Reachability and task execution speed analysis of a resonating robotic arm

Master of Science Thesis

For the degree of Master of Science in Mechanical Engineering at the Delft University of Technology

Wouter Jan Wolfslag

June 4, 2012

Department of BioMechanical Engineering · Faculty of Mechanical, Maritime and Materials Engineering · Delft University of Technology
Chapter 1

Introduction

Robotic arms potentially play a large role in industry. For many applications robotic arms are already used, but the application domain is potentially larger than it is now. In order to extend this domain, robots need to be made more efficient, lighter, safer and faster than they currently are. A very promising direction of research aimed at making those improvements is to adjust the passive dynamics of the robotic arm in design, and then control the arm in a way that uses these natural dynamics.

Studies have already shown that using natural dynamics can be highly beneficial when applied to a specific goal. Firstly, a large field of research has investigated the use of natural dynamics to make bipedal walkers more energy efficient [5],[4],[8]. Secondly, the passive effects of series elastic actuations have shown to increase the capability of a robotic arm to throw a ball a long way [2]. Lastly, the most recent study in this field resulted in the design of a resonating robotic arm [11].

A resonating robotic arm is a concept which improves the performance of a robotic arm executing a pick and place task, a common task for robotic arms in industry. Such a pick and place task is a highly repetitive motion. The idea behind resonating robotic arms is to design a mechanism that attaches to the robotic arm. This mechanism makes the repetitive motion a natural mode of the system. The pick and place task then becomes a large amplitude oscillation, hence the term resonating robotic arm. The resonating mode takes over part of the actuation, which makes the robot more energy efficient and safe.

The first prototype of a resonating robotic arm is called the Plooij mechanism, and will be the object of study in this thesis. It is a spring mechanism that actualizes the resonating arm idea by providing two characteristics. First, it provides the driving torque for a motion of a robotic arm between an initial pick position and a target place position. This provides the oscillatory motions required for pick and place tasks. Second, the spring mechanism makes holding the arm at the initial or target position effortless by having a torque of zero at these points.

In [12] a feed forward controller for this mechanism was optimized, resulting in an improvement in energy consumption of up to 20%. This energy optimization was done for only one motion of the robotic arm, which we think leaves three issues for further investigation. The aim of this thesis is to resolve these issues, which will be introduced below.

- The first issue is the capability of the Plooij mechanism to use a small input torque. A small input torque can be provided by a smaller actuator, which would make the robot cheaper, safer, lighter and more energy efficient. However, a limited input torque will also limit the capability to steer the robotic arm away from the passive dynamics of the spring mechanism, and thus give the robot arm a limited range of motion. This is especially a problem because friction makes the robot unable to reach the target position without motor torque, indicating that the range of motion with a motor torque limit could be too small. Therefore, in chapter 3 the following question will be answered:

  *What is the influence of a motor torque limit on the range of motion of the Plooij mechanism?*

- The second issue is the versatility of the Plooij mechanism. The robot is designed for one motion, and therefore one specific task. A main reason for industry to choose robotic arms
is their potential to perform a variety of tasks. It is untested if the natural dynamics of the Plooij mechanism are beneficial for a larger set of motions than the single motion for which it was originally designed. Therefore, in chapter 4 we will answer the following question: 

*What is the set of motions for which the Plooij mechanism has beneficial natural dynamics?*

- The third issue is the pick and place task execution speed. Execution speed is an important issue for industry, because of the high throughput of products that need to be picked and placed. Furthermore the speed should be provided by a limited input torque, as already discussed. We will investigate if the Plooij mechanism can be used to increase the task execution speed given a limited input torque. This would extend the Plooij mechanisms field of application beyond those where energy efficiency is the only concern. In chapter 5 we will investigate task execution speed by answering the following question:

*To what extend does the Plooij mechanism increase pick and place task execution speed given a limited input torque when compared to a standard robotic arm?*

Before we can move to answer these questions, a more detailed explanation of the Plooij mechanism and the model we used to describe it will be given in chapter 2. Lastly, chapter 6, will give a conclusion to the questions above.
Chapter 2

Plooij mechanism and its model

In this chapter we will give an overview of the Plooij mechanism, in order to understand its design and the model used in later chapters. This mechanism was the first design of the TU Delft Resonating arm project, which aims to make robots that perform their tasks in a natural dynamic manner. Figure 2.1 shows the configuration we will investigate. It consists of the Plooij mechanism attached to a one link robotic arm moving in the horizontal plane. The rest of this chapter will give a more detailed explanation of the design of the mechanism, followed by a description of the model we used.

Figure 2.1: The Plooij mechanism prototype. The Plooij mechanism is the first prototype of a resonating arm, a concept aimed to improve the performance of pick and place robots. The figure shows the setup of the prototype, consisting of the mechanism in the background and a one-link robotic arm. The hand is added to aid understanding and is not included in the prototype. Used with permission of Plooij.

Mechanism

The Plooij mechanism was conceived as a qualitative ideal for a potential energy function for performing pick and place tasks. The mechanical design was then created to fit this ideal. Our
explanation of the mechanism will follow these two steps as well.

The idea for the potential function is shown in Figure 2.2, and consists of three sub-ideas. First, the pick and place motion is meant to take place between the two half-stable outside equilibria, which makes it possible to easily hold the arm in place at the pick and place points, which would not be possible when using a linear spring. In this paper we will refer to the equilibria at a negative and positive angle as the initial point and target point respectively, see Figure 2.2. Secondly, the drop in potential energy between the pick and place positions is the driving force behind the motion, essentially providing a back-and-forth spring motion. Thirdly, the potential energy outside the equilibria continues to rise, which is beneficial because it helps reduce overshoot.

![Figure 2.2: The qualitative properties of the potential energy for pick and place tasks A mechanism for pick and place tasks ideally shows behaviour similar to a spring between the initial, pick, and target, place, locations to aid quick transition. This means a drop in the potential energy between these two points. Around the pick and place positions the potential energy should level off to make precise position easier. Finally, reverting back to normal spring behaviour in positions past the target provides a force that reduces overshoot. The combined shape of the potential energy function makes the initial and target points half stable equilibria.](image)

The sketch in Figure 2.3 shows the basic parts of the mechanism and their connections. It consists of two pulleys connected by both a timing belt and a spring. The first of these pulleys serves as platform to rigidly attach the robotic arm, and can be actuated by a motor, which would then work parallel to the mechanism. The prototype adds an extra degree of freedom, to create a SCARA type robot, but in this study we will only consider the single degree of freedom, as depicted.

Figure 2.4 shows the parameters used to describe the mechanism. The description of the parameters and their values in the prototype are shown in Table 2.1. For this parametrization, the ideal potential function is not exactly matched. Rather than one half-stable equilibrium at both the pick and the place location, it has a stable equilibrium and an unstable equilibrium close together around each of the pick and place locations. This slight alteration allows the mechanism to stay in its pick and place locations without a stabilizing controller.

**Model**

This study will be carried out completely in simulation, placing a high emphasis on the modeling. In this section the model we have used will be described and justified by discussing the assumptions and choices made to arrive at the model. This will be followed by a brief explanation of the simulation method used.

The description of the model will be done in two steps. First the mechanism is described, and second the actuation is discussed.
Figure 2.3: High level sketch of the Plooij mechanism attached to a one degree of freedom robotic arm. The sketch shows the mechanical design of the Plooij mechanism. It consists of two pulleys connected via a timing belt. A spring is connected to the two pulleys by means of connecting rods. Actuation is done by a motor directly connected to the arm.

Figure 2.4: The parameters that describe the Plooij mechanism. A description of the parameters and their values in the prototype are given in Table 2.1. Not included in this figure is $d_r$, the diameter of the attachment rods for the springs on the pulleys.

