Master of Science Thesis in Automatic Control
Department of Electrical Engineering, Linköping University, 2020

Development of a 2D Optimal Path Simulation for Ship-to-Shore Cranes

Safe Trajectories within Interchangeable Obstacle Environments

Rickard Green

LINKÖPING
UNIVERSITY

Master of Science Thesis in Automatic Control

Development of a 2D Optimal Path Simulation for Ship-to-Shore Cranes: Safe Trajectories within Interchangeable Obstacle Environments

Rickard Green

LiTH-ISY-EX--20/5280--SE

Supervisor:	**Viktor Leek**
	ISY, Linköpings universitet
	Pontus Klang
	ABB Marine & Ports
Examiner:	**Lars Eriksson**
	ISY, Linköpings universitet

Division of Vehicular Systems
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping, Sweden

Abstract

The most advanced ports as of writing of this report are at least somewhat autonomous. Whether discussing the transporters between crane and stack (temporary storage) or cranes, the ports are shifting into a completely autonomous system. This ultimate goal presents a challenge in regards to unloading and loading cargo ships in the harbour. How do you achieve unloading of a ship without human intervention while still guaranteeing secure trajectories for the containers?

ABB Ports in collaboration with the Division of Vehicular Systems at Linköping University have developed a simulation that utilises a simple control model to investigate the behaviour, limitations and capabilities of such an autonomous crane. Specifically, this simulation utilises a model of the dynamics of a Ship-to-Shore crane (STS), which has the task of unloading a ship. In order to set the crane model in context of realistic scenarios, some additions to the simulation are needed.

One of these additions is obstacles. Before this thesis work, the model enjoyed an empty simulation environment to freely optimise how quickly the containers could be transported off of the ship. The addition of obstacles in the form of other containers on the cargo ship as well as the physical presence of the crane's legs presents new challenges for the optimiser used to solve the optimal control problems formulated through the model in the simulation. The implementation of obstacles is one of the objectives for this thesis. This addition was implemented by modeling the obstacle dimensions and ship limitations by looking at the largest container ships in the world.

Due to the simulation not containing obstacles previous to this thesis work, the initial guess provided to the solver initialised the solving in an area of convergence that is unfair to the solver, This rendered the simulation useless, as any obstacle presented to the solver would generate an infeasible solution.

Another functionality needed for the obstacle implementation to be meaningful is a solution for guaranteeing safe trajectories for the containers from ship to shore. The solution utilised to reach this goal was to combine a convex hull and safety conditions where the convex hull covers the obstacles, including some padding to prevent collisions between the container carried (load) and obstacles. The safety conditions however calculates the potential positions of the load when an emergency stop occurs, and therefore can prevent the load from swinging into obstacles if there is an emergency stop.

These implementations however changes the usefulness and performance of the simulation because of how they shrink the area of convergence for the solver and making some problems non-solvable. When considering both a convex hull and safety conditions, the usability of the simulations is harmed, but can still be utilised to learn about autonomous performance of the simulation. The optimal solutions include some interesting characteristics that can learn crane operators about how the control systems can be utilised.

Such a simulation would benefit from continuous development in order to investigate further technologies and features that could improve both performance and usability. Areas such as homotopy, modelling ropes, comparison between simple and nuanced model would be truly interesting for future areas of investigation.

Acknowledgments

First, I would like to thank my supervisor Viktor Leek from the Division for Vehicular Systems at Linköping University for his active participation and support throughout this thesis work.

I would also like to thank both Lars Eriksson from the Division for Vehicular Systems at Linköping University and Pontus Klang from ABB Ports for their patience and support.

Stockholm, February 2020
Rickard Green

Contents

1 Introduction

1.1 Background

The most advanced ports in the world are using crane systems that are at least somewhat autonomous, while being supervised by personnel as well as controlled via static methods that do not change for each instance of, for instance, unloading or loading a cargo ship. Other aspects of the port, such as transportation between the cargo ship and the stack for intermediate storage in the port, are almost entirely autonomous. In worst case, these solutions are not sufficiently robust, and still need supervision by personnel. This would mean that an autonomous initiative would be unnecessary and expensive. At best however, efficiency would be close to optimal and costs for the entire operation will be sharply reduced compared to an entirely man powered port.

When looking at ports that are still very much analogous (compared to the ports discussed in the previous paragraph), the value of such an autonomous system is less clear. Whilst the development of an autonomous port might seem like a tempting future, it might not yield a robust system and therefore there would still need to be a lot of oversight and human intervention in the ship-to-shore cranes' operations. So, whilst an autonomous theoretically could support optimal performance, both from a time and energy standpoint, maybe leaning from autonomous systems are more applicable.

When comparing the two ports discussed above, there are obvious differences in both capacity when it comes to loading and unloading, but also in technical advancement. The semi-autonomous STS cranes will give ports most of the benefits of an autonomous system, whilst securing safe trajectories through full hoisting of the load before transporting it laterally. However, because of the transitions between autonomous control and automatic control (vertical to lateral to vertical) this method lacks optimal performance in both time and energy consumption.

Because of this, guaranteeing safe trajectories in autonomous trajectories are of interest. Looking at manual and or automatic steering of STS cranes instead, the implementation of an autonomous system might not be beneficial or nor realistic. In that regard, value could be gained by learning from autonomous simulations for STS cranes in order to optimise the manual or automatic steering of the cranes going forward.

So, from a technical and business perspective, there are benefits to investing in development of either autonomous steering systems for STS cranes or at least investigate the performance of such a system. Such a system and its capabilities and performance could be investigated through the development of a simulation of an STS crane containing real life scenarios and the major complications those scenarios bring to an autonomous system.

Such a simulation has been developed for some time in the division for Vehicular Systems at Linköping University in cooperation with ABB Ports. This simulation is developed using open source software for interpreting (YOP) the input of the user, and formulating (CasADi) the optimal control problem for a solver (IpOpt) of this problem. Further development of this simulation is the core focus of this thesis.

1.2 Purpose

Due to the continuous focus on optimizing both in dimensions of time and power consumption, a simulation of the entire trajectory for Ship-to-Shore crane cargo is at worst a handy tool for learning more about behaviour of an entire autonomous system. At best however, a sufficiently secure and stable simulation may lay the groundwork for an entirely optimised trajectories, instead of the semi autonomous solutions that seem to be the norm for the most advanced port systems today. These systems often involve autonomous travel when the cargo has been lifted completely over the rest of the ships cargo. This method is a widespread solution for the more advanced ports, as it both reduces the human interaction needed as well as delivers a solution that is consistent over long periods of time. This method however lacks the nuance of changing the trajectory from instance to instance. Such a solution would need consideration of the start position of the cargo as well as the position of the surrounding cargo in order to provide a safe and fast trajectory. The difference in initial position for the autonomous trajectory is key in this instance since this seems to be a problematic area in this specific use case for autonomous drive.

The purpose of this master's thesis is to provide such a solution that both can provide a trajectory before and after the cargo is hoisted above the other containers, as well as connecting these solutions the already used solution. It is of course also of highest interest to guarantee safe trajectories as this would elevate the reliability and hopefully usefulness of the software. This might entail some changes to the foundation of the simulation in its current state, as well as other areas that might need addressing. Do note however that the groundwork for this simulation already has been laid. This means that the purpose of this thesis will be dependant on the work done beforehand.

1.3 Problem Formulation

The following points are meant to express the aim of this thesis work;

- Can easily re-configurable obstacles of surrounding containers on a cargo ship be implemented into the simulation in a way that doesn't harm the performance or usefulness of the software?
- Can optimal trajectories consistently be reached in context of the obstacle profile implemented?
- If container dimensions and emergency breaking is considered, can safe trajectories be guaranteed in context of the obstacle profile?

1.4 Method

As explained in previous sections, the tasks of this thesis are neither of quantitative nor qualitative comparisons. Instead, this thesis is quite simply put software development. therefore, no predetermined method is set. Instead, the method is highly experimental and will adjust to the work needed in order to answer the problem formulations above.

1.5 Literature study

In the interest of investigating how others have implemented obstacles and handled them in in optimal path simulations, the following was found. Blackmore, Li and Williams (2006) researched a probabilistic approach to optimal path planning which resulted in varying solutions depending on the pre-decided maximum probability of collision.[5] The study resulted in a conservative model as shown by their conservatism factor.[5] In their implementation a secondary focus on fuel consumption was also presented which clearly presented a pay-off between lower maximum collision probability and fuel consumption.[5] In 2011, Blackmore, Ono and Williams present another method to optimal path with collision-based probability as their focus. [12] Bounds for maximum collision probability are in this case set so that the optimal cost has limits, while changing along the probability.[12] The findings of this study are in terms of maximum probability of collision similar.

Bergman and Axehill (2018) presents a systematic approach to motion planning in non-convex environments.[3] This approach includes relaxing the original problem and from there numerically gain solutions to problems resembling the original one, after which you can easily solve the original problem.[3] Zhao et al. (2019) presents an approach to solving energy-optimal motion planning with collision-free conditions. This approach is based on convexifying a non-convex problem and thereafter solving the problem using alternating quadratic programming until convergence has been reached.[15] Hämäläinen et al. (1995) as well as Sakawa and Shindo (1982) each present their own optimal control based on a five-section model where their methods differ such that Hämäläinen et al. (1995) uses a numerical method inspired by multiple sources while Sakawa and Shindo (1982) uses a combination of analytic tools to arrive at theirs.[7, 13]

There are several categories of control techniques for over head, or gantry cranes presented in Eihab et al. (2001).[1] Even though, at the writing of this thesis report, Eihab et al. (2001) is 18 years old, this article will be used as a point of reference for previous work in both optimal control techniques as well as other types of control.

The first article to mention a concrete strategy to control the pendulums caused by the motion of an overhead, gantry or STS crane seems to be Alsop et al. (1965).[6] This was achieved using a controller that accelerated the trolley in steps while also stopping the trolley in-between the acceleration, in order to reset the position of the container.[6] This method and many developed versions of it fall under the category of *Input-Shaping*. This category of methods however might not be of use in this project. Instead, *Optimal Control* is more akin to the task at hand and is certainly applicable to this problem. In this case, the first to propose an automatic controller or strategy for controlling a gantry crane was Field (1961).[9] He used trial and error to minimize the travel time, however he did not try to control the pendulous movement that Alsop et al. (1965) had focused their work on.[6, 9] Since then, many attempts have been made to create an optimal control strategy, while taking more and more parameters into consideration. Beestion (1969) was able to minimize travel and hoisting time under the consideration of Pontryagin's maximum principle.[11] The very same principle was later used by Karihaloo and Parberry (1982) to eliminate pendulum movement for a given solution to the optimisation of travel time and distance, that is a single solution.[4] In this case, the minimisation of pendulous movement was an after thought which segmented this solution into two parts.[4] Hämäläinen et al. (1995) then defined the trajectory of the container as a five-stage sequence.[7] Their solution to this formulation of the problem was then solved by trial and error.[7]

After citing these examples and more, Eihab et al. (2001) then mentions the limitations of both optimal control and input shaping techniques.[1] They are extremely sensitive to the parameter and nominal values as well as the fact that a change in the initial conditions may require "highly accurate values of the system parameters" to achieve satisfactory system response.[2, 8, 10]

1.6 Thesis outline

Due to the experimental nature of this thesis work, the chapters of the thesis are arranged in chronological order. This is due to the contents of some chapters are completely dependent on some of the content of other chapters. This is not true for all chapters however it represents the work done in an order that will make more sense. Chapter 2 consists of a description of the model and simulation in its initial state. Chapter 3 will touch on initial solution and some of the early work needed in the simulation. Chapter 4 will then present the findings or results of the thesis work. This chapter includes the construction of the obstacle profile, the convex hull and the safety conditions. Chapter 5 contains a short discussion regarding the behaviour of the simulation, as well as the conclusion and recommendations for future work.

Note that the report contains a lot of information regarding the thesis work and it needs to be presented somehow. It might seem that the report is segmented and not written to read, and that is correct. This report is written as both a documentation and instruction for further use and development of the simulation.

2 Model description

This chapter has the purpose of presenting the model used in the simulation so to create a context for the rest of the report. The rest of the report will refer to this chapter in language and conclusions when discussing performance and capabilities.

