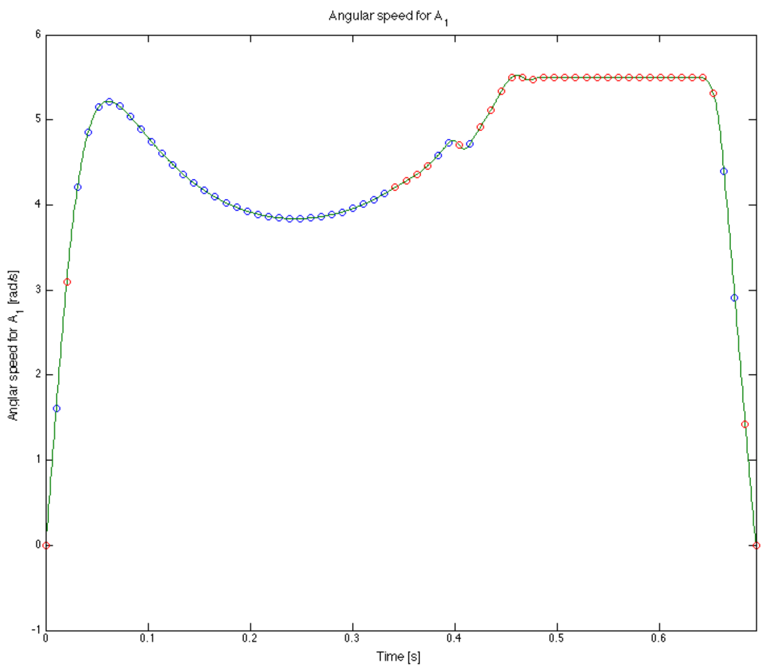

Figure 14: Speed of $\dot{A}1$

The blue circles represent the grid points where the torque hit the envelope limits imposed by the servo motor (to within a 1% difference), while the red circles represent grid points that don't hit torque limits imposed by the servo motor. However, as we will see later when we plot the speed and torque along with the envelope limit, a lot of the red circles are points where the max speed of the servo motor is reached instead. The speed and torque are never reaching beyond the limits of the servo motor, since these are fixed inequality constraints and an inability to stay within them would not result in a solution.

Despite having a number of places where the torque and speed don't reach the limits imposed by the servo motor, the general layout of the angular speed looks about what you'd expect from an optimal solution: there is a large acceleration in the beginning of the motion path where it is mostly the acceleration's contribution to the torque, and not the speed's contribution, that makes the torque hit the limitations of the servo motor. After this initial part the speed is more stable and here the speed obviously makes the major contribution to the torque. The speed does not hit the maximum speed limit of the torque/speed envelope; this is explained by the acceleration contributing too much to the torque for the speed to hit its maximum limit.

We also look the angular acceleration of the A_1 angle, i.e. $\ddot{\text{A}}_1$ (see Figure 15 below).