Spring mechanism
To model the mechanism, we assume that the mechanism adds no inertia, that the timing belt and pulleys are rigid and move without slip, and finally the spring is assumed linear. This means that the mechanism can be approximated by a torque that is only dependent on the angle $\theta$: $T_s(\theta)$. The full equations and their derivation are shown in appendix A. Friction is modeled as linear viscous damping with coefficient $d$, and is considered independent of the forces in the joints. The value of $d$ is chosen as $-0.25$ Ns.
Table 2.1: Parameters of the Plooij mechanism. The definition of the parameters is visualized in Figure 2.4

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Prototype value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_1$</td>
<td>Spring attachment radius on pulley 1</td>
<td>0.105 m</td>
</tr>
<tr>
<td>$r_2$</td>
<td>Spring attachment radius on pulley 2</td>
<td>0.02 m</td>
</tr>
<tr>
<td>$k$</td>
<td>Spring constant</td>
<td>150 N/m</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Mechanism position</td>
<td>- rad</td>
</tr>
<tr>
<td>$l_0$</td>
<td>Spring length at $\theta = 0$</td>
<td>0.1 m</td>
</tr>
<tr>
<td>$F_0$</td>
<td>Spring force at $\theta = 0$</td>
<td>6 N</td>
</tr>
<tr>
<td>$d_r$</td>
<td>Diameter of attachment rods for springs</td>
<td>0.005 m</td>
</tr>
<tr>
<td>$a$</td>
<td>Pulley ratio</td>
<td>5 rad/rad</td>
</tr>
</tbody>
</table>

The model of the robot set-up will rigidly attach a single rigid link to the mechanism, and place the mechanism such that it will be moving in a horizontal plane. A point-mass at the end of the link will serve as model of an end effector, leading to a combined inertia $I$. The value of $I$ is taken as 0.1840, caused by a weight of 1 kg at the end of an arm of 0.6 kg and 0.4 m long. The state space of the complete system thus consists of only one angle, $\theta$ and its angular velocity, $\omega$. When the Plooij mechanism is compared to a standard robotic arm, the only change is that $T_s(\theta)$ is set to 0.

**Actuation**

To allow us to focus on the effects caused by the mechanism, rather than those caused by, possibly ill chosen, motor characteristics, a very simple actuation model is needed. We simply use a torque: $T$. The absolute value of this torque is limited by a maximal value $T_{max}$. Aside from that, no limits are set and it is assumed that discontinuities in the torque signal are possible. This last fact will be used extensively, as the optimal control signals studied will all turn out to be of bang-bang type. The equations of motions are the ordinary differential equations shown in equation 2.1.

$$
\begin{pmatrix}
\dot{\theta} \\
\dot{\omega}
\end{pmatrix} = \begin{pmatrix}
\omega \\
\frac{1}{I} (T_s(\theta) + d \cdot \omega + T)
\end{pmatrix}
$$

(2.1)

**Simulation of the model**

To simulate the equations of motions we used the ode45 function of Matlab. This function implements the Dormand Prince method, which is a fifth order explicit Runge Kutta scheme. It has the added property that a fourth order scheme is part of the calculations allowing quick error estimation, and with that automatic stepsize selection. As such it is a fast and reliable integration method for our purposes.

**Overview**

In this chapter we have described the Plooij mechanism and our modeling of it. The main result which caries to the remainder of this paper are the equations of motions we will use in simulation. The equations of motions are visualized in Figure 2.5 by means of the vector field of the passive system. Two typical passive trajectories are given as example.
Figure 2.5: Vector field and two passive trajectories showing phase-plane of the Plooij Mechanism model. The equations of motion of the model of the Plooij mechanism are visualized in this state space plot. The passive trajectories shown converge to the target point and 0 rad position. Also shown are the initial and target points, both equilibria of the passive system.
Chapter 3

Range of motion with motor torque limit

The Plooij mechanism was designed to aid the motion between a pick and a place location. Yet when limited motor torque is available, friction in the mechanism can cause the robot to get stuck in the potential energy minimum between these two points, and thus fail to reach the target. Furthermore, such limited motor torque will also make it more difficult to steer the robot to other positions than the exact target point, which is necessary for many industrial applications. This means that the range of motion of the Plooij mechanism with limited motor torque might be too small. However, a small motor, and thus limited motor torque, is required because it makes the robot cheaper, lighter and safer. Therefore a trade-off between range of motion and input torque must be made. This chapter will explore that trade-off.

To describe the range of motion the one swing reachable set is introduced. This is the set of points in state space that can be reached from an initial position without moving back-and-forth in a swing-up sequence. Disallowing such back-and-forth motions ensures that only points that the mechanism can reached directly -and thus easily- are taken into account. Three variations of the one swing reachable set were computed. Section 3.1 explains what these variations are, and how to compute them, with section 3.2 showing and analysing the results. Section 3.3 will draw conclusions from these results, followed by further recommendations in section 3.4.

3.1 Computation of the one swing reachable set

The three variations of the one swing reachable set will be discussed in this section. This will be done by giving a description of each variation, including the trajectories that make up the boundary of the reachable set. Note that these boundaries are numbered for cross referencing with Figure 3.1 and 3.2, which show the motor torque in the trajectories, and the trajectories in statespace.

Forward, only one direction The first variation of the one swing reachable set is computed forward in time, meaning that the set consists of those points that can be reached from the initial point in one swing. This -one swing- demand is expressed as a constraint that the motion is only in one direction, i.e., the sign of the angular velocity does not change. Note that this set therefore consist of two subsets: one with a positive velocity, and the other with negative velocity. The boundary of each of these sets are found by two trajectories of the mechanism. Below the description of the control signal for these trajectories is given, first for the reachable set with positive velocity, then with negative.

Positive velocity

1. The outer boundary is given by the controller $T = T_{max}$. Any point further away from the origin is outside the reachable set.
2. The inner boundary uses a slightly more complex control signal. This boundary is also in a positive direction, but with the lowest possible velocity as allowed by the spring mechanism. The control approach is to initially move in positive direction, keeping the velocity at nearly 0. Initially this requires a positive torque, but the required torque will becomes negative when $T_s(\theta) > 0$. At some point in the motion the required force $T_{req}$ equals $-T_{\text{max}}$, see Figure 3.1. When moving past this point the control will simply be $T = -T_{\text{max}}$, slowing the arm maximally.

**Negative velocity**

3. The outer boundary uses $T = -T_{\text{max}}$.

4. The inner boundary does not require a two phase feed forward signal. Because the mechanism already provides a torque to slow the arm down in this area of state space, the $\omega = 0$ line can simply be followed. To reach the points faster, the motions can start with $T = -T_{\text{max}}$ and switch to $T = T_{\text{max}}$ at the appropriate time.

**Forward, not circling origin**

The second variation of the one swing reachable set is also computed forward in time, but with a more relaxed constraint on back and forth motions. The constraint is now that trajectories are not allowed to circle the origin of the state space. This slight relaxation leads to a larger reachable set, without violating the idea that the motions should be mostly direct. The control signals for to find the boundary of this reachable set are as follows:

5. The outer boundary requires two switches in velocity. Initially, the trajectory is the same as in trajectory 2. However, instead of moving past the point $T_{req} = -T_{\text{max}}$, the maximal negative is applied shortly before reaching this point. As a result, the robot arm will move away from the target point until the mechanism torque forces the robot arm to a standstill again. At this point the final phase of the trajectory starts and the control torque switches to $T_{\text{max}}$.

2. The inner boundary does not change between the two definitions of one swing motions. It is therefore the same as the inner bound of the forward one direction reachable set, trajectory 2.

**Backward, not circling origin**

The third variation is computed backward in time. This means that the simulation starts at the target point, and moves backward in time to find all points from which the target point can be reached. Such a backward reachable set can therefore be seen as a measure of robustness, as it indicates from what points, and thus disturbances, the target can be reached.