2.1 Crane model

The model that previously has been implemented in this simulation follows the characteristics of the *Lumped-Mass Models* according to Eihab et al. (2001).[1] More precisely, the model subscribes to *The Reduced Model*, however, the elasticity in the rope that is incorporated in the model presented in Eihab et al. (2001) is not present in this model.[1] Also, it's important to note that the simulation is set in two dimensions. This means that problematic swaying or tilting of the cargo is not considered in the model. This is both due to the lack of a third dimension to consider, but also since there is an assumption regarding the mass of the cargo. Namely, its centred in the container, or in dynamical terms, the container is considered a point mass. See figure 2.1 for an understanding of the model. It should however be mentioned that this figure is sourced from Sjöberg (2018) and that the notations for the variables are not correct for this report. [14]

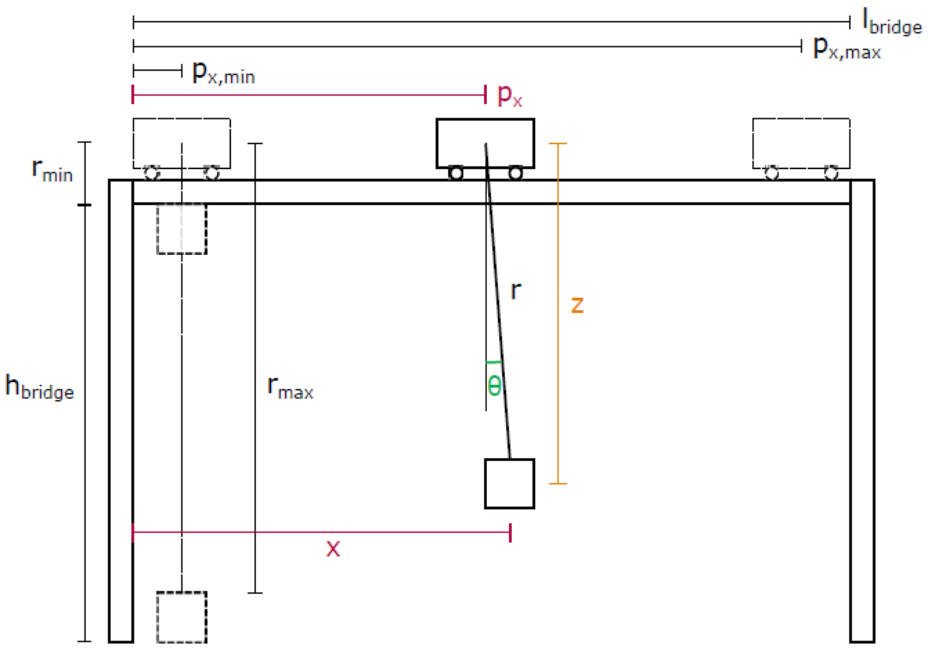

Figure 2.1: *2D model for a STS-crane with rope as solid body and load as point mass Sjöberg (2018) [14]. Please note that the notations for the variables are not correct for this report.*

The model is complemented in the simulation by an objective function which the simulation works towards optimizing under the rules of the model. In the first version of the simulation, the objective function had the following shape;

$$\phi = \min t_f \tag{2.1}$$

Further, the model contains the states;

- $x_1 = x_t$: The lateral coordinate (x-axis) of the trolley.
- $x_2 = v_t$: The velocity of the trolley along the horizontal axis (x-axis).
- $x_3 = l_r$: The length of the rope between the trolley and the load (container).
- $x_4 = v_r$: The velocity of the rope between the trolley and the load (container). This refers to the velocity of retracting or extending the rope.
- $x_5 = \theta$: The angle of the rope between the trolley and load, as measured from the rope's position at rest (vertical).
- $x_6 = \omega$: The angular velocity of the rope between the trolley and the load, as measured from the rope's position at rest (vertical).

So, for this model the position of the load (container) is given by

$$x = x_t + l_r sin(\theta) \tag{2.2}$$

$$z = l_r cos(\theta), \tag{2.3}$$

Where x is the lateral position of the load (container) and z is the vertical position. Also, see the model states for definitions of l_r and θ.

The model has two inputs, or control signals which are as follows;

- $u_1 = a_t$: The acceleration of the trolley (fixed to x-axis).
- $u_2 = a_r$: The acceleration of the rope when either shortening or extending.

So, to sum up the notation of this model, these are listen in table 2.1.
The kinematics of the system are described by the model, which consists of the following non-linear differential equations in terms of inputs and state variables;

$$\begin{array}{ccc} \dot{x_1} & = & x_2 \\ \dot{x_2} & = & u_1 \\ \dot{x_3} & = & x_4 \\ \dot{x_4} & = & u_2 \\ \dot{x_5} & = & x_6 \\ \dot{x_6} & = & \frac{-2v_r x_6 - a_t cos(x_5) - g sin(x_5)}{x_3} \end{array} \tag{2.4}$$

Symbol	Unit	Explanation
x	[m]	The lateral position of the load
z	[m]	The vertical position of the load (measured from the boom down)
x_t	[m]	The lateral position of the trolley
v_t	[m/s]	The lateral velocity of the trolley
v_r	[m/s]	The velocity of the rope(s) as it extends and retracts
θ	[rads]	The angle between the rope and the z-axis (ropes position at rest)
ω	[rads/s]	The angular velocity of the same angle as for θ
$u_1 = a_t$	$[m/s^2]$	The lateral acceleration of the trolley
$u_2 = a_r$	$[m/s^2]$	The acceleration of the rope as it extends and retracts

Table 2.1: *Table of notation for the variables of the model described in this chapter.*

These equations can also be formulated using the variable names used henceforth in this report, which are also easy to remember variable notations. The only equation that is interesting to reformulate once more is the following one;

$$\dot{\omega} = \frac{-2v_r\omega - a_t cos(\theta) - g sin(\theta)}{l_r} \tag{2.5}$$

These equations are of course accompanied by some restrictions of both the variable values, but also other parameters presented below.

- $x_{initial}$: The initial lateral coordinate (x-axis) of the load (container) at the initial position.
- $z_{initial}$: The initial vertical coordinate (z-axis) of the load (container) at the initial position.
- x_{final}: The final lateral coordinate (x-axis) of the load (container) at the terminal position.
- z_{final}: The final vertical coordinate (x-axis) of the load (container) at the terminal position.
- x: The lateral coordinate (x-axis) of the load (container).
- a_z: The vertical acceleration of the load (container).

These constraints are needed for the optimisation to not behave outlandish. That is, absurd velocity, acceleration or angles must be controlled. These restrictions will also be discussed further into the report, in the relevant sections. There are three types of restrictions implemented in the simulation, Initial, Terminal and Box constraints.

- **Initial constraints**

$$\begin{aligned} v_t &= 0 \\ a_t &= 0 \\ v_r &= 0 \\ a_r &= 0 \\ \theta &= 0 \\ \omega &= 0 \\ x_{t_{initial}} &= x_{initial} \\ l_{r_{initial}} &= z_{initial} \end{aligned} \tag{2.6}$$

- **Terminal constraints**
  Note that $x_{c_{final}}$ and $z_{c_{final}}$ are the values of the coordinates for the terminal position.

$$\begin{aligned} x_{final} &= x_{c_{final}} \\ z_{final} &= z_{c_{final}} \end{aligned} \tag{2.7}$$

- **Box constraints**

$$\begin{array}{rcl} 0 \leq & x_t & \leq 120 \\ -4 \leq & v_t & \leq 4 \\ -0.8 \leq & a_t & \leq 0.8 \\ 20 \leq & l_r & \leq 50 \\ -3 \leq & v_r & \leq 3 \\ -0.6 \leq & a_r & \leq 0.6 \\ -\frac{\pi}{4} \leq & \theta & \leq \frac{\pi}{4} \\ -\frac{\pi}{8} \leq & \omega & \leq \frac{\pi}{8} \\ 0 \leq & x_c & \leq 130 \\ -\infty \leq & a_{c_z} & \leq 9.81 \end{array} \tag{2.8}$$

The values that are given in the constraints are either provided by previous similar studies, personnel at ABB Ports, estimated or restricted due to some unstable behaviour of the model. This will be touched on through out the report.

Note that this description is of the initial model that will be further developed into the model used currently. This model is therefore part of the groundwork for this thesis and will be a minor focus point in this report. However, in this section, one thing not discussed in depth is the initial guess of the solution for the optimisation. This is due to the subject being a major part of the workload during the project and will instead be presented and discussed in chapter 3 - Initial guess. However, a quick presentation of the initial guess as well as the optimal solution is found below.

The initial guess for this version of the model is lacking consideration for obstacles and is very simple. Its only objective was minimising the time from start to finish for the load or cargo. This meant that the initial solution looked something like what is presented in figure 2.2 below.

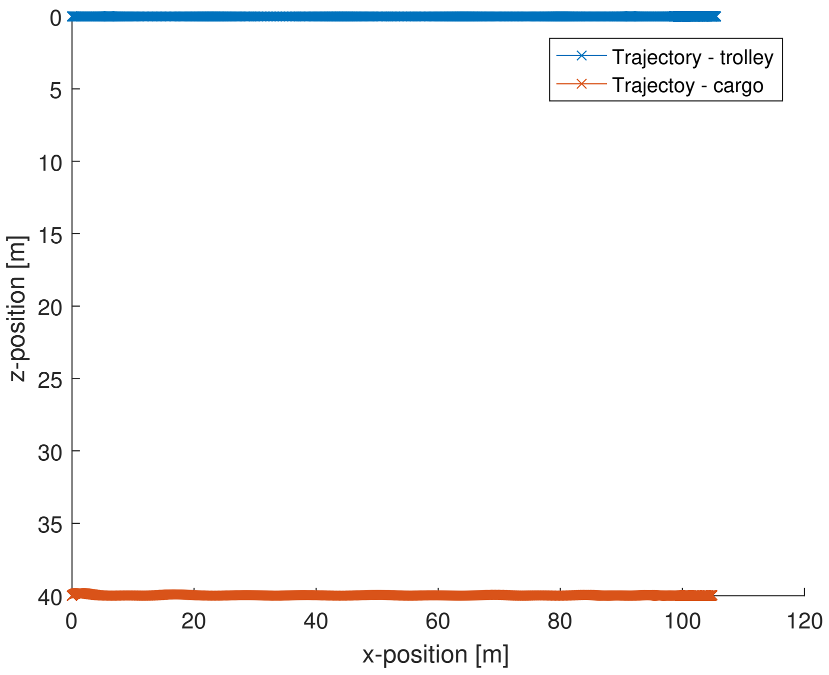

Figure 2.2: *Initial solution for the simulation in its initial state. Note that there is no to little elevation of the cargo during transportation.*

As seen in figure 2.2, this initial solution will be of little use when there are several obstacles in the way of this trajectory. This will be further elaborated in in chapter 3.

It is also important to have the perspective of the optimal solution of the initial model, when considering the results of this thesis. Therefore, in figure 2.3, 2.4 and 2.5 this will be presented.

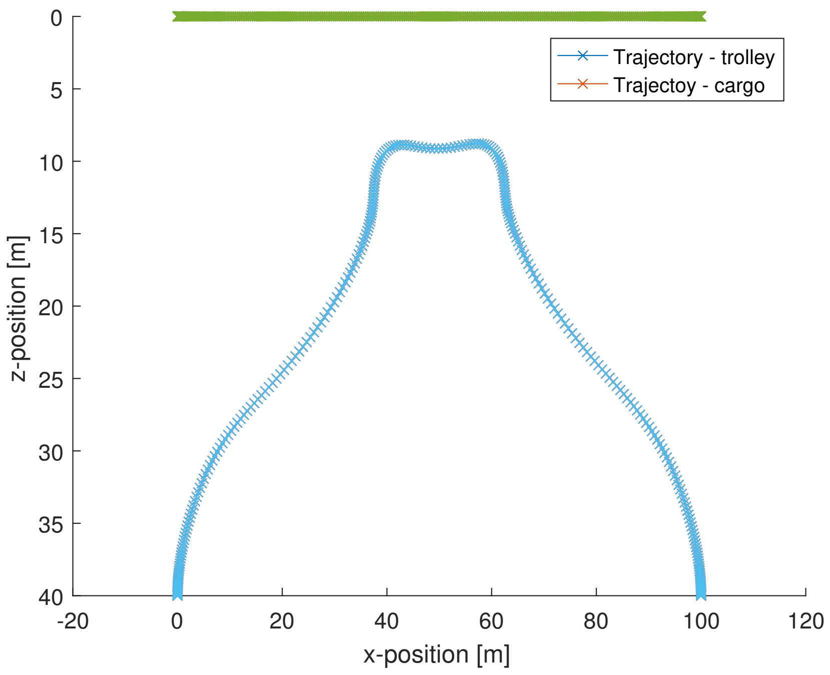

Figure 2.3: *Trajectory of the load (container) for the optimal solution with the model in its first version.*

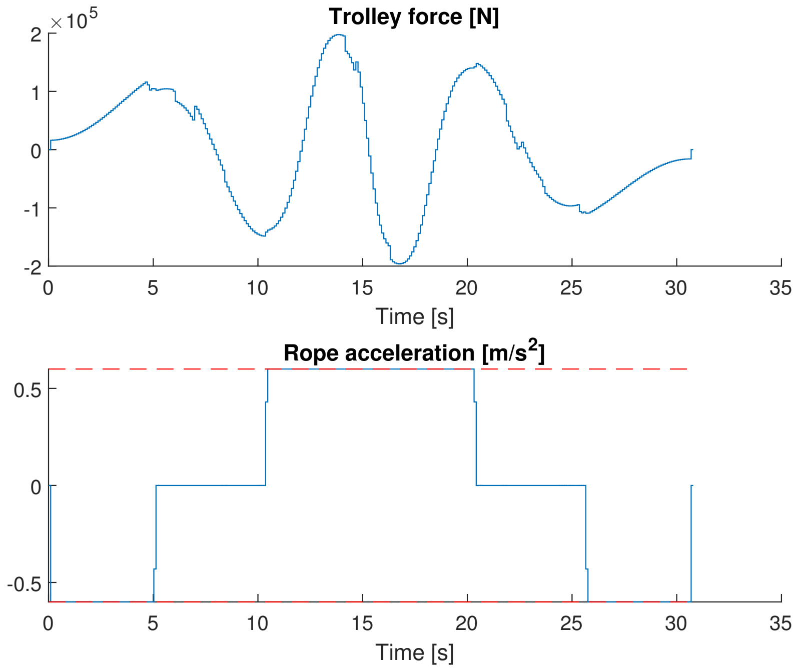

Figure 2.4: *Model inputs*
a) Trolley force [N] for th e complete run time of the optimal solution of the simulation in its initial state.
b) Rope acceleration [m/s^2] for the complete run time of the optimal solution of the simulation in its initial state.