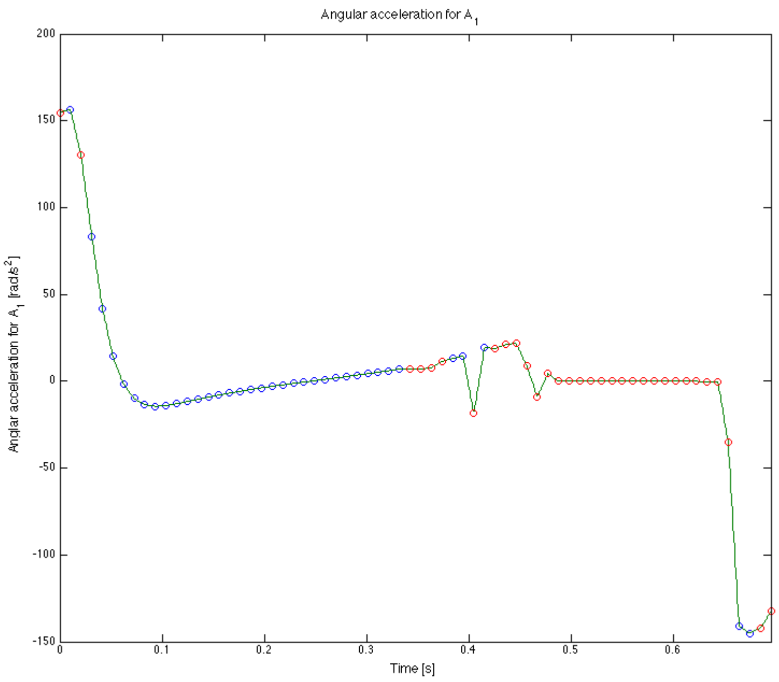

Figure 15: Acceleration of A_1

Again the blue circles represent the grid points where the torque and speed hit the envelope limits imposed by the servo motor, while the red circles represent grid points that don't hit the envelope limits imposed by the servo motor. The acceleration follows a smooth curve when it passes through the blue circles, but it strays from this curve at a few of the red points, where the torque doesn't hit envelope limit (however, at the majority of these red points the curve hits the speed limit instead of the torque limit). The fact that we stray from the envelope limits at these points could be caused by a few different reasons:

The deviations could possibly be explained by the motion needing to make adjustments quickly so as not to go past the envelope limits on the succeeding grid points. If curve had not strayed from the limit at these points, it might have hit later points in the motion profile with a larger speed or acceleration, so it might be more optimized to stray from the points at a few places so that the angle can rotate faster later.

There is also a possibility that we are finding a suboptimal solution that we are converging on. If this is the problem then a possible solution is to change the obtained curve a little by hand and then run the optimization again with that as an initial guess. There is also the possibility of just changing the initial guesses, adding more grid points or lowering the tolerance level for convergence, though this has not resulted in a better result in these numerical experiments.

One possible improvement to the method is adding constraints saying that the difference between the envelope limit and the curve can not be above a certain number. However, in this experiment this has not helped obtain a better solution, and has either led to a suboptimal solution or not converging at all depending on the model parameters , tolerance level and number of grid points used.

One solution to this problem could be to add a penalty function near the limits of the envelope, which could lead to a smoother curve. One negative aspect of this is that it could result in a less predictable Hessian and lead to an inability to find an optimal solution.

To explain why we don't hit the envelope limits at all grid points we also analyse the torque as a function of time (see Figure 16, with the grid points color-coded like the previous graphs in this section).

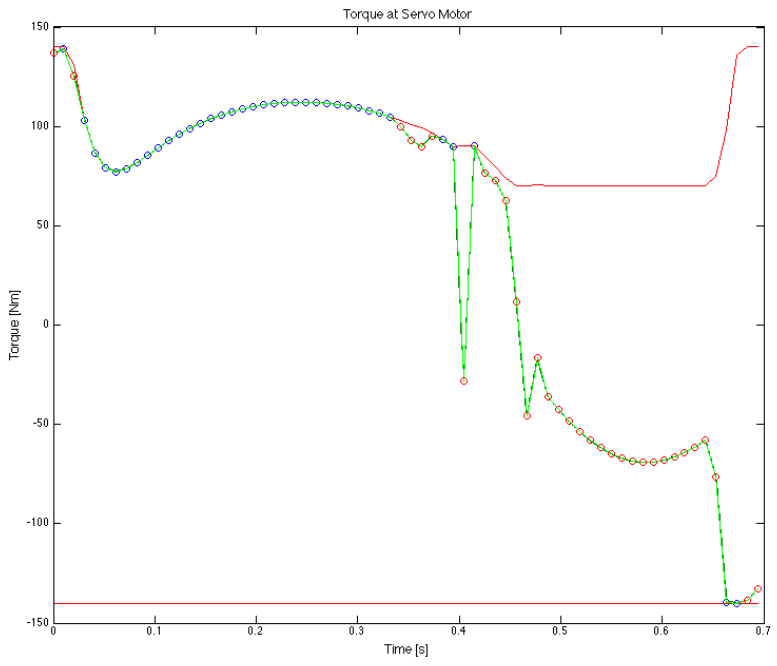

Figure 16: Torque

Here the higher and lower torque limits imposed by the servo motor are drawn as red lines. The graph doesn't tell us too much about the deviations from the envelope limit, except to say that a few of the deviations occur when the torque switches quickly between the maximum and minimum limit.