For the one swing backward reachable set, only the constraint that disallows circling the origin will be considered, as this relaxed constraint allows a little room for maneuvering around the target point, much alike overshoot in traditional control. The constraint that does not allow changing of direction does not give this room for maneuvering, which is the reason we deem it too strict for this backward-in-time case. The two control signals that lead to the trajectories that form the boundary of this reachable set are given below.

6. The outer boundary is a backward version of trajectory 5, but with reversed torques and starting at the target point. Instead of switching torque input at the point $T = -T_{\text{max}}$, switching occurs at $T = T_{\text{max}}$.

7. The inner boundary requires two switches in velocity. Initially, the trajectory is the same as in trajectory 2. However, instead of moving past the point $T_{req} = -T_{\text{max}}$, the maximal negative is applied shortly before reaching this point. As a result, the robot arm will move away from the target point until the mechanism torque forces the robot arm to a standstill again. At this point the final phase of the trajectory starts and the control torque switches to $T_{\text{max}}$. 


Figure 3.1: Control strategies for boundary of one swing reachable sets. This figure shows the motor torque in each of the 5 boundaries for the forward reachable set. The torques are shown as a function of position, with the progress of time indicated by the arrow. The arrows indicating $-T_s$ shows the regions for which the motor torque is the opposite to the spring force, driving the robotic arm at an infinitesimal speed. The arrow indicating $\omega = 0$, indicates that the switch in torque at that point occurs because the angular velocity of the arm goes through 0.
3.2 Analysing the one swing reachable set

In this section the three variations of the one swing reachable set of the Plooij mechanism will be shown and analysed. First, the three variations of the one swing reachable set are shown and compared. Then, the two forward in time versions are shown for varying $T_{\text{max}}$. Finally, the backward one swing reachable set is discussed.

Figure 3.2 shows the seven boundary trajectories in two graphs. These trajectories form four reachable regions, each bounded by two of the trajectories.

- The first region is bounded by trajectories 1 and 2, and consists of those points that can be reached while only moving forward.
- The second region is bounded by trajectories 3 and 4, and consists of those points that can be reached while only moving backwards. This region is of limited interest because trajectories of the Plooij mechanism in a practical setting move towards the target point.
- The third region is bounded by trajectories 2 and 5, and consists of those points that can be reached without circling the origin.
- The fourth region is bounded by trajectories 6 and 7.

The first and second region combined make the forward reachable set without changing direction. Because the second region is small and in an irrelevant direction, it is excluded from further analysis. The third region is the forward reachable set for trajectories that do not circle the origin. Region four is the backward reachable set, also for trajectories that do not circle the origin.

Figure 3.3 shows both forward types of one swing reachable set for varying $T_{\text{max}}$. In both cases the reachable set forms a relatively narrow tube moving from the initial point to some point short of the target points, which shows how the mechanism forces the robotic arm into a motion with little room for variation. For higher torques this tube suddenly expands to form a bulb around the target point, showing that the mechanism allows free movement around that point. This expansion occurs around the angle where $T_s(\theta) = -T_{\text{max}}$, a part of the trajectory where $T = T_{\text{max}}$. Therefore, at the point of expansion, the combined torque on the arm becomes positive, allowing the arm to continue its motion. In Figure 3.4 the expansion effect is visualized. For the reachable set that only uses positive velocity, the minimal torque required to reach this expansion point is $0.45 Nm$. Note that a further increase of motor torque only leads to a limited increase in the size of the reachable set.

Figure 3.5 shows the one swing backward reachable set. The backward reachable set is similar in shape to the forward reachable set, however it is much smaller. This difference in size is caused by friction. Damping enlarges the backwards reachable set by providing an extra braking torque, while this same braking torque limits the size of the forward reachable set.

The effect of the value of the maximal input torque is not the same along all parts of the boundary. Specifically, the difference in size of the reachable set between torques is much larger at the boundary away from the origin than for the boundary near the origin. As such, it is the inner boundary that can be seen as a measure for the robustness of the robot against large disturbances. If the actuators maximal torque is too low, it means that a relatively small disturbance can bring the system into a state below this boundary, causing failure to reach the target.

3.3 Conclusion

This chapter aimed to investigate the effect of a motor torque limitation on the range of motions the Plooij mechanism can perform. This was accomplished by introducing the one swing reachable set: the set of points that can be reached without back and forth motions. The main conclusions are:

- The one swing reachable set forms a tube between the initial point and target point.
Figure 3.2: Boundary trajectories. This figure shows the boundary trajectories of the two one swing forward reachable sets in state space for a motor torque of 0.45 N. Region 1 is the forward, one directional reachable set with a positive angular velocity. It combines with the much smaller region 3 (which is the set for negative angular velocity) to make the one-directional reachable set. Region 2 is the reachable set for trajectories not circling the origin. Region 4 is the backward reachable set for trajectories not circling the origin. The boundary trajectories are numbered in the same order as in their explanation in section 3.1 and Figure 3.1.

Figure 3.3: Forward one swing reachable set. The left graph shows the one-directional reachable set for four different torques. For a torque of 0.46 the outer boundary trajectory almost reaches 0 angular velocity when approaching the target point. However, the control torque is sufficiently large to remain forward motion. The outer boundary for torques larger than 0.45 show a smaller decrease. A torque smaller than 0.45 will not reach the target point. The graph on the right shows the reachable set for motions that do not circle the origin. This reachable set is somewhat larger. Most noticeable feature of this graph is the narrow tube the reachable set forms between the initial and target points. Around these two points the tube expands, becoming much wider. This tube and expansion effect shows off the properties of the Plooij mechanism design: a driving torque that forces the arm into a motion between the initial and target points combined with the capability of free movement around the initial and target point.
Angular velocity \[ \text{rad/s} \]

Figure 3.4: Visualization of the expansion point by sample trajectories and torque characteristic. With a constant torque as input, there is an unstable equilibrium between the 0 rad position and the target position. This figure shows that a large divergence in trajectories takes place around this unstable equilibrium. This unstable equilibrium can be interpreted as a bottleneck point, because trajectories around this point move with low velocity before either failing and falling back to the origin or jumping forward to reach the target point and beyond.

Torque \[ Nm \]

Figure 3.5: Backward one swing reachable set. The backward one swing reachable set is similar in shape to the forward one swing reachable set. It shows from which points in state space the target position can be made, so it can be seen as a measure for robustness against large disturbances.
• For a torque larger than $0.45\,\text{Nm}$ this tube expands to an area in state space with low velocities around the target point. For a lower torque this expansion does not take place and the target point cannot be reached. Further increasing the input torque has limited effect.

• The backward reachable set can be seen as a measure of robustness against large disturbances. The most critical part of the boundary of this set is the inner boundary, especially because increasing the available torque has limited effect.

3.4 Recommendations

The one-swing reachable set also caused us to identify questions for further research. This section lists the three we think are most important.

• Firstly, trajectories inside this reachable set can be divided by a part inside a narrow tube, and a part within the expansion. It seems that control during an expansion phase will be more in line with the natural dynamics than control during a tube phase. Can we incorporate knowledge of tube and expansion phases in a controller? One idea would be to implement a controller that uses a feed forward signal during each of these phases. This is effectively a model predictive controller, where the horizon is the end of one of these phases. Investigating such an approach is left for later research.

• Secondly, for the one degree of freedom robot investigated here, there is limited variety in which to move between two points in space, especially considering the restraint on back and forth motions. However, when there are redundant degrees of freedom, many motions become possible between two points in state space and many pairs of points in state space can describe one pair of points in actual space. Future research should investigate whether this redundancy can be used to make mechanisms capable of performing a wider array of different tasks while using its natural dynamics, and how we can translate actual space into state space to find optimal solutions.