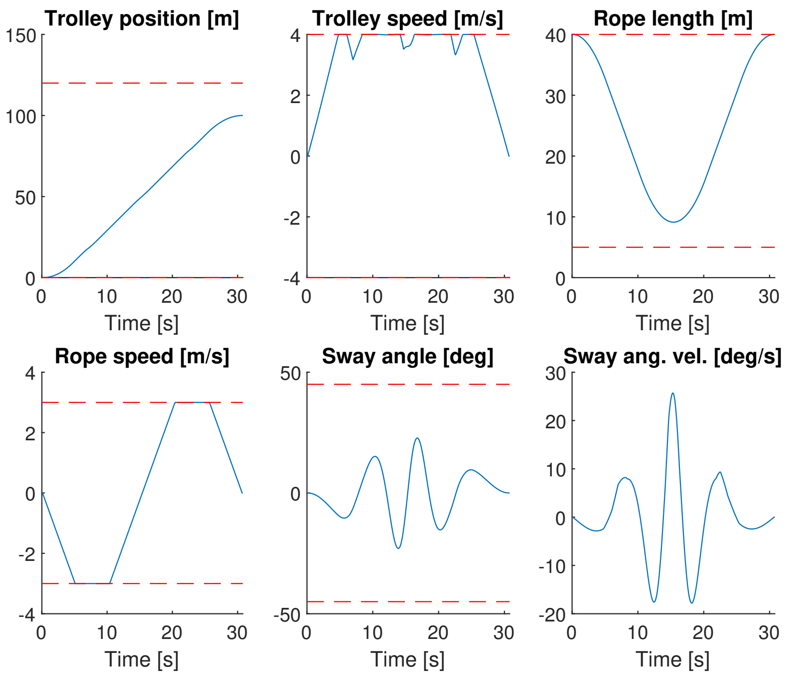

Figure 2.5: *The six states of the model for the entire run time of an acceptable solution of the simulation in its initial state.*

Note that the initial guess of this version of the model differs quite drastically from the optimal solution in how high the load is hoisted. This is not a problem, since the constraints for the model are much looser than will be presented later in this report. These constraints also mean that the solver can iterate and arrive at a solution to the optimisation problem in a matter of a few hundred iterations. This number of iterations is not unusual for this iteration and would in some cases be perceived as solid performance. This will prove a problem area later in the project and will be presented as a part of later chapters in this report.

3 Initial solution

Because of the iterative behaviour of the optimiser in the simulation, it was discovered that the initial guess would be a crucial part of the simulation and therefore the optimisation. It would also become clear that the initial guess could make up the difference between a solution being found in less than 200 iterations and not finding a solution at all. This points to the initial guess being of great importance to the optimisation, performance and therefore the usability of the simulation.

In order to accurately recount the iterative process that was the initial guess, this chapter is split into three sections. These will separately present and explain the different iterations that the initial guess went through.

3.1 First iteration and the problems it caused

As repetition, figure 3.1 below shows the initial guess of the first iteration of the model, as presented in chapter 2.

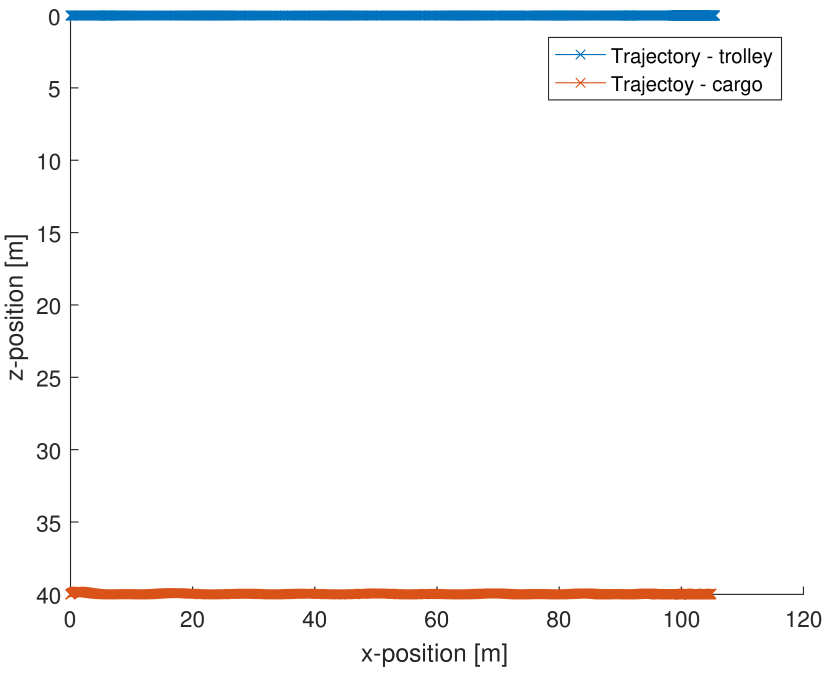

Figure 3.1: *Initial solution for the simulation in its initial state. Note that there is no to little elevation of the cargo during transportation.*

In chapter 2, the problem with the first iteration of the initial solution was discussed briefly. This does however not cover the entirety of the realisation of what the initial solution meant for the implementation of obstacles. During the first try of implementation of an obstacle in the trajectory of the container was successful, but with a huge asterisk. This was of course the sensitivity of the placement and height of the obstacle. The solution to this was of course not obvious, and several tries for making the model more agile and flexible in the face of obstacles were somewhat successful. It was however concluded that some change to the initial solution was needed and that in its current form, no consideration was given to instances of the simulation where obstacles would be implemented. An example of the performance when using the first iteration of the initial guess can be seen in Figure 3.2 below. This performance is relatively poor compared to the performance using a later iteration initial guess. This will be illustrated later in in this chapter.

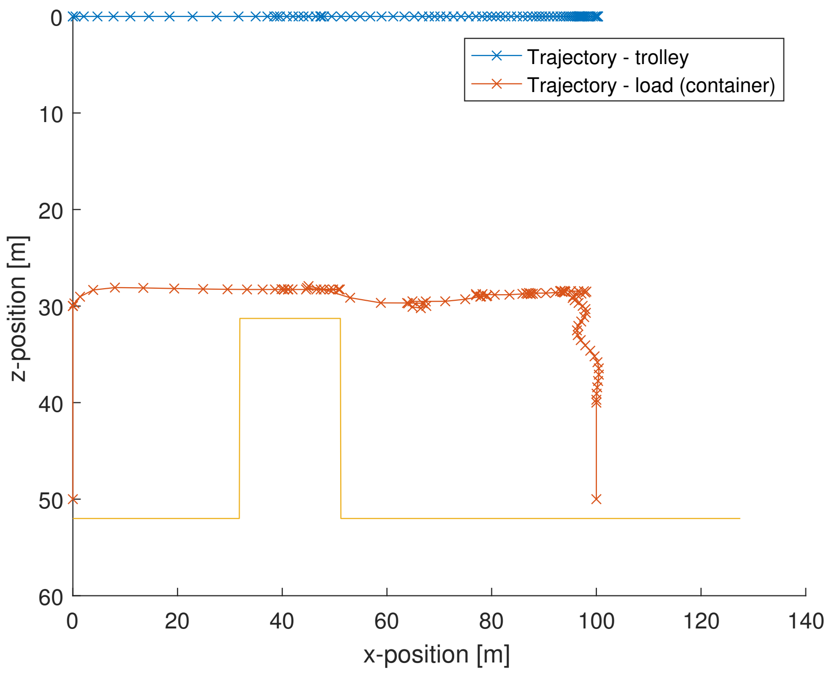

Figure 3.2: *Solver converged to a point of infeasibility using the first iteration of the initial guess. Note that the obstacle presented in this problem is very easy considering the initial and terminal coordinates of the load.*

One of the problems that occurred when the simulation faced an obstacle that was too aggressive or difficult, was that the simulation would not be able to produce a valid solution, instead resulting in a solution that was infeasible. This was due to a few reasons, among them that the initial solution did not take the obstacle into consideration when giving the solver a valid starting point. Another such problem was the previously mentioned issue regarding the initial solutions only objective, minimising the time from start to finish for the simulation. These were the root causes identified at this stage and these issues resulted in both infeasible solutions, but also non-optimal solutions that took thousands of iterations to identify. This was the biggest lead that the initial solution was so far from the optimal solution that the solver ultimately never reached it.

These problems lead us to the second iteration of the initial solution and the reformulating of the priorities for it. This will be presented in the next section.

3.2 Re-weighing the LQR-controller and normalisation

After identifying the problems caused by the initial guess of the model, one change was implemented to try to remedy how slow the simulation was to find solutions to the problems presented via the objective function and constraints of the model. The objective of changing the initial guess of the model was to lower the number of iterations needed for the simulation to converge to a feasible solution, and therefore lower the simulation time so that the workload of the thesis work could proceed faster. This was in the end successful even though this specific change was not a contributing factor in that change.

In order to make incentivise the initial guess to be more like a feasible solution to the problem, the most obvious change that was needed was to make the load (container) leave the ground and potentially match the curvature of the feasible solution. Even though there were several issues to tackle in the initial guess regarding the performance of the simulation. The one that had the most potential once solved, was deemed to be the height problem. This caused a change in the initial solution that in this specific remark was successful. However, with this change came other unwanted behaviour that will be visualised later in this part.

The first iteration of the initial guess contained an LQR-controller in order to weigh the states and control variables (inputs) of the model. These weightings were quite optimized for the simulation environment that did not contain any obstacles. This can not be said for obstacle filled environments however, which is quite obvious considering Figure 2.2.

To reach an initial guess that would contain some actual airtime for the load, there needed to be a change in the weighting of the parameters, and there were some other improvements done as well.

First, the original weightings for the model states were

$$Q_i = \begin{bmatrix} w_{x_{1_i}} & & & & & \\ & w_{x_{2_i}} & & & & \\ & & w_{x_{3_i}} & & & \\ & & & w_{x_{4_i}} & & \\ & & & & w_{x_{5_i}} & \\ & & & & & w_{x_{6_i}} \end{bmatrix} = \begin{bmatrix} 10000 & & & & & \\ & 1 & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & & 1 & \\ & & & & & 1 \end{bmatrix}$$

whilst the weightings for the model control signals (inputs) were

$$R_i = \begin{bmatrix} w_{u_{1_i}} & \\ & w_{u_{2_i}} \end{bmatrix} = \begin{bmatrix} 0.05 & \\ & 1 \end{bmatrix}$$

where $w_{x_{1_i}}...w_{x_{6_i}}$ are the initial weightings for the model states and $w_{u_{1_i}}, w_{u_{2_i}}$ are the initial weightings of the model inputs. These weightings are not so important as their relation to one another. When compared, it becomes quite clear that the most punished state is the lateral position of the trolley (x_1). This incentivises the initial solution to restrict the value of the lateral position of the trolley, which combined with the constraint of the terminal position makes the trolley move to the terminal position as soon as possible to minimise the cost (weighted cost) of the state in the initial guess. This is also affected once the weighting of the inputs is discussed. Due to the liberal weighting of the trolley acceleration (u_1), the initial guess is incentivised to use the model's maximal acceleration for as long as possible and therefore land in the terminal position as quickly as possible.

From these weightings it's obvious that this first iteration of the initial guess is pushed to rush to the terminal position without consideration of the benefits of gaining some elevation. These benefits can be observed in Figure 2.3 and show that using the dynamics of the model to your advantage, in this case swinging the cargo forward, can improve the performance of the simulation in light of the model and object function.

To remedy this problem, some adjustments were made to the implementation of the LQR-controller in the initial guess. The first of which was to use Jacobian matrices linearised in the initial position for the simulation, instead of linearising the differential equations of the model individually. Beyond that, the states and the inputs (control signals) are normalised using their maximum value before entered the LQR controller used in the initial guess. This change means that the weightings have the same default weighting, that is the $w_{x_1} = 1$ has the same significance to the LQR controller as w_{x_6}. In order to make the weightings proportional, they are simply normalised this way.