Finally we study how the the torque and speed fits inside the envelope limit of the servo motor(see Figure 17):

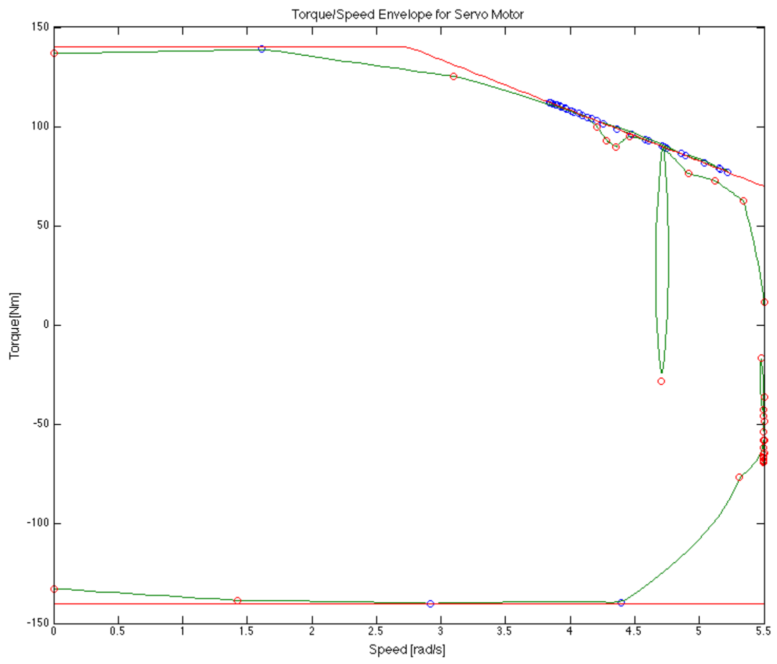

Figure 17: Envelope limit

Here we find that the vast majority of our grid points are within 1% of the envelope limit. It is unclear if the deviations from the torque limit can be explained by the reasons outlined in this section, or if it could be possible to obtain a faster motion-path, making the one we have currently obtained sub-optimal. However, most of the deviations still follow the envelope limits pretty closely and there are just a few major deviations. As such, the motion path that has been obtained can be described as very efficient and it would be very time consuming to achieve a better one using the current iterative process of defining motion paths.

The initial guess, and especially the guess for the end time, was very important in that a bad guess could result in an inability to find a solution, however when a solution was found it always converged to the current solution.

11.3 Comments and comparison to other literature

There are various properties of the model that can be changed: lengths and weights of the components, as well as different limitations on torque imposed by the motor. Variations in these usually leads to similar results as the presented model, though very unrealistic

I have also tried the Adapted Forward-Backward Sweep method described in Optimal Control Applied to Biological Models [4], but I have failed to get the method to converge to a solution with any reasonable tolerance level. This book also mentions the difficulty in finding solutions to minimum time problems with a combination of fixed and constrained states.

The method described in this theses is largely based on "Numerical Solution of Optimal Control Problem" by Oskar Von Stryk [3] as well as theory described in Practical Methods for Optimal Control and Estimation Using Non-linear Programming [1].

I have also looked at the software packages SNOPT [7] and DYNOPT [8] for inspiration on how to solve the optimal control problem.

12 Conclusions

12.1 Work Accomplished

The optimization method described in this thesis can be summarized in these steps:

1. Define the mechanical system where we want to optimize a motion path
2. Define the torque of the mechanical system
3. Verify that the mechanical system and torque are set up correctly
4. Define the Optimal Control Problem, i.e. what we want to optimize and what our restrictions are
5. Discretizing the Optimal Control Problem and transcribing it to a Non-linear Programming problem
6. State the necessary KKT conditions for optimality
7. Linearising the nonlinear constraints and setting up a Quadratic sub-problem
8. Solving the quadratic sub-problem with a Sequential Quadratic Programming method in MATLAB

This method of optimization finds a path that follows the edges of the servo motor's envelope limit for the majority of grid points. While the method in its current implementation is not yet perfect, it finds a faster and better optimized path than could reasonably be obtained by current methods of defining motion paths used by Tetra Pak.