• Thirdly, the constraint of not moving back and forth is a very application specific use of the idea that motions that repeatedly cross a certain part of state space are not in line with the natural dynamics of the task. In future research we will aim to use this more general notion of non-natural motions to quickly assess a mechanism’s suitability for a task, allowing partially automated mechanism design.
Chapter 4

Versatility of the Plooij mechanism

The natural dynamics of the Plooij mechanism have been shown to be beneficial to a pick and place task. However, this was only shown for one particular task, whereas in practical applications robotic arms are required to perform a variety of tasks. Therefore we ask what set of motions benefit from the natural dynamics of the Plooij mechanism. This chapter shows these motions as a reachable set in state space, similar to the one swing reachable set from the previous chapter.

The key difference is that instead of the one swing limitation we now use a time limit, hence we compute the time limited reachable set. By comparing the time limited reachable set of the Plooij mechanism to that of a standard robotic arm, we can see what additional motions are possible with the Plooij mechanism, and which motions are no longer viable. This method implicitly uses time to reach, and thus speed, as a measure of performance in order to establish for which motions the Plooij mechanism is beneficial. This measure is relevant because of the high production throughput for robots applied in industry.

The layout of this chapter is as follows. First the computation of the time limited reachable set is discussed. Then the results of the computation are shown and analysed. Finally we draw a conclusion on the versatility of the Plooij mechanism and give recommendations for further research.

4.1 Computing the time limited reachable set

In this section the computational method we used to find the time limited reachable sets is discussed. We will start with an explanation of two currently available methods and why we have chosen not to use them. Then two attempts at using optimal control methods will be briefly described, including the reasons why they failed. Finally, the novel method we used, which is a variation of one of the attempted optimal control methods, will be explained.

4.1.1 Current methods

The two approaches to computing the reachable set that are now most commonly used are polygon approximation and front evolution methods. The discussion below explains why these two methods were not used.

- In polygon approximation, first a polygon is set as an approximation of the reachable set at time $t = 0$. From there on, the possible trajectories at the vertices of the polygon are computed, which creates a new polygon for the next time step. This method is often used in safety verification because it is fast and is guaranteed to find an over-approximation of the reachable set. Unfortunately no concave sets can be approximated this way, and the approximation is not very tight. This makes the computation too inaccurate for our purposes.
• In front evolution, we try to integrate the boundary of the reachable set forward in time, rather than approximate its evolution by polygons. There are various ways to do this, but the most promising are the Level Set Methods. In these methods, an auxiliary function of time and state variables is created in such a way that the reachable set at a certain time are those points for which the auxiliary function equals 0. The evolution of the auxiliary function can then be written as a partial differential equation, which can be numerically solved by means of finite differences. The ideas are represented clearly in textbooks [10][14], with applications in the area of robust control in [7][15]. Unfortunately these methods are numerically very complicated and sensitive. Therefore this method was not implemented.

4.1.2 A grid of optimal control solutions

Because existing methods did not apply, we proposed a new approach: computing the minimal time to reach for a grid of points in state space. Level sets of this minimal time to reach function would be the required reachable sets, and are easily found using contour functions available in Matlab. The minimal time to reach can be computed using optimal control techniques. In optimal control there are two roads to obtaining the optimal feed forward control signal for a task. These two roads form the basis of two computation methods for the reachable set that failed to work.

Indirect method

The first road is called the indirect method, in which analytical optimization takes place, which leads to the formulation of a boundary value problem, see [9]. This boundary value problem is then solved numerically by discretising state space and time. The indirect method gives insights in the structure of the control signal, but unfortunately is not suitable for this task because the boundary value problem is difficult to solve. There are two reasons for this difficulty. First, nonlinearities cause problems with convergence for shooting methods, because the optimal solution has a small basin of attraction. Second, the optimal control turns out to have non-differentiable points in the control signal. This non-differentiability of the control signal causes problems for the boundary value problem solver in Matlab. For these two reasons this method is not used.

Direct method

The second road is called the direct method in which state space and time are first discretized and then numerical optimization takes places. This road is often preferred as it tends to solve the issues with the small basin of attraction in indirect optimal control. Furthermore, there is software written specifically to perform the discretization and optimization tasks involved, via collocation and pseudospectral methods. This software uses algorithms that use the sparsity of the optimization, allowing for fast computation. First, a simple collocation scheme, see [3], was implemented, which worked well for most individual runs of the optimization. However, when trying to compute the reachable set there were issues with computation time and the robustness of the scheme against initial guesses. Therefore the GPOPS package[13], which uses the pseudospectral method, was tried. This mostly solved the problems mentioned above but showed two new flaws:

• Trouble with non-differentiability: The parametrization does not allow fast convergence in case of discontinuities or non-differentiability. A solution for this was sought in the form of breaking up the whole optimization into differentiable pieces, however analysis via indirect optimal control showed that the number of discontinuities is unknown a priori.

• Non feasible solutions: There are some points in state space that are not reachable even for large $t$. In non-reachable points the optimization gets stuck, and takes very long. Although this doesn’t prevent the method from working, it is a very significant issue for the computation time.
4.1.3 Indirect optimal control with initial costate variation

This section will explain the computation method we used for finding the time limited reachable set. Searching for optimal control solutions for a grid of target points in state space did not work. However, indirect optimal control allows us to grid over a different set of variables, as we will explain in this section. With that different grid, a workable method was found. Before explaining this new grid, the indirect optimal control method as it applies to our problem is briefly explained.

In indirect optimal control an analytical optimization is performed with calculus of variations. This leads to a boundary value problem, consisting of a system of ordinary differential equations and a set of boundary constraints. The differential equations are formed by the optimization goal and the equations of motion of the system that needs to be controlled. The boundary values specify the initial and final conditions of the specific problem at hand. First, the approach from [9] is used to derive the differential equations.

The first step in deriving the differential equations is to write down the Hamiltonian, \( \mathcal{H} \).

\[
\mathcal{H} = 1 + \lambda_1 \omega + \frac{\lambda_2}{T} (T_s(\theta) + d \cdot \omega + T)
\]  

(4.1)

The second step is find the optimal control, \( T^* \) by setting the partial derivative of the Hamiltonian to the control variable to zero. The control equation found this way is then substituted in the Hamiltonian to create the optimal Hamiltonian, \( \mathcal{H}^* \):

\[
T^* = \arg \min_T \mathcal{H} \rightarrow T^* = -\text{sign}(\lambda_2) \cdot T_{\text{max}}
\]  

(4.2)

so:

\[
\mathcal{H}^* = 1 + \lambda_1 \omega + \frac{\lambda_2}{T} (T_s(\theta) + d \cdot \omega - \text{sign}(\lambda_2) \cdot T_{\text{max}})
\]

The last step uses \( \mathcal{H}^* \) to find the differential equations for the state and costate, again by taking partial derivatives. Equation 4.3 shows the required derivatives and their solution.

\[
\begin{pmatrix}
\dot{\theta} \\
\omega \\
\lambda_1 \\
\lambda_2
\end{pmatrix} = 
\begin{pmatrix}
\frac{\partial \mathcal{H}^*}{\partial \lambda_1} \\
\frac{\partial \mathcal{H}^*}{\partial \lambda_2} \\
-\frac{\partial \mathcal{H}^*}{\partial \omega} \\
-\frac{\partial \mathcal{H}^*}{\partial T}
\end{pmatrix} = 
\begin{pmatrix}
\frac{1}{T} (T_s(\theta) + d \cdot \omega - \text{sign}(\lambda_2) T_{\text{max}}) \\
2 \lambda_1 d T_s(\theta) \\
-\lambda_2 \\
-\lambda_1 d
\end{pmatrix}
\]  

(4.3)

The normal use of these equations is to set boundary conditions on initial and final state. This leaves the initial costate free to be varied. A shooting method is used to find the initial costate that matches the initial and final state conditions. The optimal control signal can then be found from equation 4.2.