When the linearisation in the initial position and the normalisation is completed, the parameters are weighted in Q and R matrices just as before, but with new weighting values. The updated Q and R matrices in the second iteration of the initial guess can be found below. These are first presented as the matrices containing the normalisation

$$Q_{normalisation} = \begin{bmatrix} w_{x_{1_n}} & & & & & \\ & w_{x_{2_n}} & & & & \\ & & w_{x_{3_n}} & & & \\ & & & w_{x_{4_n}} & & \\ & & & & w_{x_{5_n}} & \\ & & & & & w_{x_{6_n}} \end{bmatrix} =$$

$$\begin{bmatrix} \frac{1}{120^2} & & & & & \\ & \frac{1}{4^2} & & & & \\ & & \frac{1}{40^2} & & & \\ & & & \frac{1}{3^2} & & \\ & & & & \frac{1}{0.1745^2} & \\ & & & & & \frac{1}{0.05^2} \end{bmatrix}$$

$$R_{normalisation} = \begin{bmatrix} w_{u_{1_n}} & \\ & w_{u_{2_n}} \end{bmatrix} = \begin{bmatrix} \frac{1}{0.8^2} & \\ & \frac{1}{0.5^2} \end{bmatrix}$$

and multiplying the normalisation matrices with the new weightings seen below

$$Q_{weighting} = \begin{bmatrix} w_{x_{1_w}} &&&&& \\ & w_{x_{2_w}} &&&& \\ && w_{x_{3_w}} &&& \\ &&& w_{x_{4_w}} && \\ &&&& w_{x_{5_w}} & \\ &&&&& w_{x_{6_w}} \end{bmatrix} = \begin{bmatrix} 10000 &&&&& \\ & 1 &&&& \\ && 100 &&& \\ &&& 1 && \\ &&&& 1 & \\ &&&&& 1 \end{bmatrix}$$

$$R_{weighting} = \begin{bmatrix} w_{u_1} & \\ & w_{u_2} \end{bmatrix} = \begin{bmatrix} 0.01 & \\ & 0.1 \end{bmatrix}$$

generates Q_2 and R_2, the matrices for the LQR-controller in the second iteration of the initial guess. These are presented below.

$$Q_2 = Q_{linearisation} Q_{weighting} =$$

$$\begin{bmatrix} \frac{1}{120^2} &&&&& \\ & \frac{1}{4^2} &&&& \\ && \frac{1}{40^2} &&& \\ &&& \frac{1}{3^2} && \\ &&&& \frac{1}{0.1745^2} & \\ &&&&& \frac{1}{0.05^2} \end{bmatrix} \begin{bmatrix} 10000 &&&&& \\ & 1 &&&& \\ && 100 &&& \\ &&& 1 && \\ &&&& 1 & \\ &&&&& 1 \end{bmatrix} =$$

$$\begin{bmatrix} 69.444 &&&&& \\ & 0.0625 &&&& \\ && 0.0625 &&& \\ &&& 0.111 && \\ &&&& 32.84 & \\ &&&&& 400 \end{bmatrix}$$

$$R_2 = R_{linearisation} R_{weighting} = \begin{bmatrix} 0.01 & \\ & 0.1 \end{bmatrix} \begin{bmatrix} \frac{1}{0.8^2} & \\ & \frac{1}{0.5^2} \end{bmatrix} =$$

$$\begin{bmatrix} 0.0156 & \\ & 0.4 \end{bmatrix}$$

Since only the relation between the weightings in the matrices are important, for readability, the Q and R matrices are normalised one final time below.

$$Q_{2_n} = \frac{1}{69.444} Q2_n = \begin{bmatrix} 1 &&&&& \\ & 0.0009 &&&& \\ && 0.0009 &&& \\ &&& 0.0016 && \\ &&&& 0.4729 & \\ &&&&& 5.76 \end{bmatrix}$$

$$R_{2_n} = \frac{1}{0.0156} R_2 = \begin{bmatrix} 1 & \\ & 25.64 \end{bmatrix}$$

Keep in mind that these values might seem arbitrary, but they are produced by normalisation as explained above combined with purposeful adjustments to the weightings of the Q and R matrices in order to achieve desired results as explained in part 3.1.

As can be seen in Q_{2_n}, the weightings are greatly reducing the punishing factor for x_2, x_3 and x_4 compared to Q_i as well as increasing the punishing factor of for x_6 (see Table 2.1 for state descriptions). By only looking at this matrix, it might seem that we don't want to punish high values in x_3, however this is not the case. Since the vertical axis of the coordinate system (can be observed in Figure 2.1), the desired result is that x_3 has small values during the trajectory of the load, that is, has small values. Therefore in $Q_{weighting}$, you can see that we are punishing the length of the rope in order to achieve the desired height of the load and consequentially shortness of the rope. This implementation was somewhat successful, even though the implementation of the initial guess added some unstable behaviour that was very difficult to counter or adjust with the weighting of the matrices for the LQR-controller. The initial guess containing these changes can be observed below in Figure 3.3.

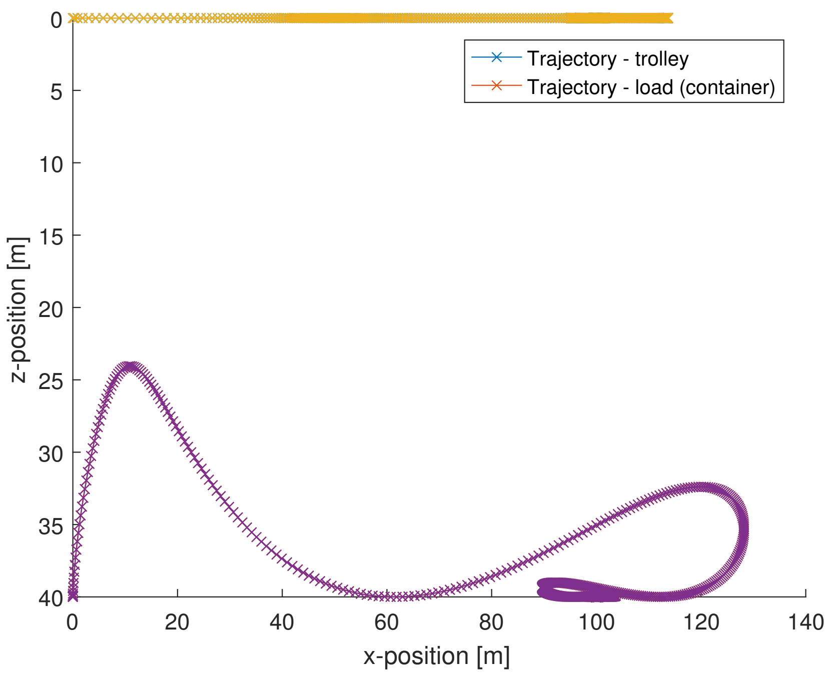

Figure 3.3: *Second iteration of the initial guess containing the Q and R matrix weighting values presented above.*

Due to the behaviour of this solution to the initial guess problem, some creative work was needed in order to provide a robust initial guess that the user of the simulation could rely on. This was easier said than done, but some interesting solutions were discussed. Two of them are presented below.

The first and maybe most straight forward idea for solving this issue was to use dynamic programming along with a lookup table which would contain a grid representing some discrete positions possible to attain by the load. In order to generate an in initial guess this way, one had to take the obstacles that are the focus of this thesis, into account. Then choosing the discrete positions that hugs the obstacles with as little elevation as possible for the load, and then use that trajectory as an initial guess. This method has some strengths and some weaknesses. Theoretically, with enough resolution in the look-up table, the solver in the simulation should be able to reach the optimal solution of the problem presented along with the model. When the initial guess not only avoids the the obstacle by accounting for it (which the first and second iteration of the initial guess doesn't), the trajectory will be optimal in context of the possible coordinates in the lookup table. This lookup table will of course be the same for all scenarios and environments for the simulation, which could mean that this method lacks adaptability. Because dynamic programming is built on the premise of minimising travel cost, where the steps between positions has a cost attached, there might be some unexpected decisions as well. For example, a lower value will be attributed to lower elevation and distance between the two positions that are under evaluation. This could cause the initial guess to hug the profile of the obstacles too much, and present a problem in combination with the model, that the solver in the simulation can't solve to a reasonable degree. This method has shows promise could be investigated further.

Another method that was interesting and had through previous studies achieved some interesting results, is called Homotopy. A short description of this method is given in part 1.4 when referring to Axehill and Bergman (2018).[3] This is not as much a method for constructing an initial guess, as it is a method to simplify the problem for the simulation and the solver and incrementally raise the difficulty. This is done by initially removing the obstacles in its entirety, and after that introducing them little by little and letting the solver generate a solution to be used as an initial guess for the next iteration and so forth until the final problem is presented, and an initial guess that is similar to a solution is guaranteed. This is a very interesting concept and even after the completion of this thesis, it might be an interesting area of further research.

The two methods presented above were discussed and researched lightly. Before any trials for implementation were initiated, a drastic change to the optimal control interface software YOP was released. This both contributed and caused more work to this thesis but for the initial guess of the simulation, this was a great improvement.

3.3 YOP changes and the current version

Roughly half time into the project, there was an update to the optimal control interface that handles the translation between the MATLAB code and the open source software CasADi, called YOP. Several updates were performed, which for instance included how the constraints in the model could be formulated. This update is implemented in the description of the constraints in Chapter 1. Before this update, the constraints were limited to limiters, whilst the new format allows for box constraints. Due to the previous way the constraints were formulated, the decision for including a convex hull was made (see Chapter 5). This was due to the inability to implement padding to the coordinate of the load on all sides.

This new capability for input of an initial guess also opened some interesting possibilities, such as introducing even more nuanced weighting between different states and control inputs. To begin with, the easiest and most straight forward initial guess imaginable is simply to get a trajectory that takes the load straight up vertically, then all the way to the correct position laterally and then straight down vertically. This was the first and most obvious way to improve the initial guess from the previous one, and so this was implemented, quite successfully as can be observed in Figure 3.4 below.

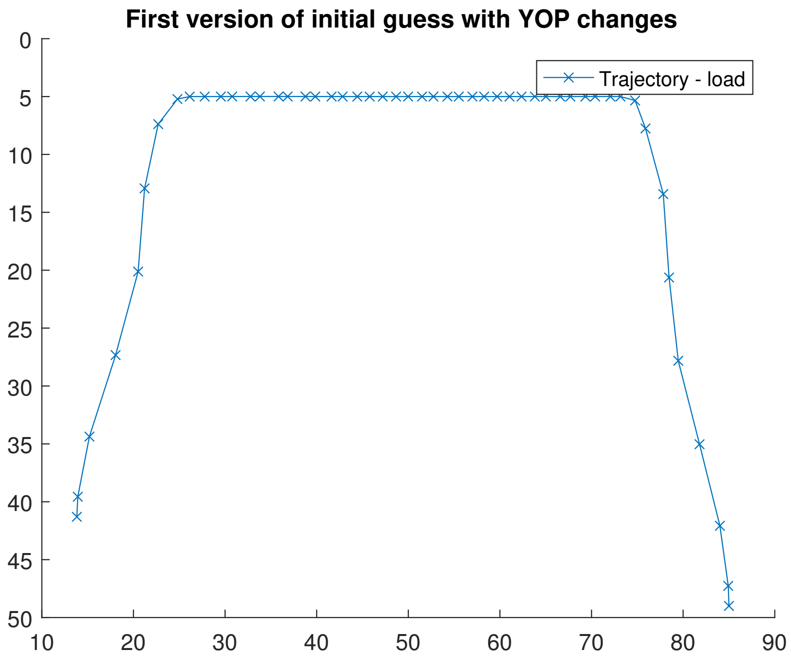

Figure 3.4: *First iteration of the initial guess using the changes in YOP to the solvers advantage. Note that the start position is elevated, which will be presented in Chapter 4.*

This initial guess worked well and was used for the better part of the development of the obstacle profile, convex hull and the safety conditions. However, during the development of these soon to be presented features another problem with initial guesses was discovered. When using different initial guesses for the same model and obstacle profile, the solver would converge to different acceptable or optimal solutions. These non unique solutions created another level of complexity to how the initial guess should and could be used. Because the areas of convergence that the solver works within, the initial guess is extremely important to finding an optimal solution. The previous discussion regarding dynamic programming and Homotopy suddenly became highly relevant, as to how to generate an initial guess that can guarantee an optimal solution by the solver. It is easy to observe that the initial guess in Figure 3.4 above is not hugging the obstacle profile and might not generate an optimal solution.

So, instead of restructuring the initial guess in the simulation again, this version of the initial guess follows a common principle regarding initial guesses in control theory. That is, it is important to know the environment in the simulation and recognise the behaviour of the model, so that the initial guess can be placed in an area of convergence that is likely to benefit the solver in its task. This of course also depends on the objective function and what kind of optimisation the user of the simulation wants to study. This is not only important in order to for the solver to converge on a solution that is optimal, but also shorten the number of iterations in the simulation and thereby the time it takes for the solver to arrive at the optimal solution.

Note that this problem mostly occurs when the obstacle profile and therefore the convex hull presents a real challenge to the solver.

Below are two examples of a well suited and not so well-suited initial guess for a specific obstacle profile.