12.2 Unsolved Problems

The main unsolved problem of the method described in this thesis is that we get certain deviations from the torque/speed envelope defined by the servo motor that are not easily explained. These deviations suggest that there is possibility for a better optimization of the motion paths. There is however a possibility that these deviations are needed for the optimal path, for reasons outlined in section 11.2. Possible solutions to these deviations are also explained in section 13.1.

One probable cause to this is that our method converges to suboptimal solution. Possible solutions to this are again outlined in section 11.2.

Another unsolved problem is making the method more generic so that it can easily allow for optimization of motion paths for other mechanical systems, as well as optimizing certain parts of the mechanical systems as well. These problems could hopefully be solved by spending more time on the project, and will as such be discussed in the next section, "Future Research".

13 Future Research

If Tetra Pak wants to continue research on this method these are the main areas that need to be further explored:

- Solving deviations
- Making the method more generic
- Optimizing machine design
- Interfacing to other design tools

They will be discussed individually in the following three sections.

13.1 Solving Deviations

One possible solution to the problem of motion paths deviating from following the servo motor envelope limits making a function that takes a solution given by the optimization method and makes a better starting guess, that is then used to obtain a new solution. For example, one better starting guess could be to either decrease or increase the final time a little bit but keep the general curves of the angle, velocity and acceleration looking similar, to see if this guess returns a better solution.

It could also be possible to define that the motion path can't deviate from the torque/speed envelope too much as part of the optimization limitations; however, in the numerical experiments I have run this has led to the solution converging to a sub-optimal motion path and more research on the subject is needed.

13.2 Generic Method

The optimization method as implemented in this thesis is tailored to fit the model described, but it is of course possible to use the same method for other models and configurations as well. At the moment changing the model to another means the user having to derive and define new functions for the mechanics of the new model, as well as define the torque at the motor.

This can all be done in a fairly straightforward manner, but if Tetra Pak wants to move forward with this optimization method there is probably a need for streamlining this process. One possible solution could be just having the user define one function that describes the mechanics of the new system in a specified way, and one function that defines the torque on the motor. For the model described in this model there are a lot of derivatives defined from manual computation, however these could also be derived numerically which would lessen the need for user input and make it easier to implement the method on new models.

Implementing a way for a user to add the different specific components of a model in a graphical environment in MATLAB as well as decide where the motor should be and defining the motor's limitations on torque and speed, and then having the program derive the necessary equations from this would certainly be possible. The scope of creating this interface would however be significant and further research is needed to know the work needed to implement this.

13.3 Machine Design

One area which this thesis has not delved into is the actual design and definition of the machine that is to be used and simulated. One way to use this optimization method to help design the machine that is detailed in this thesis could be to set, for example, the lengths of the bars L_1, L_2, L_3 and L_4 (see Figure 9) as variables to be optimized as well. There would have to be restrictions on the lengths of the bars as well as a requirement for point B in Figure 9 to have a specific distance between its maximum and minimum positions. Including the machine parameters in the variables that need to be optimized could be help shave off the time needed for the motion path, but it would be at the cost of added complexity on the user input as well as making the optimization take longer.

13.4 Interfacing to other design tools

This area is very important for effective use within Tetra Pak and should inform most decisions on future research. The end goal for Tetra Pak is something that outputs data that can easily be used when actually defining the motion path for the real servo motors, as well as having a method that can be used easily with current hardware. One goal here can be to greatly simplify the derivations of the mechanical system and the torque on the motor, possibly with the help of a visual interface, which obviously ties in deeply with making the method more generic.

Integrating the method with a CAD model would be a good next step, but there is a lot of research needed to integrate this in a convenient way. MATLAB does have the functionality of importing CAD models into its SimMechanics environment, and exporting this model into the the MATLAB workspace and importing would be a possible solution.

13.5 Future at Tetra Pak

The possible future research described in this chapter are all worthwhile endeavors which could benefit Tetra Pak if they choose to continue research in this area. While there aren't any concrete plans on how to continue with this method at the moment, Tetra Pak does have a joint project with Chalmer's University of Technology about optimizing the different operations performed by packaging machines already in production and it is possible that someone there could move this research forward.