For our application (computing the time limited reachable set) we need to solve a family of optimal control problems, all with the same initial state but different final state. The only thing that can be varied is the initial costate, meaning that every initial costate must correspond to a final state. This means that computing the reachable set can be done by integrating the optimal control differential equations for all possible initial values of the costate. This is useful because instead of making a grid of the final states and optimizing the initial costate for each grid point, we can make a grid of initial costates and immediately find the associated final state, without numerical optimization.

The boundary conditions of the optimal control problems that needs to be solved allow some rewriting which leaves only one costate parameter to be varied. In free final time optimal control problems, the boundary constraint that determines the final time is \( \mathcal{H}^*(t = t_f) = 0 \). Furthermore, when there is no direct dependence of the Hamiltonian on time, the Hamiltonian is constant. Therefore, for our problem, which has a free final time and no direct dependence of...
the Hamiltonian on time, it holds that $H^*(t = 0) = 0$. Because our initial state has $ω = 0$ and $T_s(θ) = 0$, this reduces to:

$$λ_2(t = 0) = ± \frac{I}{T_{max}}$$

This leaves just one free initial parameter to choose: $λ_1$, which can vary from $−∞$ to $∞$. This has to be done twice, both for positive and negative initial $λ_2$.

### 4.1.4 Interpretation of the costate

In this section the interpretation of the costate will be discussed. This interpretation allows us to grid the initial costate in a more practical way.

In the case of time optimal control a property of the $λ$-vector shows up. If we would evolve the boundary of the reachable set, as in section 4.1.1, then on all points on the boundary the control input would be chosen to maximize the increase in size of the reachable set. This is done by maximizing the movement in the direction normal to the reachable set boundary. Equation 4.5 shows the required maximization, using $N$ for the vector normal to the boundary of the reachable set and $u$ for the control signal.

$$u^* = \arg \max_u \left( N \cdot \left( \frac{ω}{u + T_s(θ) + d \cdot ω} \right) \right) = \arg \max_u \left( N_1 \cdot ω + N_2 \cdot \frac{u + T_s(θ) + d \cdot ω}{I} \right)$$

The argument of the maximum does not change if the above function is multiplied by $−1$ and then minimized. We also add the constant 1, which also does not change the maximization, to obtain:

$$u^* = \arg \min_u \left( 1 - N_1 \cdot ω - N_2 \cdot \frac{T_s(θ) + d \cdot ω}{I} \right)$$

When substituting $λ$ for $−N$ the expression for $u^*$ becomes exactly the same as the expression for $T^*$ in equation 4.2. This means that $λ$ is always pointing in a direction perpendicular to the reachable set surface.

The interpretation of the costate as a direction gives two advantages. First, it is possible to calculate the reachable set when starting from an initial set rather than a point, by simply assigning the initial costate at each point on the boundary as the normal direction. Second, instead of creating a grid over $λ_1$ when computing the reachable set, we can create a grid over directions, where the length of the costate vector is known because $λ_2$ is known from equation 4.4. In practice, this grid over directions leads to a more even spacing of data points on the reachable set boundary when compared to using a grid of $λ_1$ between two values. This improvement is caused by the even spacing in the initial preferred direction.

### 4.2 Analysing the time limited reachable set

This section will show and analyse the results of the time limited reachable set computation in order to asses the versatility of the Plooij mechanism. First the evolution of the time limited reachable set will be shown. Then we will compare the reachable sets of the Plooij mechanism with that of a standard robotic arm. In all the computations a maximal input torque of 1 N is used, along with a step size in the initial costate vector angle of 0.005 rad, meaning 1256 simulations.

Figure 4.1 shows how the time limited reachable set evolves over time, for a maximal input torque of 1 N. At first, the set is shaped almost as an ellipse, with the exception of a thin spike in the direction of the target point. The ellipse part of the reachable set shows relatively little change over time, but the spike extends further and wider, much like the tube we saw in the one swing reachable set. At $t = 1.5s$, a large part of the reachable set has a negative velocity, moving back to the initial point. Three features are noticeable:

- The target point is reached by an extension in the reachable set that forms between 0.84 and 1.12 seconds. This expansion causes the part of the reachable set that moves back towards the initial point to consists of two lobes.
• The boundary of the reachable set seems to cross itself, creating an internal region. This region is caused by the optimization method: setting the costate gives local optima only. The points inside the internal region have two local optima, meaning we should be careful to select the global optimum when referring to points inside this region.

• The reachable set only shows slow growth at most of its boundary, but grows fast at the area near the tip of its spike. Initially the tip of the spike is the trajectory that only uses positive torque.

In Figure 4.2 the comparison between the Plooij mechanism and a standard robotic arm without any torque from a spring mechanism, but with the same parameters otherwise, is made. This is done by showing the reachable sets at 1.12 s for both the Plooij mechanism and the standard robotic arm. Two regions of note are inside the reachable set of the Plooij mechanism but outside that of the standard arm. First there is the area with an angle around zero and with a large angular velocity. This angle range is where the potential energy of the spring is minimal. The second where the reachable set of the Plooij mechanism arrives first is an area around the target point, with low angular velocity. A more detailed investigation of the gain in time needed to reach this second set is the subject of chapter 5.

Figure 4.3 shows what regions of state space are reached faster by the Plooij mechanism, and what regions by the standard arm. This figure was made by finding the points where the reachable sets of the Plooij mechanism and standard robotic arm intersected. Only points that can be reached within 1.3 s were investigated, to prevent overly long lasting motions to be taken into account. The region for which the mechanism is faster initially forms a narrow tube directly towards positive velocities, which levels off at a velocity of around 3 m/s. The tube then expands and drops towards lower velocities at a position around the target point. Therefore, the shape of the region for which the Plooij mechanism is faster further supports the finding that the Plooij mechanism is faster in two main areas: one for high velocities around a 0 rad angle, and another around the target point with low angular velocity. However, not that these areas are both part of one connected region.

![Figure 4.1: Time limited reachable set](image)
Angular velocity \([\text{rad/s}]\)  

Angle \([\text{rad}]\)  

<table>
<thead>
<tr>
<th>-3</th>
<th>-2</th>
<th>-1</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
</table>

Without mechanism  
With mechanism

Figure 4.2: Comparing time limited reachable set for Plooij mechanism and standard robotic arm.

This graph shows the reachable set of both the Plooij mechanism and a standard robotic arm at \(t = 1.12\) s, for a torque of 1 Nm. The time limited reachable set of the standard robotic arm extends equally towards positive and negative angles. Therefore its reachable set is larger than that of the Plooij mechanism. There are two regions in the state space for which the Plooij mechanism is faster; one at high velocity around a 0 rad angle, and one around the target equilibrium point.

4.3 Conclusion

The goal of this chapter was to assess what range of motions benefit from the Plooij mechanism. This was done by computing the time reachable set of the Plooij mechanism and then comparing it to that of a standard robotic arm. It was found that the standard robotic arm is faster than the Plooij mechanism in reaching most of the state space, indicating the standard arm is more versatile. However, a region was found for which the Plooij mechanism was faster. This region mainly consists of two connected areas: the first an area with low velocities around the target point, the second an area of large velocities around the 0 rad position.

4.4 Recommendations

A number of questions and ideas for future research arose from this time limited reachable set analysis. The five we think most interesting are listed here.