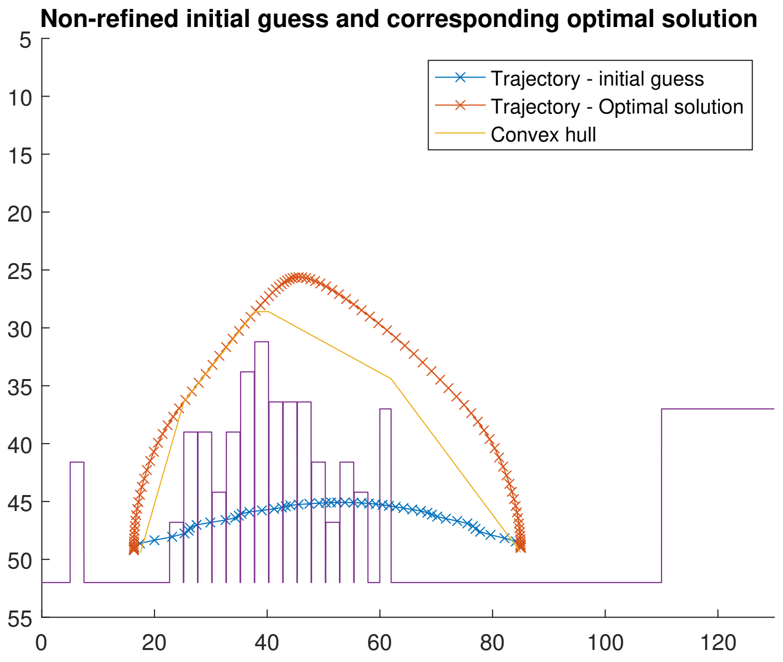

Figure 3.5: *A version of the third iteration of the initial guess that is not suited to the obstacle profile, and may therefore not converge to the optimal solution. This obstacle profile however is too simple to force that behaviour. Note that this obstacle profile uses a convex hull as well.*

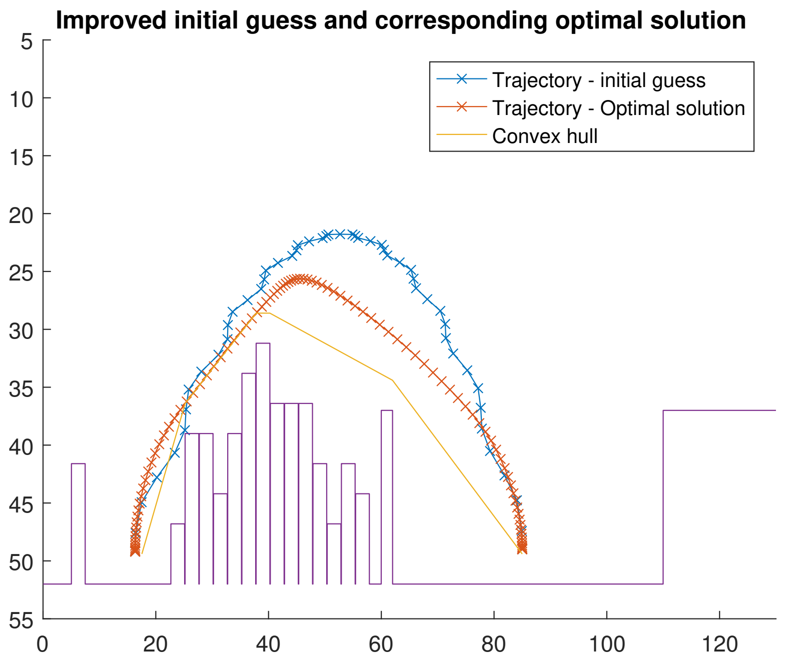

Figure 3.6: *A version of the third iteration of the initial guess that is suited to the obstacle profile, and may therefore give the solver a better chance of converging to the optimal solution. Note that this obstacle profile uses a convex hull as well.*

4 Results

In this chapter, the results of this thesis work will be presented. This will be done in chronological order as the work was done. Some findings will also be discussed, with the majority of the discussion left to Chapter 5.

4.1 Obstacle profile

The core element of this thesis work is the obstacle profile that the containers on a cargo ship construct in 2D space in this simulation environment. This along with some of the physical restrictions of the STS-crane make out the environment for the model and constraints that the solver must account for. Below is an early implementation of the obstacle profile without specific measurements cranes in mind.

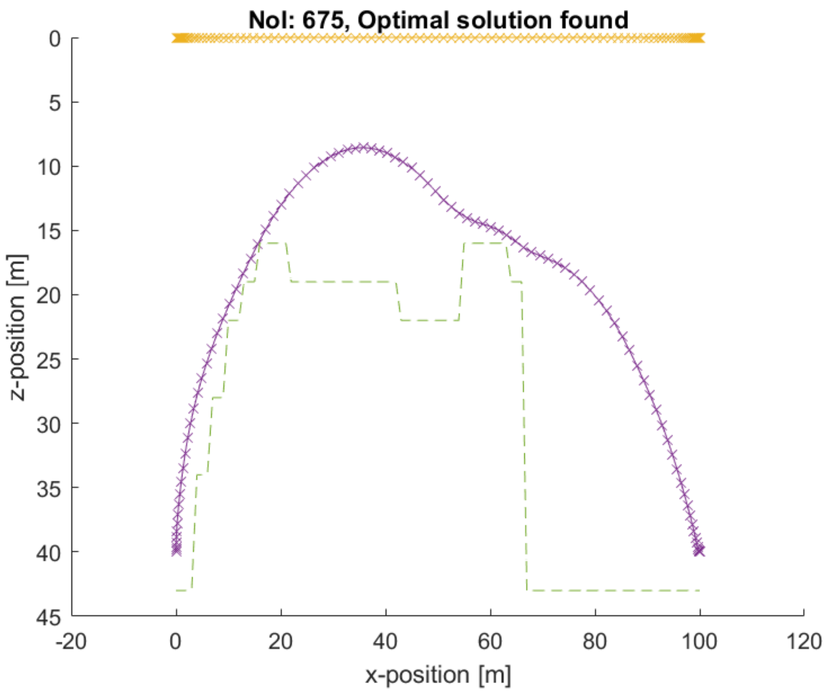

Figure 4.1: *Early optimal solution to a problem containing an obstacle profile.*

Starting the implementation of the obstacle profile, the major task was to investigate the behaviour of the solver and model in context of the obstacles. This was important since the simulation had thus far not been designed with obstacles in mind. So, to begin the implementation, the analysis of the performance became quite clear. In the case of an obstacle that is tall and thin, the solver would try its best to place the discrete points that construct the loads trajectory on separate sides of the obstacle, whilst the trajectory of the load (container) would pass straight through the obstacle. This was quite the problem, and can still be observed in the final version of the simulation, but more about that in Chapter 5. For an example of this behaviour, see Figure 4.2 below.

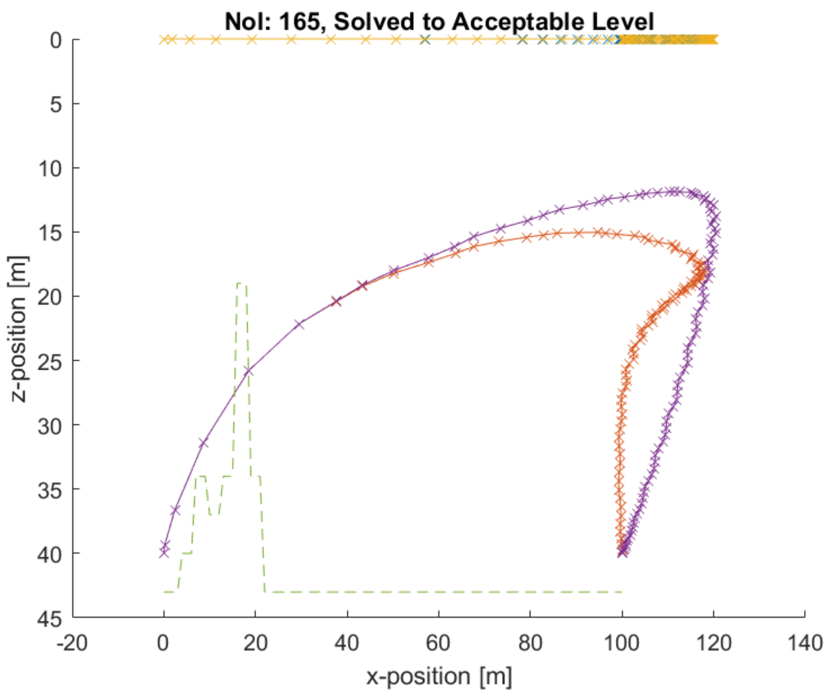

Figure 4.2: *Early optimal solution to a problem containing an obstacle profile.*

In Figure 4.2 above, it is quite obvious that there is a change needed in order to prevent this behaviour. Through testing, this behaviour could be remedied through limiting the rotational velocity θ. This was done because it could be observed that this behaviour was mostly possible due to the swinging capabilities of the load by acceleration the trolley in such a way that would maximise the lateral velocity of the load whilst trying to clip through the obstacles. Another possibility that was investigated was to add padding around the load, such that a larger area would need to be clipped through the obstacle, making it more difficult for the solver to do so. However, this solution lacked the reliability that was desired and since a robust solution to this problem could be very meaningful to the future performance of the simulation, another route was taken. This solution consists of the combination of two separate convex hulls and will be presented in Chapter 4.2.

Because this simulation should be able to function for all the possible cranes and ports of the users liking, it was important to make sure of complete functionality for the largest STS-cranes and cargo ships available to date. These turned out to be the Super Post Panamax/Megamax Gantry Crane and the New Panamax or Neopanamax container ships. While the crane size is quite typical and can be summarized in a neat table of measurements, the container ships are all quite different and can not be defined by some specification. Instead i opted to base the measurements of the boat in the simulation after the dimensions of the crane.

The cranes dimensions are described by Table 4.1

Component	Measurement
Outreach	46 - 70+ m
Lift height	30 - 49 m
SWL	65t twin | 80 t tandem
Hoisting speed	70/175 m/min
Trolley speed	210 - 240 m/min
Travel speed	45 m/min
Wheel load**	60 - 80 t per metre

Table 4.1: *Approximate measurements and capabilities of the Megamax Crane. ** Based on 8 wheels per corner at 1 m spacing.*

The front legs of the crane (the left most) have a beam running between them. This beam presents another obstacle that needs to be considered. In some cranes, there are also a platform on the back of the crane, to hold containers temporarily. This feature was desired by ABB, and so it is an option for a terminal position in the simulation. It was also a request from ABB to make the simulation easy to handle, so that an engineer in the field could use it quickly and smoothly. This was easily made by making an interface where the user inputs the number of containers per column on the cargo ship in the form of a vector. This vector is then translated to both an obstacle profile that is plotted, but also the convex hull that includes the physical features of the crane explained previously. An example of that input vector is

$$[5, 2, 4, 5, 1, 1, 3, 4, 4, 2, 5, 5, 3, 5, 7, 8, 6, 6, 6, 4, 2, 4, 3]$$

which contains 23 columns since that is what the dimensions of the gantry crane could tolerate. This vector was used to provide the obstacle profile and the convex hull for Figure 4.3 below.

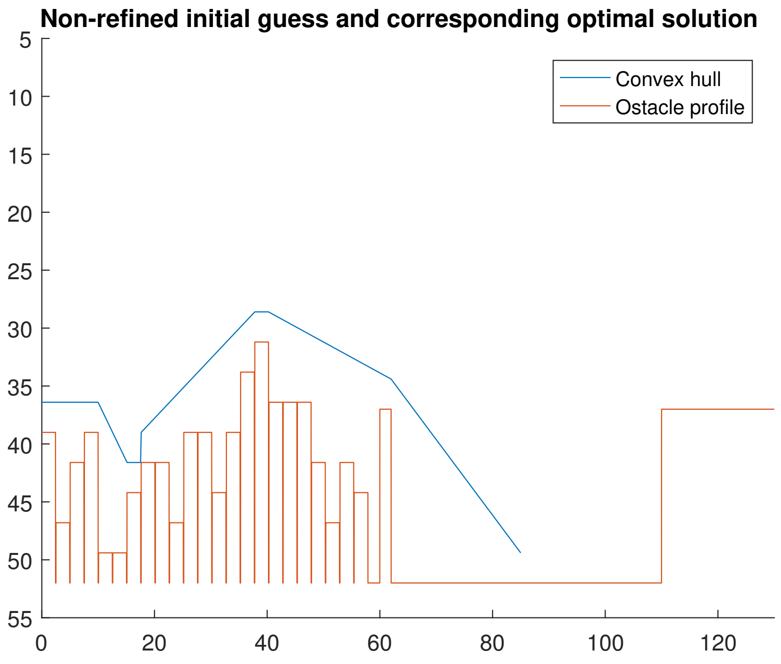

Figure 4.3: *An example of the final state of the obstacle profile as well as the physical limitations of the crane.*

4.2 Convex hull

Because of the unwanted behaviour of the solver when it comes to tall, thin obstacles another feature was added to the simulation (see figure 4.2 for reference). This originated as an idea to add padding to all sides of the point mass signifying the load (container) but implementing such a solution turned out to be not compatible with the earlier version of YOP. Instead, another technique was used, a convex hull. A convex hull in 2D space, as in this simulation uses a point mass (the points providing the obstacle profile) and drawing straight lines between the points positioned in the outer rim of the point mass. This creates a geometric form that contains all the points in the point mass. However, the only useful part of a convex hull for this simulation is the part that is positioned between the trajectory of the load and the obstacles. See figure 3.6 for reference. Also, the only part of the simulation environment that was relevant for such a hull was the lateral positions between the loads initial and terminal positions, therefore the hull could be implemented such that the hulls initial position was equal to the loads.