14 Appendices

References

[1] John T Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming.* The Boeing Company, Seattle, Washington, 2001.

[2] Björn Bachman, Joakim Lubeck *Kurvgenerering med variationsformulerad optimering för styrning av maskinrörelser.* Institutionen för Datalogi och Numerisk Analys (DNA) LTH, Tetra Pak R&D, Motion Control, Machine Automation, 1995.

[3] Oskar Von Stryk, *Numerical Solution of Optimal Control Problems by Direct Collocation.* International Series of Numerical Mathematics 111, 1993.

[4] Suzanne Lenhart, John T. Workman *Optimal Control Applied to Biological Methods.* Chapman & Hall, Taylor & Francis Group, LLC, 2007.

[5] MATLAB *Mathworks Optimization Toolbox User's Guide.* The MathWorks, Inc., 1990-2013.

[6] Kevin Eric Witzberger *Application of Direct Method Transcription for a Human-Class Translunar Injection Trajectory Optimization.* Master thesis, Department of Aerospace Engineering, The University of Alabama, 2011.

[7] Philip E. Gill, Walter Murray, Michael A. Saunders *SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization.* SIAM Journal on Optimization, Society for Industrial and Applied Mathematics Vol. 12, No. 4, pp. 979–1006, 2002.

[8] M. Cizniar, D. Salhi, M. Fikar, M.A. Latifi *DYNOPT - Dynmic Optimization Code for MATLAB.* http://www.kirp.chtf.stuba.sk/∼fikar/research/dynopt/dynopt.htm, Department of Information Engineering and Process Control, FCFT-STU, Laboratoire des Sciences du Genie Chimique, CNRS-ENSIC.

14.1 Appendix - Derivatives

Below are the calculations used for calculating the torque, $M_\theta = \frac{d}{dt}(\frac{dL}{d\dot{A}_1}) - \frac{dL}{dA_1}$, that is contributed by the angular kinetic energy and translational kinetic energy of bar 2. The torque contributed from the gravity and the rest of the model are calculated in the same way.

$$I_2 = \frac{m_2 \cdot \mathrm{l}_2^2}{12}$$

$$J_2 = \frac{1}{2}(m_2 \mathrm{l}_{c2}^2 + I_2)$$

$$\begin{bmatrix} \dot{A}_2 \\ \dot{A}_3 \end{bmatrix} = \begin{bmatrix} \frac{l_1(sin(A_1-A_3)}{l_2 sin(A_2-A_3} \\ \frac{l_1(sin(A_1-A_2)}{l_3 sin(A_2-A_3} \end{bmatrix} \cdot \dot{A}_1 = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \cdot \dot{A}_1$$

$$L = J_2 S_1^2 \dot{A}_1^2$$

$$\frac{dL}{d\dot{A}_1} = 2J_2 S_1^2 \dot{A}_1$$

$$\frac{d}{dt}(\frac{dL}{d\dot{A}_1}) = 2J_2 S_1^2 \ddot{A}_1 + 2J_2 S_2(\frac{dS_1}{dA_1} + \frac{dS_1}{dA_2} \cdot \frac{dA_2}{dA_1})\dot{A}_1^2 = 2J_2 S_1^2 \ddot{A}_1 + 2J_2 S_2(\frac{dS_1}{dA_1} + \frac{dS_1}{dA_2} \cdot S_1)A_1^2$$

$$\frac{dS_1}{dA_1} = \frac{l_1(\cos(A_1 - A_3)}{l_2 \sin(A_2 - A_3)}$$
$$\frac{dS_1}{dA_2} = -\frac{l_1 \sin(A1 - A3)\cos(A2 - A3))}{l_2 sin(A_2 - A_3)^2}$$
$$\frac{dS_1}{dA_3} = -\frac{l_1 cos(A1 - A3)}{L2 \sin(A_2 - A_3)} + \frac{L1 \sin(A1 - A3)\cos(A2 - A3)}{L2 \sin(A2 - A3)^2}$$

$$\frac{dL}{dA_1} = [2J_2 S_1(\frac{dS_1}{dA_1} + \frac{dS_1}{dA_2} \cdot \frac{dA_2}{dA_1} + \frac{dS_1}{dA_3} \cdot \frac{dA_3}{dA_1})] \cdot \dot{A}_1^2 = [2J_2 S_1(\frac{dS_1}{dA_1} + \frac{dS_1}{dA_2} \cdot S_1 + \frac{dS_1}{dA_3} \cdot S_2)] \cdot \dot{A}_1^2$$