- **Using more degrees of freedom:** Firstly, the reachable sets computed here only involve a one degree of freedom system, meaning only a 2D state space. For larger systems, this analysis might still prove useful, but is currently limited in two ways. First there is the computation time. The current computation runs 1256 simulations and takes about 50 s on a standard desktop computer\(^1\). Increasing the dimension will lead to an exponential increase in time, requiring more than 2.5 years to compute the answer for a 2 degree of freedom arm, with a 4D-state space. By using a dynamic grid for the variations of the costate that adjusts itself according to the required accuracy, improvement can be made. From the step sizes in state space in our static method, this improvement is estimated to reduce the computation time.

\(^{1}\)The processor used is the Intel Core i5-2400 CPU @ 3.10 GHz
Angular velocity \( \text{rad/s} \)

Angle \( \text{rad} \)

\[ -3 -2 -1 0 1 2 \]

\[ -3 -2 -1 0 1 2 3 4 \]

**Figure 4.3: Regions which benefit from Plooij mechanism.** This figure shows the set of state space that is reached faster by the Plooij mechanism; and the set that is reached faster by a standard robotic arm. An motor torque of 1 Nm was used, and the time to reach was limited 1.3 s, because longer motions are not relevant for possible applications.

- **Analysing control schemes:** The second proposed area of research is to compute the reachable set caused by a disturbance acting on a controlled system. This has already been done under the heading of safety verification [7], but would be new for this area of robot research. The main difference between the proposed research and the safety identification that has been done is the direction of the reachable set. In available research the focus is on backward reachable set, to verify that some bad states, such as collisions, cannot possibly occur. The new direction would involve forward reachable set, and view this set as a measure of task performance. As such it analyses the robustness of task execution.

- **Using a more complex model:** Thirdly, a recommendation for better modeling of the system by incorporating Coulomb friction. At first this seems a trivial addition, but the problem is that impulsive effects in the costate start to occur [6]. Furthermore, Coulomb friction can lead to the robotic arm being unable to move from its place. The effects, and perhaps...
even uses, of Coulomb friction warrant a separate investigation.

- **Aiming for other objectives:** The last area of proposed research are different optimization goals. The algorithm allows the use of other optimization goals than execution time. An obvious choice would be to look at energy. We could add such an extra goal in two ways. First it can be used as a replacement of the $V(x, u)$ function in optimal control, as such it would replace the 1 in the Hamiltonian of equation 4.1. The second option would be to add it as a constraint. The value of this optimization goal could be tracked as a state. The optimal control would then be a partially fixed final state problem. This would lead to a state space one degree higher than the original system, and hence one more costate to vary. The constraint on the final state would make the solution form a hypersurface again. One way to implement this final state constraint in the computations would be to solve the optimal control problem for one instance and then use the coordinate projection method [1] and contour-finding methods to walk along the constraint-line caused by the fixed final state.

It is important that much care is taken in choosing the alternate optimization goal, and the method in which it is incorporated in the algorithm. For instance, incorporating the energy consumption as an equality constraint does not provide a meaningful analysis. This because such a constraint leads to motions of a very different timescale, which are hardly comparable. A further downside is that motor and gearbox effects are now also taken into account, which means that it is difficult to find the origin of the effects.

An interesting measure would be to optimize for the change in energy caused by the actuation: the integral of the absolute value of the input power. In our case, this leads to $V = |\omega \cdot T|$. The use of energy change as measure of how natural the motion is seems a promising direction of research, so looking at that optimization more carefully is recommended for further studies. Furthermore, a separate investigation of bang-coast-bang controllers, which are optimal for this objective, will likely prove fruitful.
Chapter 5

Task execution speed

Pick and place robotic arms are required to execute their task at high speed in order to cope with the large throughput encountered in industrial applications. The current method to enable this high task execution speed is to use large actuators, and therefore actuation torques, to provide the accelerations and decelerations required. The issue is that it would be preferred to use small actuators, because they allow the robot to be made lighter, cheaper and safer.

To determine if the Plooij mechanism could resolve this issue, this chapter investigates to what extend this mechanism increases pick and place task execution speed for a given motor torque limitation when compared to a standard robotic arm. This investigation builds upon the research in [11] and [12] by using task execution speed instead of energy consumption as performance measure. Furthermore, it provides a more application oriented investigation of the task execution speed than the versatility oriented analysis found in the previous chapter.

To give a full overview of the increase in task execution speed we investigate the motions between one initial point and a range of stationary target points, a realistic portrayal of a pick and place task. The following sub-questions are used to guide this in-depth investigation of the task execution speed of the Plooij mechanism.

1. *How much does the current Plooij Mechanism improve task execution time when compared to a standard robotic arm?* This is the most basic question of the main investigation, and will be answered using two measures. The first measure is the average minimal time to reach for motions between initial point and target range. The second measure is the time to reach for the worst case point in the target range. These two measures show the expected and worst case performances for the pick and place task.

2. *What is the effect of the torque limit on this improvement?* Clearly, the motor torque available greatly influences the task execution speed. It likely also influences the improvement between a robot with mechanism and one without. For instance, consider that a very low motor torque might limit the way in which we can use the dynamics of the mechanism to improve task execution speed.

3. *What is the effect of the mechanism’s parameters on this improvement?* Perhaps it is possible to further increase the improvement by altering some of the mechanism’s parameters. This question is answered by means of a parameter optimization. The measure for task speed used in this optimization is the worst case time to reach. This measure is chosen because it prevents the optimization algorithm from trading a poor performance in one area of the target range for increased performance in another area.

The outline of this chapter is as follows. First, section 5.1 will explain the methods used to analyse the task execution speed. Secondly, section 5.2 will show and analyse the results of the computations. Finally, sections 5.3 and 5.4 give a conclusion and recommendations for further research.
5.1 Methods

The computational method to find the time optimal controller and its bang-bang nature, are already explained in section 4.1.3. However, in order to perform the analysis as indicated, three issues must be addressed. First, the range of target points in our pick and place task must be justified. Secondly, the method for numerical optimization must be selected. And lastly, constraints on the parameters must be defined in order to obtain realistic results in our optimization.

Range of target points
The range of target points is taken as $\pm 0.2$ rad around the equilibrium point, with 0 angular velocity. This range was taken as a fair representation, however it is not a range that is defined by physical limitations or literature. Therefore the influence of altering this range must be analysed, which is done by finding the derivative of the results to the size of the target range. We shall see that this derivative is small for all results, which means that these results will hold for a reasonable range of task variations around the $0.2$ rad chosen, justifying this choice.

Numerical optimization
At two points in this analysis numerical optimization is required.

- **Finding the point in the target range that takes longest to reach.** This is done by the Matlab function `fminbnd`, which uses a golden section approach to find the initial costate vector that leads to the worst-case point. The constraint on the target range is implemented with a penalty-function that increases linearly outside the feasible domain.

- **Finding those mechanism parameters that minimize the worst case time to reach.** For this we used the Matlab function `fmincon`. This is a quasi-Newton method which uses an active set method to keep track of inequality constraints. This method uses gradient information to locally optimize the parameters. The underlying assumptions that the problems at hand are smooth and that a local optimization suffices is justified by the results of the initial calculations, in which a sparse grid search showed that the time to reach is a smooth function, with only one maximum inside the range of target points.

Constraints
Finally, we will discuss the constraints used for the mechanism parameter optimization problem. These can be divided in two parts. First, there are the upper and lower bounds on the parameters themselves. These are shown in Table 5.1. Note that the diameter of the spring connection rods is fixed. The other bounds are chosen as to allow changes that will keep the mechanism of similar size as the prototype, making it suitable for the same size arm.

Secondly there are two other constraints, which will be listed and explained below.