Beyond the prevention of the solver using the dynamics of the model to its advantage, the resolution of the discretisation for the trajectory of the load is vital to preventing the behaviour presented in figure 4.2. A low resolution may result in more instances of the trajectory clipping through the obstacles, while using a higher resolution will result in this behaviour being less common. However, this fact is coupled with a higher resolution resulting in a longer optimisation and vice versa. So, in order to avoid clipping of the load's trajectory, one should try to use the convex hull whist increasing the number of discretised points in the trajectory of the load, but not too much. In figure 4.4 the amount of discretised points is 300, which both ensures a trajectory that shows less clipping behaviour, but the solution took a long time to converge on. In contrast, figure 4.5 contains 50 discretised points and therefore runs the risk of making the load clip through the obstacles. The sweet spot for this calibration depends on the obstacle profile, but between 150 and 200 discretisation points seems to work fine in most instances.

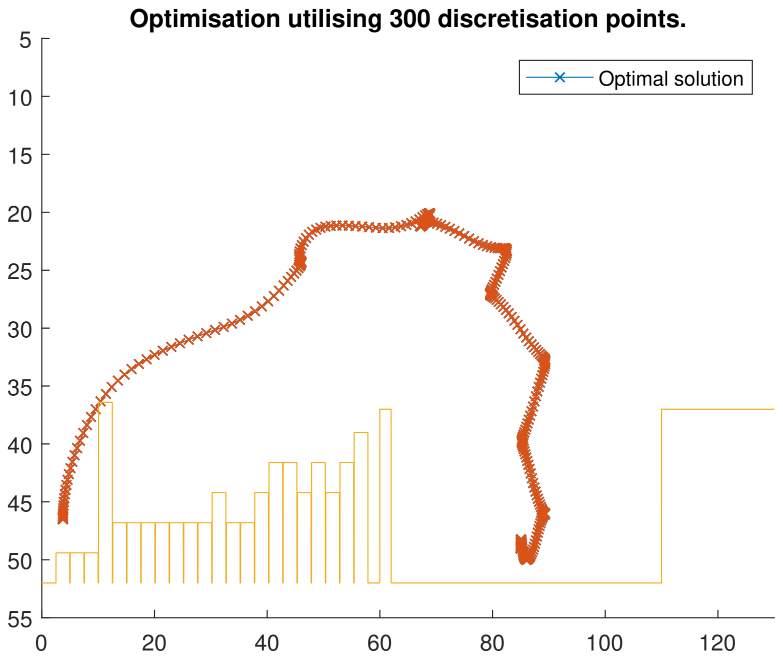

Figure 4.4: *A convex hull implemented in the simulation whist running optimisation of a trajectory containing 300 discretised points.*

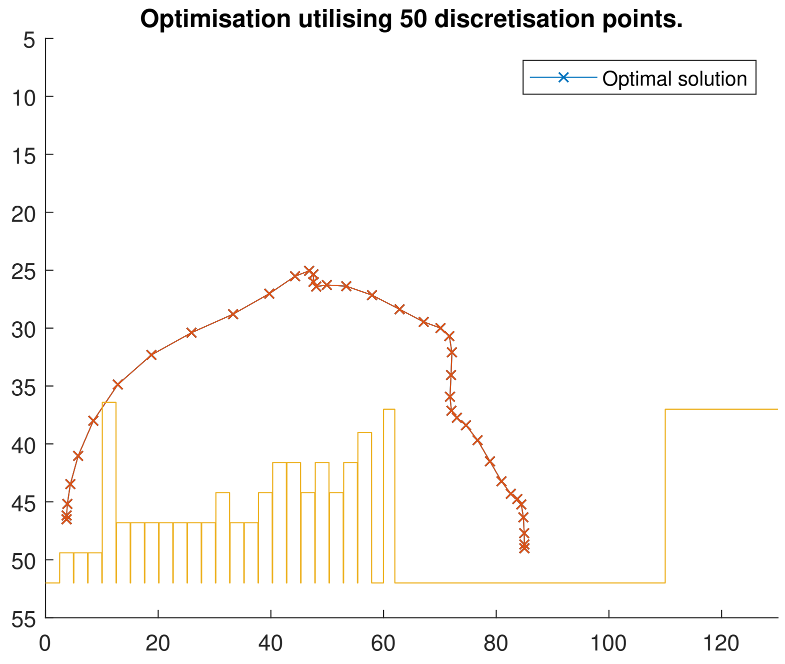

Figure 4.5: *A convex hull implemented in the simulation whist running optimisation of a trajectory containing 50 discretised points.*

The first instance of the convex hull used in the simulation was a hull that hugs the outer points of the point mass that is the obstacle profile. An example of this convex hull can be observed in figure 4.6 below.

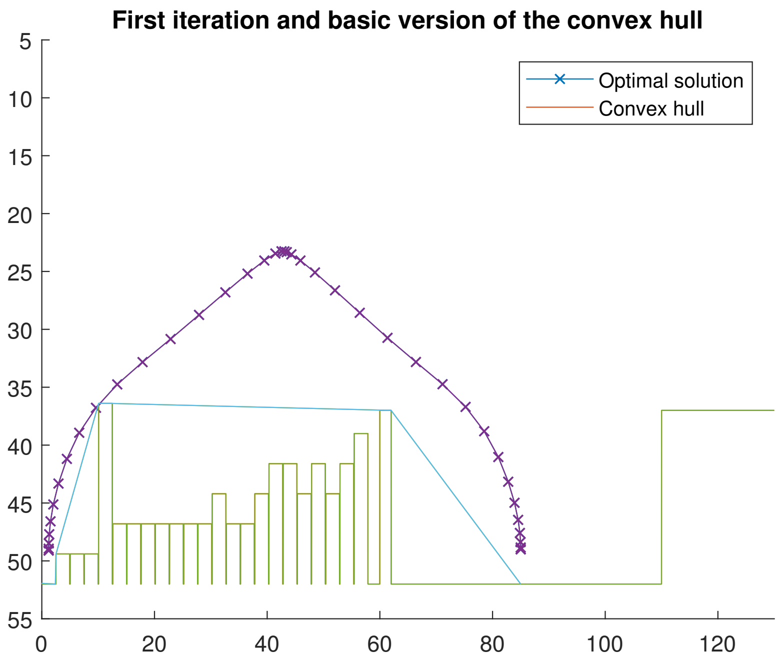

Figure 4.6: *The first iteration of the convex hull constructed from the points most extreme in the point mass that describes the obstacles.*

This hull was successful in preventing the clipping behaviour presented in figure 4.2, however it lacked some of the features that a completely secure convex hull (guaranteeing a safe, collision free trajectory) should contain. Another behaviour that was observed was the swinging of the load to the left according to the plots presented throughout the report, in order to gain speed and to then swing to the right towards the terminal position of the load. In some rare occasions, this behaviour became a problem because the clipping started to happen to the left of the initial position of the load. Adding another hull between the edge of the environment of the simulation and the initial position of the load became important and was done in order to prevent this flaw in the behaviour of the solver. This hull is included in the coming figures containing convex hulls in this chapter.

In order to guarantee the safe trajectory of the load that is the major point of the problem formulation of this thesis work, some additional constraints were added to the convex hull. These include both lateral and vertical padding. That is, the hull was compared to the obstacle profile positioned one container height above the obstacle profile (2.6 meters). This feature is presented in figure 4.7 below.

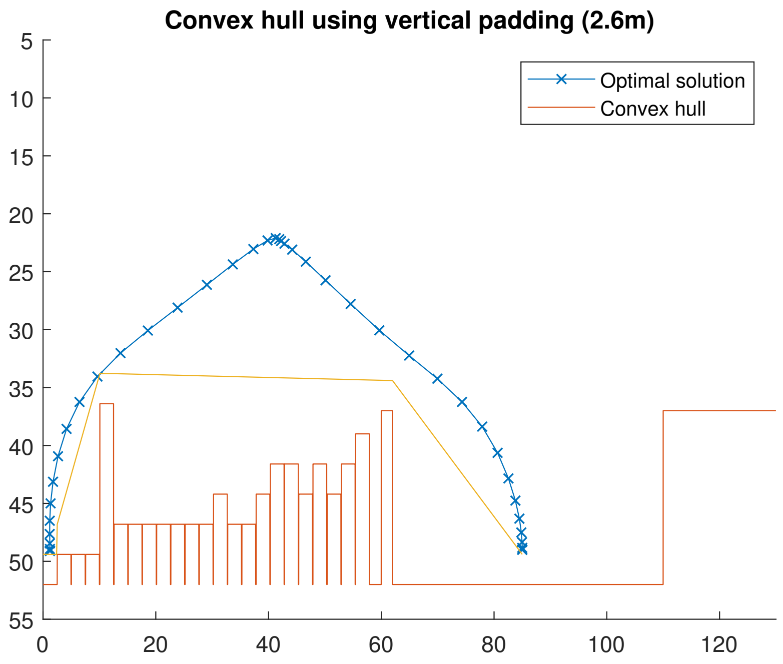

Figure 4.7: *Convex hull using the vertical padding of 2.6 meters. This is the vertical measurement of a standard container.*

Due to how the obstacle profile is constructed, the possible combinations of the obstacles and the behaviour of the solver, the sole place in the simulation environment that needed lateral padding was the starting position. This was extra apparent when the starting position of the load was below and beneath the neighbouring container columns. Without this lateral padding, swaying in of the load was a common behaviour of the solver, which of course was not wanted in between two tall rows of containers. This lateral constraint can be observed in figure 4.8 below.

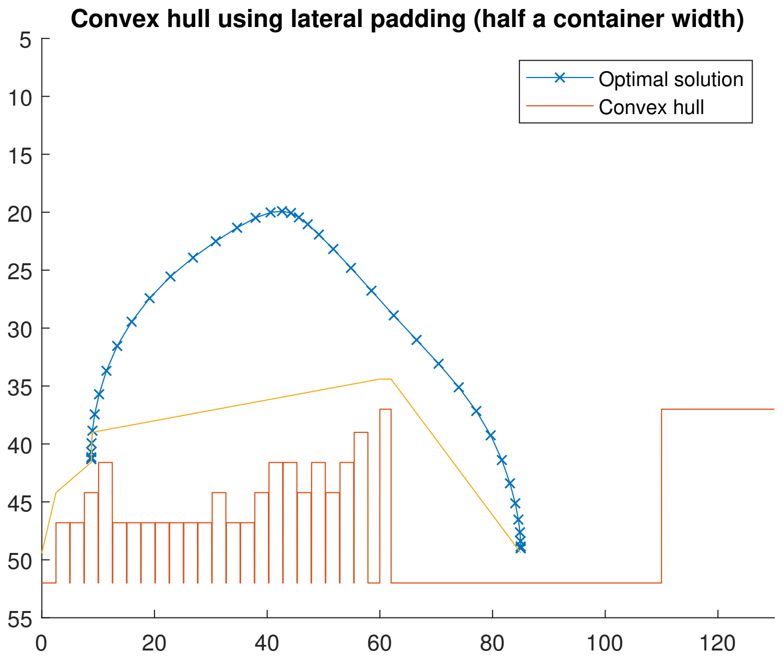

Figure 4.8: *Convex hull used containing the lateral padding. This padding leaves 0.09 meters of lateral space for the solver to work in if the load is positioned between two higher stacks of containers, as in figure 4.9 and 4.10.*

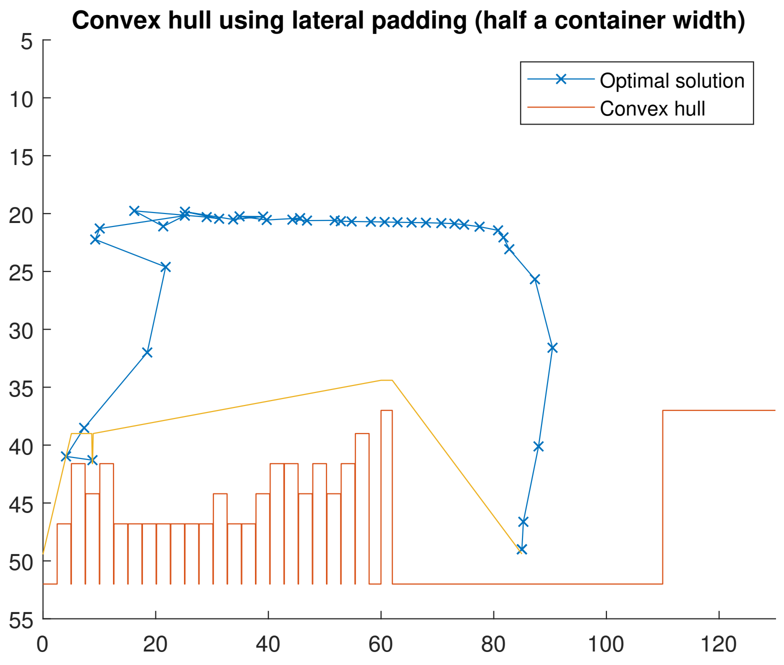

Figure 4.9: *Note the Initial coordinate of the load, Wedged between two higher stacks of container leaves 0.09 metres for optimisation and operation.*