14.2 Appendix - Variable names

$\mathrm{l}_1, \mathrm{l}_2, \mathrm{l}_3, \mathrm{l}_4,$: lengths of bars
m :
$\mathrm{l}_1, \mathrm{l}_2, \mathrm{l}_3, \mathrm{l}_4$: Angles of model
$\mathrm{A}_1, \mathrm{A}_2, \mathrm{A}_3, \mathrm{A}_4$: Angles of model
$\mathrm{s}_1, \mathrm{s}_2$: relationships between angles and their derivatives, according to:

$$\begin{pmatrix} \frac{\mathrm{l}_1 \sin(A_1 - A_3)}{\mathrm{l}_2 \sin(A_2 - A_3)} \\ \frac{\mathrm{l}_1 \sin(A_1 - A_2)}{\mathrm{l}_3 \sin(A_2 - A_3)} \end{pmatrix} \cdot \dot{A}_1 = \begin{pmatrix} S_1 \\ S_2 \end{pmatrix} \cdot \dot{A}_1$$

I : Moment of Inertia
f : Collocation constraints
t : Time
$\mathrm{t}_{c,j}$: $/varT_{c,j} := /varT_{j+1/2} := (/varT_j + /varT_{j+1})/2$, $j = 1, ..., N$
f :
$\mathrm{h}(\mathrm{t_F})$: time interval between points, $\mathrm{h}(\mathrm{t_F}) = \frac{\mathrm{t_F} - \mathrm{t_I}}{\mathrm{N}}$
m : Mass
M_θ : Torque
N : Number of grid points
L : Lagrangian (Classical Mechanics), $\mathrm{L} = T - U$
$\mathrm{L}(\mathbf{Y}, \lambda)$: Lagrangian (cost function), $\mathrm{L}(\mathbf{Y}, \lambda) = \Phi(x_N, t_n) + \sum_{j=1}^{m} \lambda_j g(x_j, u_j, t_j)$
T : Kinetic energy
U : Potential energy
g : Boundary conditions ($\mathrm{g}_i[\mathbf{y}(\mathrm{t_F}), \mathrm{u}(\mathrm{t_F}), \mathrm{t_F}] = 0$ when $i = 1, ..., m_e$) and Inequality constraints ($\mathrm{g}_i[\mathbf{y}(\mathrm{t}), \mathrm{u}(\mathrm{t}), \mathrm{t}] \leq 0$ when $m_e + 1, ..., m$)
$\mathrm{t_I}$: The initial time (0 in our case)
$\mathrm{t_F}$: The final time
$\phi[\mathbf{y}(\mathrm{t_F}), \mathrm{t_F}]$: The functional we want to minimize, $\phi = \mathrm{t_F}$ in our case

$q(t, \mathrm{x}_1, \mathrm{x}_2, u)$: Limit on torque imposed by the motor

$\mathrm{x}_1(\mathrm{t}_j)$: Angle $\mathrm{A}_1(\mathrm{t})$

$\mathrm{x}_2(\mathrm{t}_j)$: $\dot{\mathrm{A}}_1(\mathrm{t})$

$\mathrm{u}(\mathrm{t}_j)$: control variable, $\mathrm{u}(\mathrm{t}) = \ddot{\mathrm{A}}_1(\mathrm{t})$

$\mathrm{u}_{\mathrm{app}}(\mathrm{t})$: Approximation of u at t

$\mathrm{x}_{1,\mathrm{app}}(\mathrm{t})$: Approximation of x_1 at t

$\mathrm{x}_{2,\mathrm{app}}(\mathrm{t})$: Approximation of x_2 at t

$\mathbf{y}$: State variables, $\mathbf{y} = [\mathrm{x}_1, \mathrm{x}_1]$