- **Location of outer equilibrium points** One of the ideas behind the design of the Plooi mechanism is the easy of control around the initial and target points obtained by making these points equilibria. The optimization must maintain this property, meaning $T_s(\theta_{\text{initial}} = 0)$

- **Potential energy at equilibrium** The optimization will increase the torques provided by the mechanism in order to make the robot arm accelerate and decelerate faster. Although such optimization will find the fastest mechanism, it is also interesting to see if the mechanism can be made faster without increasing the torques. The optimization was therefore also carried out with a limitation on the torques. This limitation was implemented by constraining the potential energy at the equilibrium to be the same as that of the prototype.
Table 5.1: Upper and lower bound for mechanism parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Lower bound</th>
<th>Upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>r1</td>
<td>spring attachment radius pulley 1 [m]</td>
<td>0.0525</td>
<td>0.1575</td>
</tr>
<tr>
<td>r2</td>
<td>spring attachment radius pulley 2 [m]</td>
<td>0.01</td>
<td>0.03</td>
</tr>
<tr>
<td>a</td>
<td>Pulley transfer ratio []</td>
<td>2.7</td>
<td>8.1</td>
</tr>
<tr>
<td>k</td>
<td>spring constant [N/m]</td>
<td>75</td>
<td>225</td>
</tr>
<tr>
<td>l0</td>
<td>initial length of spring [m]</td>
<td>0.05</td>
<td>0.15</td>
</tr>
<tr>
<td>dl</td>
<td>distance between pulleys [m]</td>
<td>0.05</td>
<td>0.15</td>
</tr>
<tr>
<td>d_r</td>
<td>diameter of spring connection rods [m]</td>
<td>0.005</td>
<td>0.005</td>
</tr>
<tr>
<td>F0</td>
<td>pre tension in spring [N]</td>
<td>3</td>
<td>9</td>
</tr>
</tbody>
</table>

5.2 Results

This section will describe the results of the analysis of the task execution speed. The results will be presented in the order of the subquestions as given at the beginning of this chapter.

Current prototype

To analyse the improvements on the task execution speed caused by the Plooij mechanism we computed the time to reach for a range of target points. Figure 5.1 shows the results of these computations. For the system without spring we see a monotonous increase in time to reach in progressing distance away from the initial point. For the system with spring however, a more complicated graph arises. Note that the maximum time to reach is for a position inside the range. Also, it is clear that the robotic arm is faster with than without the mechanism. The average time to reach, the worst case time to reach and the relative improvements caused by the mechanism are listed in Table 5.2

![Figure 5.1: Minimal time to reach for target range.](image)

The horizontal axis shows the target position of the robot, with both the target range and the target equilibrium point marked. The vertical axis is the minimal time to reach each point in this range of positions, all at 0 angular velocity. This is done for both a standard robotic arm, without mechanism, as for the Plooij mechanism. Note that due to the scale of the axis the robotic the time to reach seems to increase linearly for the standard robotic arm, whereas it is a slightly less than linear increase.

26
Table 5.2: Time to reach values. This table shows the time to reach values for both the prototype spring and a standard robotic arm.

<table>
<thead>
<tr>
<th>Mechanism</th>
<th>average time [s]</th>
<th>average improve [%]</th>
<th>worst case time [s]</th>
<th>worst case improve [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard arm</td>
<td>1.20</td>
<td>-</td>
<td>1.28</td>
<td>-</td>
</tr>
<tr>
<td>Prototype spring</td>
<td>1.07</td>
<td>10.7</td>
<td>1.10</td>
<td>12.7</td>
</tr>
</tbody>
</table>

Effect of maximal torque

To investigate the effect of the maximal torque we have performed the calculation of the average and worst case time to reach for a grid of values of the maximal torque. This grid is made in the interval between 0.53N and 5N. The lower bound is the minimal torque for which the upper bound of the one direction forward reachable set reaches the furthest point in the target range, see section 3.2 for an explanation of this bound. The upper bound is taken sufficiently high to insure that all effects of the mechanism are studied. The results are shown in Figure 5.2. In this figure we see that the maximal improvement is 14.1% at a torque of 0.72Nm, which is close to the minimal required torque. Increasing the torque leads to smaller gains in task execution speed.

![Figure 5.2: The effect of a torque limit on the average and worst case task execution time.](image)

This figure shows the improvement in minimal time to reach caused by using the Plooij mechanism instead of a standard robotic arm for a range of torques. The minimal time to reach is expressed as the average over the range of target points and as the worst case within this range.

To verify if these results hold reasonably well for alterations of the target range, the partial derivatives of the improvement in task execution speed with respect to the target range size was taken. The results are shown in table 5.3, and show that the target range does indeed have only a limited effect on the results. The measure for which it has most effect is the worst case time to reach, at a torque of 0.54. But even for that situation the effect is small: a 10% increase in target size, to 0.22 rad around the target point, decreases the improvement by 4.3%.

Parameter optimization

To study the effects of the parameters of the Plooij mechanism on the task execution speed, these parameters were optimized, as explained in 5.1. This optimization was done both with and without a constraint that set the potential energy at the initial point to that of the
Table 5.3: Effect of target range on improvement. To investigate the effect of the target range size on the improvement in task execution speed, the partial derivatives of the improvement to the size of the target range where taken using finite differences. The table shows the value of this derivative for a torque of 1 Nm and the minimal and maximal values of the derivative in the range (0.64, 5.00) Nm, including the torque values for which these extrema occur.

<table>
<thead>
<tr>
<th>Type</th>
<th>At 1 Nm [%/Nm]</th>
<th>Derivative [%/Nm]</th>
<th>Torque [Nm]</th>
<th>Derivative [%/Nm]</th>
<th>Torque [Nm]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Worst case</td>
<td>24.58</td>
<td>-34.23</td>
<td>0.54</td>
<td>29.37</td>
<td>0.59</td>
</tr>
<tr>
<td>Average</td>
<td>21.07</td>
<td>8.43</td>
<td>5.00</td>
<td>21.92</td>
<td>0.71</td>
</tr>
</tbody>
</table>

Table 5.4: Optimal values for the parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Prototype</th>
<th>Potential constraint</th>
<th>No constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>r1 [m]</td>
<td>0.105</td>
<td>0.118</td>
<td>0.155</td>
</tr>
<tr>
<td>r2 [m]</td>
<td>0.020</td>
<td>0.0219</td>
<td>0.030</td>
</tr>
<tr>
<td>a []</td>
<td>5.40</td>
<td>5.01</td>
<td>4.87</td>
</tr>
<tr>
<td>k [N/m]</td>
<td>150.00</td>
<td>150.09</td>
<td>150.67</td>
</tr>
<tr>
<td>l0 [m]</td>
<td>0.10</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>F0 [N]</td>
<td>6.0</td>
<td>3.0</td>
<td>9.0</td>
</tr>
</tbody>
</table>

| Time to reach [s] | 1.10 | 1.05 | 0.94 |
| Improvement [%]   | 12.7 | 16.7 | 25.4 |

The two optimizations gave the parameters listed in Table 5.4. This table also shows the resulting time to reach and improvement when compared with a standard robotic arm. The optimization without constraint on the potential energy lead to an improvement of 25.4%.

The torque function of the prototype and the two optimized mechanisms are shown in Figure 5.3. The most noticeable feature of the two optimized torque profiles is that the initial point is now a half-stable point, as opposed to the prototype for which the initial point is stable, with an unstable equilibria nearby. The optimization without potential constraint more than doubles the maximal torque. This is done by increasing the radii of the pulleys, while keeping the spring constant nearly the same. The same parameter changes also apply to the optimization with constraint on the potential energy, however the radii are increased less because the constraint limits the maximal torque.

Figure 5.4 shows the time to reach for all points in the target range. The main improvement caused by the optimization with potential constraint is caused by flattening the minimal time to reach curve, and lowering it for positions with a smaller target angle. Further improvement by the optimization without potential constraint is an overall lowering of the minimal time to reach, likely mostly caused by the overall increase in spring torque.