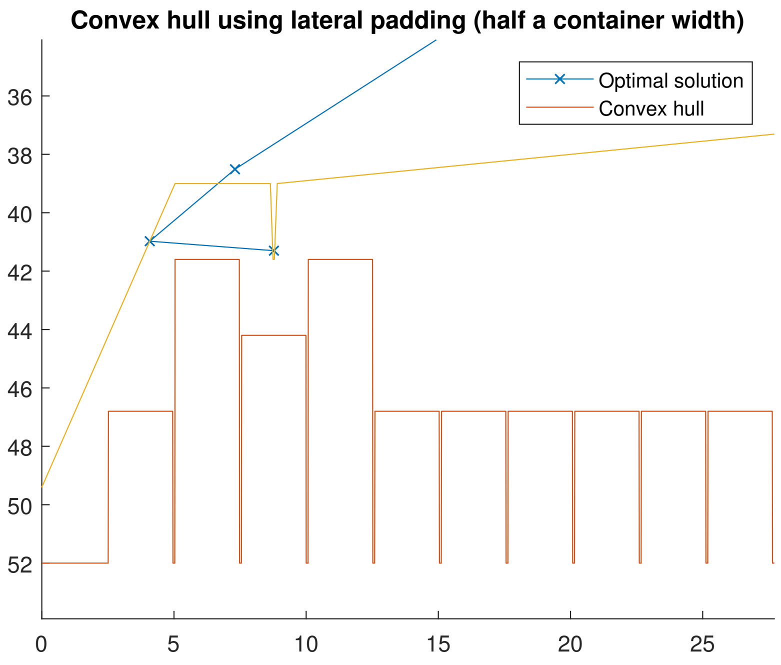

Figure 4.10: *An enlarged figure of the initail coordinate for the load in figure 4.9 above.*

This complete convex hull using both left and right hull, vertical and lateral padding is highly successful in its designed task, guaranteeing a safe trajectory. This however does not guarantee neither good solver performance nor actual solving of the problems presented to the solver. This might seem strange, but due to how the convex hull forces the solver to operate in a more confined environment, the degrees of freedom of the model is lowered. This in turn results in the solver not being able to find an optimal solution or not solving a problem at all. In simple terms, this latest version of the convex hull may push the solver into a space where optimisation is no longer possible. Since the whole purpose of this simulation is optimisation, this feature is at best questionable and at worst useless. This does not mean that the convex hull is useless however, since the less safe version of the convex hull is still usable in problems where the lateral and vertical padding would destroy the solvers chance at optimising. Due to this fact, the convex hull without lateral nor vertical padding is the most useful for generating and understanding optimal solutions to a problem presented, even though it might be difficult for the solver. Conversely, the padding is most useful to solve easier problems when safe trajectories are in focus.

4.3 Safety conditions

The second implementation needed for guaranteed secure trajectories are security conditions. These conditions consider the possibility of a stop in the movement of the trolley due to some sort of error or emergency reaction. This causes the load to swing to either way and achieve a position that potentially is dangerous to the surrounding containers on the cargoship as well as the load carried by the crane. A swinging load will combine its current total velocity and potential energy and translate that into a worst possible position. This position is what this safety condition considers and makes sure that this position can be reached but safely. This is done by evaluating the position and velocity of both load and trolley in all the discretised positions of the trajectory and creating a trajectory with this condition in mind. This results in a trajectory that differs from an optimal trajectory not using the safety conditions but will instead guarantee safety for the load and surrounding containers.

The calculation for the potential position of the load is done using the kinetic and potential energy of the current position of the load and translating that into a position where there is no kinetic energy left, only potential. This position is reached by using the formula for translating kinetic and potential energy between two positions, as seen below, where the second position contains no kinetic energy.

$$m_{load} g h_{initial} - m_{load} v_{perp,load}^2 = m_{load} g h_{terminal} \tag{4.1}$$

Where m_{load} is the current load of the container carried by the crane, $h_{initial}$ is the initial height of the load when the emergency stop happens. $v_{perp,load}$ is the velocity of the load perpendicular to the angle of the rope at the time of the emergency brake. $h_{terminal}$ is the terminal and highest possible height of the load after swaying due to the emergency stop.

It should be noted that the sign for the second term in the left side of the formula is switched due to the inverted z-axis of the simulation. A lower height in the coordinate system used by the simulation is corresponding to a larger height in the real world.

In order to visualise the positions in discussion, figure 4.11 below contains both the position before the stop of the trolley (x_1) and the position after the stop (x_2).

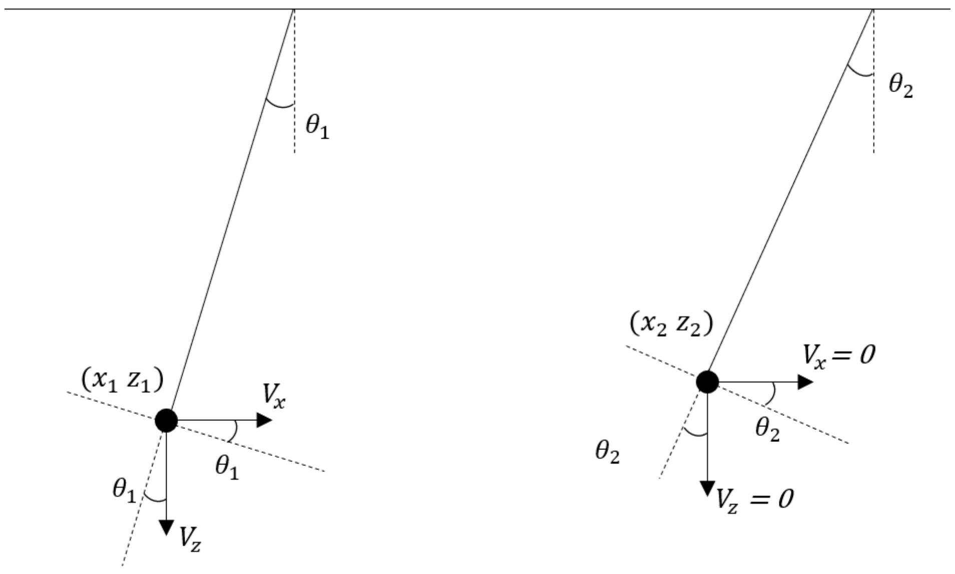

Figure 4.11: *a) Position of the load in relation to the trolley before emergency where the trolley is halted to stand still. b) Position of the load in relation to the trolley when the trolley has stopped as a result of the emergency stop. Note that the load has gained some height compared to in sub-figure a.*

After the first implementation of this feature into the simulation, a flaw in the implementation was quite apparent. When considering swinging the load forward in the environment when the trolley is stopped, there are two corresponding worst case positions. These are on the same height but on both sides of the loads resting position ($\theta = 0$). After this implementation, the trajectory of the load can be guaranteed to be secure due to both the twin safety conditions. The safety conditions are visualised in figure 4.12 below.

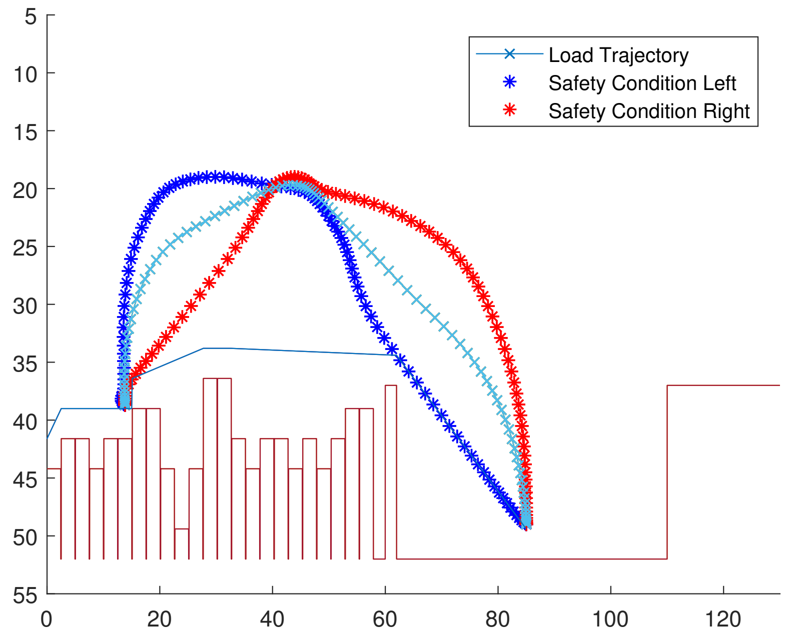

Figure 4.12: *First implementation of dual safety conditions. Observe that the safety conditions visualise the edge position reachable for the load in each instance of the discretisation of the trajectory.*

Also, due to the conservative nature of these conditions. This conservative nature is cased by not taking inertia of the rotating load around the static trolley into account. It should be said that this inertia could be implemented, but the focus of this thesis was not to increase the complexity of the model, but to reach usefulness using a simpler model and dynamics.

Since the conditions are conservative, it adds some difficulty for the solver due to how it lowers the degrees of freedom for the model and therefore shrinks the room for investigating optimal trajectories. This feature is however both beneficial and harmful to the performance and usefulness of the simulation. It is obvious that the safety conditions create a safety guaranteed environment and therefore can help to generate trajectories that the user of the simulation can trust. An example of this is shown in figure 4.13 below.

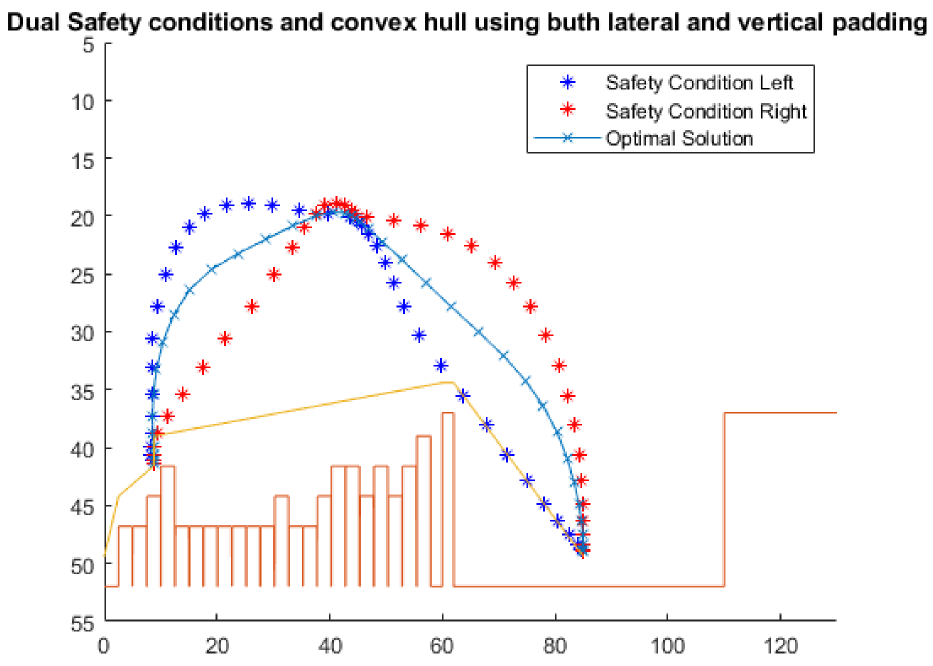

Figure 4.13: *An example of an entirely safe trajectory for the load. This is due to both the safety conditions, but also the most secure version (latest version) of the convex hull.*

However, the safety conditions lower the degrees of freedom contained by the model, which makes problems harder to solve for the solver, and in some cases impossible. An example of this can be seen in figure **??** and **??** where figure **??** contains an obstacle profile that is possible to solve without using safety conditions and figure **??** shows the same obstacle profile that is not solvable using the safety conditions.

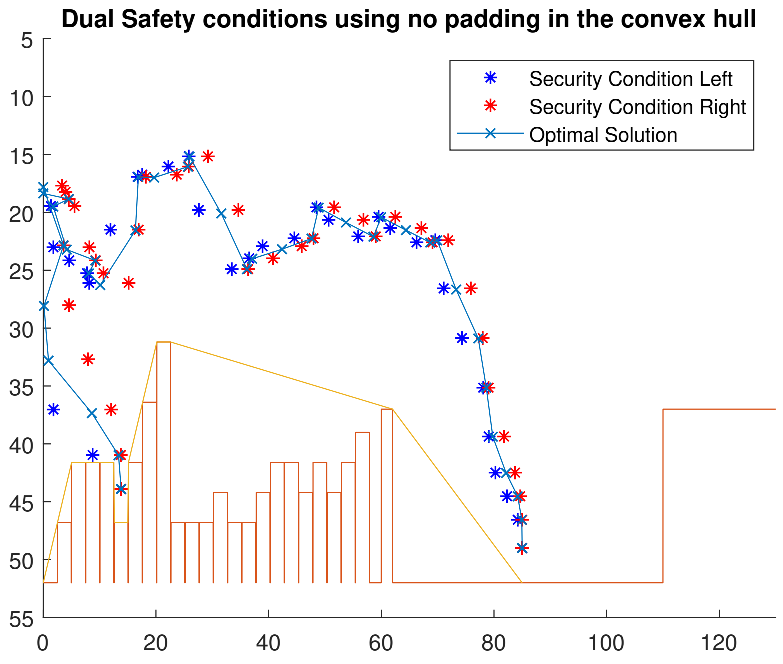

Figure 4.14: *Obstacle profile not cleared by the solver due to the safety conditions.*

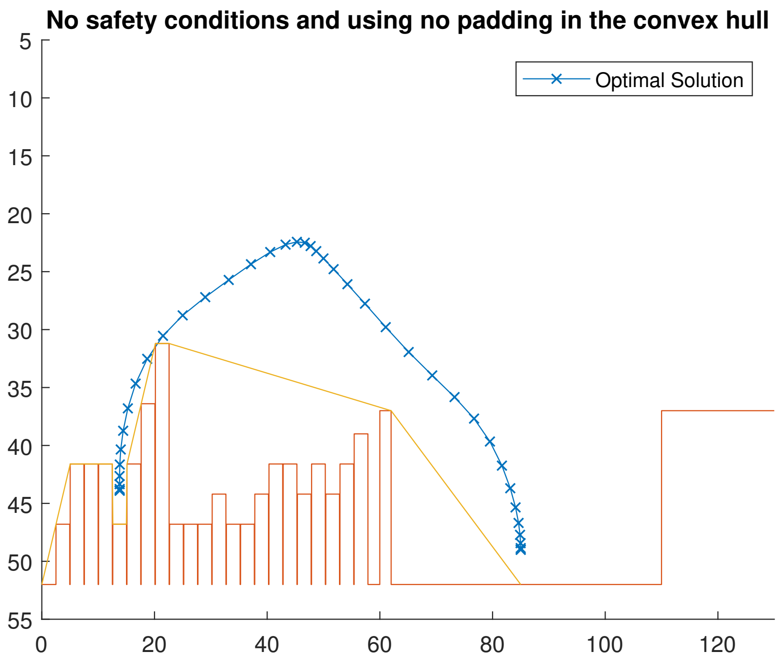

Figure 4.15: *Obstacle profile cleared by the solver due to ignoring the usage of safety conditions.*

In this way, the safety conditions are both crucial to the usefulness of the simulation, but also detrimental to the performance. This can be handled by users in different ways. For instance, if the obstacle profile is difficult, then maybe not using safety conditions is suitable, and can yield a trajectory that has the rough shape of what a secure trajectory would look like. This way, the user could learn roughly how the optimal trajectory looks in light of a certain obstacle profile. This insight could be useful even though it might not be as applicable to real life situations as a safe trajectory.

Along with the convex hull implemented into the model, the safety conditions present a real challenge to the solver in the simulation. These features are highly useful as they can be implemented to guarantee safe trajectories, but they can also present problems to the solver that are close to impossible using these implementations to their full extent.

5 Discussion and conclusion

5.1 Discussion

This section will reflect on solver behaviour and general behaviour of the simulation that has yet to be discussed in this report. Since these areas are important to understand when working with this simulation, this discussion as a part of this report is surely valid.

Regarding behaviour of the solver used in this simulation (IPOPT), there are some characteristics observed throughout working with this simulation. First, the most general behaviour observed is the tendency to strive to solutions that might not be feasible. This would most likely be correctly attributed to the solver being pushed into a few degrees of freedom for optimising. This can be observed in many iterations of the simulations both using convex hull and safety conditions or not. An example of this is a problem with an initial position squeezed between two high columns of containers, where the beginning of the trajectory needs to be almost completely vertical. This presents a challenge to the solver that it might interpret as difficult, and as a result might retreat to another area of convergence in order to find another solution which happens to be infeasible. If this behaviour is observed using the simulation, there are some suggestions for solving this issue. It could be fixed using a higher resolution for discretisation. This is a very simple and effective remedy for this type of problem, just as the original problem remedied by the convex hull. Another is to position the initial position in a vertical position that makes the optimisation easier for the solver. This will of course affect the time of trajectory but will be much kinder to the solver in regards of how easy the problem is to solve. If the users sole purpose of using the simulation in this instance is to learn the optimal trajectory of the load in context of the obstacle profile, then this solution can be recommended since the target time for the trajectory will not be a close representation of what to expect in real life situations.

Another trend in behaviour of the solver that has been observed is that the solver can really struggle to find the optimal solution, but with enough time it can find the correct area of convergence and find the optimal solution. This behaviour can show when presenting a very difficult problem to the solver and it is taking some 2000+ iterations to get close to optimal solution time. It would be useful to note that the trajectory time for the optimal solution should not exceed some 36 seconds. This would mean that the solution is indeed not optimal, and the solver has converged to some solution that is not optimal, but this behaviour has been minimised using initial guesses that closely resemble the optimal solution for the obstacle profile. This slow convergence to an optimal solution after 2000+ iterations could possibly make the user cancel the optimisation, when with a couple of hundred or thousand iterations more, the optimal solution can be found. This can be done if the user wants to reduce the time for the solver to find the optimal solution and change the configuration of the model or simulation configuration, but this information is vital for proper understanding of the simulation's behaviour. It is also important to note that the solver will mostly only extend the number of iterations to 2000+ if the solver has a potential to solve the problem. That is, a problem that is unsolvable or where the solver converges to an infeasible solution, will be solver (mostly) in under 1000 iterations.

A quirk of the simulation that has yet to be discussed is the existence of non-unique optimal solutions. This has been discovered in a couple of different facets of the simulation, both in the length of the rope and the velocity of the trolley. An explanation that is easier to grasp is that an optimal solution to a problem could contain some variety in rope length or trolley acceleration. For example, an optimal solution could not retract the rope very far and be just as fast as the solution retracting the rope a lot. This issue could appear in the future as well, which is why it is important to know how to remedy this problem. The objective function in its current state can contain some of the states or control signals (with appropriate weighting) so that the variety of for instance rope length can be eliminated, Remembering this problem can be useful in the future if this problem appears again, but concerning another state or control signal.

5.2 Conclusion

Firstly, the conclusion regarding the business need for this type of simulation is nuanced. For advanced customers and ports with semi-autonomous STS cranes, the value of such a simulation is quite clear. The simulation closes the gap between vertical-lateral-vertical transportation of cargo and optimises both in aspects of time and energy consumption. This is obviously desirable to the ports with resources to further invest in a control system and or simulation that is shown in this report. It is also clear that the value provided by this simulation for a customer of this type can be improved with further investigation and development. However, that also means that the value of this simulation for such a customer in its current state is not optimal.

Looking at less advanced ports and how they may benefit from a simulation or control system such as this one, it becomes clear that only one use-case is interesting. Using the optimised trajectories from this simulation as references for learning or optimising the manual control of the STS cranes could very well be of value. With that said, the usability of the simulation could need some work if users of this type were considered.

There are a few conclusions regarding the performance and characteristics of the simulation. The most important one is that the safety conditions and convex hull should be used when possible. This may not be possible in all situations, but since generating safe trajectories is one of the core objectives for the simulation, this is important. When the solver does not perform as well, try and retract either the safety condition or the convex hull in order to relax the condition of the model and increase the area of convergence for the solver.

When it comes to comparing the results of this thesis to the problem formulation, there are several points of discussion. The simulation has been changed in a way where guaranteed safe trajectories in environments with obstacles can be produced to a large extent. However, as presented along with this successful implementation are some problems caused by both the solver and the addition of these conditions that lower the amount of degrees of freedom for the solver to work with. There are several problems with using both the convex hull and the safety conditions to their full extent in context of an obstacle profile that presents a difficult problem for the solver. These problems have been presented and discussed previously in the report and will not be repeated. What should be repeated however is how those problems affect the usability and practicality of the simulation. Due to how affected the solver and therefore the simulation is by the convex hull and the safety conditions, the trajectories generated for certain obstacle profiles should not be applicable to autonomous cranes. This is the case when the problem, model, convex hull or safety conditions are changed in order to reach an optimal solution, when not changing any of them would result in a problem that is not solvable, or the solutions are infeasible. This weakness to the simulation should be taken very seriously and should not be exploited. That is why the recommended use of the simulation in its current state is not to generate optimal solutions for sets of obstacle profiles, but to study and understand the behaviour of the model used and how these trajectories can be attained through non-autonomous means. Note that this recommendation is valid as of writing of this report and might change as the simulation or model is further developed.

5.3 Further development

As of the writing of this report, there are several areas of interest that can be further investigated in order to improve both the performance and the capabilities of the simulation. These areas include Homotopy, Discrete programming, nuanced model, flexible rope modelling and more. This section will be restricted to the areas mentioned above.

As descried in chapter 3, there seem to be several ways to reach appropriate initial guesses for the solver to start optimising from. Two of these are Homotopy and Dynamic programming. The latter seems exclusively interesting for the initial guesses of the simulation, as there is no way to use it for the actual optimisation performed by the solver. Dynamic programming would be an interesting area to further research and could lead a reliable and trustworthy method for acquiring initial guesses. Homotopy on the other hand presents a whole different potential for usage in the simulation. This potential includes the usage of Homotopy in the optimisation performed by the solver. This would require the solver to be warm started between iterations of the method of Homotopy, which could be a challenge to reach. Most importantly however, Homotopy presents some choices in its implementation which could be investigated further. These options include to include some of all the obstacles in the model to the solver a small step at a time. Or including the edge-most obstacles at first in order to focus the resources of the solver on getting over the first obstacle. Maybe introducing the highest obstacle first could give the solver a chance to adapt its final trajectory to the profile of the obstacles as they were inputted. All of these could be very interesting to investigate in order to chart what use cases of Homotopy are suited to this simulation.

Introducing a more nuanced model and system dynamics is mostly interesting for the comparison in both performance and usability. After this thesis work has been done, no validation of model characteristics has been made in comparison to a more advanced model. This may not be an issue at all, but it would be of interest for future use of this simulation to validate or denounce a difference in performance. The usability of a simulation containing a more nuanced dynamics model would present more complex calculations for the solver, and therefore prolong the simulation time for each iteration of the simulation. This might affect the usability of the simulation is a severe way, or not. This is of course of interest to investigate.

Furthermore, when starting this thesis work the agreement of work included the modelling of the ropes from the trolley to the load, which should not be a stiff body. This workload was scrapped in order to focus on the first objective of this thesis, producing safe trajectories in an obstacle filled environment. Since this area has yet to be investigated thoroughly, it is added to this section as a genuine area of interest for future research.

Bibliography

[1] Eihab M. Abdel-Rahman, Ali H. Nayfeh, and Ziyad N. Masouid. *Journal of Vibration and Control*, 9:863–908, 2003. URL `https://www.researchgate.net/publication/258162496_Dynamics_and_Control_of_Cranes_A_Review`.

[2] Zinober A.S.L. and Fuller A.T. *International Journal of Control*, 17:673–703, 1973.

[3] Kristoffer Bergman and Daniel Axehill. Combining homotopy methods and numerical optimal control to solve motion planning problems. volume IV, pages 347–354, Changshu, Suzhou, China, June 26-30 2018. IEEE. Found in 2018 IEEE Intelligent Vehicle Symposium (IV).

[4] Kariholoo B.L. and Parbery R.D. *Journal of Optimization Theory and Applications*, 36(3):409–417, 1982.

[5] Lars Blackmore, Hui Li, and Brian Williams. A probabilistic approach to optimal robust planning with obstacles. pages 2831–2837, Minneapolis, Minnesota, USA, June 14-16 2006. IEEE. Found in proceedings of the 2006 American Control Conference.

[6] Alsop C.F., Forester G.A., and Holmes F.R. Ore unloader automation - a feasibility study. pages 295–305, Tokyo, Japan, 1965. Found in Proceedings of IFAC Workshop on Systems Engineering for Control Systems.

[7] J.J. Hämäläinen, L. Baharova A. Marttinen, and J. Virkkunen. *IEE Proc. Control Theory Appl.*, 142:51–57, 1995. URL `https://ieeexplore.ieee.org/document/363557`.

[8] Virkkunen J. and Marttinen A. Computer control of a loading bridge. pages 484–488, Oxford, UK, 1988. Found in Proceedings of the IEE International Conference Control 88.

[9] Field J.A. *Transactions of the Engineering Institute of Canada*, 5(3):163–169, 1961.

[10] Yoon J.S., Park B.S., Lee J.S., and Park H.S. Various control schemes for imlemetation of the anti-swing crane. pages 472–479, Monterey, CA, USA, 1995. Found in Proceedings of the ANS 6th Topical Meeting on Robotics and Remote Systems.

[11] Beeston J.W. Closed-loop time optimal control of a suspended load: a design study. volume 39.5, pages 85–99, Warsaw, Poland, 1969. IFAC. Found in Proceedings of the IFAC 4th World Congress.

[12] Masahiro Ono Lars Blackmore and C. Williams. *IEE Transactions on Robotics*, 27:1080–1094, 2011. URL `https://ieeexplore.ieee.org/abstract/document/5970128`.

[13] Y Sakawa and Y Shindo. Optimal control of container cranes. *Automatica*, 18(3):257–266, 1983. URL `https://www.sciencedirect.com/science/article/pii/0005109882900863`.

[14] Ingrid Sjöberg. Modelling and fault detection of an overhead travelling crane system. Master's thesis, Linköping University, Division of Automatic Control, Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden, 2018.

[15] Yiming Zhao, Yebin Wang, MengChu Zhou, and Jing Wu. *IEEE Transaction on Automation Science and Engineering*, 16:327–338, 2019. URL `https://ieeexplore.ieee.org/document/8466117`.