5.3 Conclusion

To investigate the improvements in pick and place task execution speed that can be made by using the Plooij mechanism we have investigated three sub-questions. In this section we will briefly state the answers to those questions.

- How much does the current Plooij Mechanism improve task execution time when compared to a standard robotic arm? An improvement between 10.7 and 12.7% was found, depending
Figure 5.3: The torque profile of the optimized mechanism. This figure shows the spring mechanism torque characteristic for the prototype and the two optimized parametrizations.

Figure 5.4: The time to reach for the optimal mechanism. The minimal time to reach for each position in the target range is shown for both optimized versions of the Plooij mechanism. For comparison the prototype mechanism and the standard robotic arm are also shown.
on whether the average or worst case time to reach was taken as measure for task execution speed.

• What is the effect of the torque limit on this improvement? Improvement in task execution speed of more than 5% was only found for $T_{\text{max}}$ in the range (0.53 Nm, 5.00 Nm), with the maximal improvement being 14.1% at a torque of 0.72 Nm.

• What is the effect of the mechanism’s parameters on this improvement? There is an optimal mechanism, for which the parameters are given in Table 5.4. This optimized mechanism improves worst case task execution speed by 25.4% when compared to a robotic arm without mechanism.

5.4 Recommendations

The results show that the Plooij mechanism is capable of increasing task execution speed, and indicate in what way the mechanism should be altered to obtain the largest improvement. However, there are three main issues that are left for further research.

• The first issue is the simple model used in these computations. Although added complexity is unlikely to change the overall conclusions, it might lead to more insight. Three things to add are Coulomb friction, a motor model and a gearbox model.

• The second issue is the use of a higher dimensional system. The current prototype supports the use of an extra joint. The added dynamics and freedom to choose different paths towards the target range could provide further benefits. It is as of yet unclear how the second dimension is affected by the Plooij mechanism.

• Finally, the improvement in task execution speed is very close to the decrease in energy consumption and the decrease in motor size found in previous research. This hints that the same underlying mechanisms might reward a very large class of optimization goals. Contrast this to the fact that the same mechanism looses versatility in state space, see chapter 4. It seems that to create a versatile robot we must use many degrees of freedom, which allow qualitatively different motions to reach points in state space. For every point in state space one of these motions should be optimized, but the optimization goal that is used to design the mechanism is to a large degree irrelevant.
Chapter 6
Conclusion

This research provided a step in the direction of creating an industrially applicable robotic arm that uses its natural dynamics. This step was taken by a detailed investigation of the Plooij mechanism, a passive mechanism that enhances the natural dynamics of a pick and place robotic arm. The investigation was done to find three issues that were unresolved by previous research on the Plooij mechanism: its range of motion under limited motor torque, its versatility and its capability to increase the speed with which a robotic arm can execute a pick and place task. The main conclusions on these three issues will be summarized here, followed by recommendations for further research.

Range of motion with motor torque limit

To assess how well the Plooij mechanism can be controlled using only a limited motor torque, the one swing reachable set was introduced. This is the set of points that can be reached without back and forth motions. It provides insight in the trade-off between controllability and the maximum allowed motor torque.

- The one swing reachable set forms a narrow tube between the Plooij mechanism’s outside equilibrium points. This tube expands around these equilibrium points.
- The minimal torque that allows picking and placing around the outside equilibrium points is $0.45Nm$.

Versatility

The versatility of the Plooij mechanism was investigated by computing the time limited reachable set. This was done by means of a novel computation method. The time reachable set is the set of points that can be reached given limited time, and allows qualitative comparison of the versatility of different mechanisms. The main conclusion is as follows:

- The Plooij mechanism’s natural dynamics are beneficial compared to a standard robotic arm in only two areas of state space: one around the target equilibrium, and a second at high velocities around a 0 rad position.

Task execution speed

The capability of the Plooij mechanism to increase the pick and place speed of a robotic arm was done using the same computational methods as those used to compute the time limited reachable set. The analysis allowed the pick and place task to be investigations for a variety of tasks, all with the same initial, pick, position, but with the place position varying in a range around the target point of the mechanism. The task execution speed was analysed by finding the
minimal time to reach for all points in the target range. The effects of maximum motor torque and mechanism parameters were also studied. The main results from the task execution speed analysis are:

- The current mechanism can improve the task execution speed by 12.7%
- The maximum motor torque changes this improvement. For small motor torques, (< 0.72), the motor torque is not capable of fully using the natural dynamics. For larger motor torques the improvement decreases, as the acceleration of the robotic arm start to rely more on the motor instead of the mechanism.
- By optimizing the parameters a further improvement was found, to a total of 25.4%

**Recommendations**

In the analysis of each of the three issues, recommendations for further research were discussed. The two most important recommendations are repeated here. First, the analysis should be extended from a one degree of freedom arm to an arm with multiple degrees of freedom. This extension is required for more complex tasks, and allows the robot more freedom to choose a natural dynamic trajectory. The second issue is the generation of a natural dynamic controller. The ultimate goal for such a controller is to be able to adapt its motion trajectory online to the variety of tasks the robotic arm has to perform, making the arm truly applicable in industry.
Appendix A

Derivation of model equations

In this appendix the derivation of the mechanism torque equation is given. This is done by first computing the potential energy stored in the spring, and then taking the derivative to the mechanism angle. To find the potential energy, first the locations of the end positions of the spring must be written down as function of the mechanism angle. In these functions $x_1$ and $y_1$ describe the spring attachment point on the first pulley, while $x_2$ and $y_2$ are on the second pulley. The other parameters are as listed in 2.1.

\begin{align*}
x_1 &= -r_1 \sin(\theta) \quad \text{(A.1)} \\
y_1 &= -r_1 \cos(\theta) \quad \text{(A.2)} \\
x_2 &= r_2 \sin(a\theta) \quad \text{(A.3)} \\
y_2 &= -r_1 - l_0 - r_2 + r_2 \cos(a\theta) \quad \text{(A.4)}
\end{align*}

The length of the spring, $l_s$ is given by:

\begin{equation}
l_s = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} + d \quad \text{(A.5)}
\end{equation}

Giving a potential $E_p$ of:

\begin{equation}
E_p = \frac{1}{2}k(l_s - l_0)^2 + F_0(l_s - l_0) \quad \text{(A.7)}
\end{equation}

Substituting the previous equations using Matlab, we obtain:

\begin{align*}
E_p &= F_0 \cdot (d - l_0 + \sqrt{(l_0 + r_1 + r_2 - r_2 \cos(a\theta) - r_1 \cos(\theta))^2 + (r_2 \sin(a\theta) + r_1 \sin(\theta))^2}) + \\
&\frac{1}{2}(k(d - l_0 + \sqrt{(l_0 + r_1 + r_2 - r_2 \cos(a\theta) - r_1 \cos(\theta))^2 + (r_2 \sin(a\theta) + r_1 \sin(\theta))^2})^2) \quad \text{(A.8)}
\end{align*}

Then because the mechanism torque $T_s$ is the derivative of the potential energy to $\theta$ we get:

\begin{align*}
T_s &= -\frac{(F_0 + dk - kl_0 + k\sqrt{(l_0 + r_1 + r_2 - r_2 \cos(a\theta) - r_1 \cos(\theta))^2 + (r_2 \sin(a\theta) + r_1 \sin(\theta))^2})(r_1 \sin(\theta))}{\sqrt{(l_0 + r_1 + r_2 - r_2 \cos(a\theta) - r_1 \cos(\theta))^2 + (r_2 \sin(a\theta) + r_1 \sin(\theta))^2}} + \\
&\frac{l_0 r_1 \sin(\theta) + r_1 r_2 \sin(\theta) + ar_2^2 \sin(a\theta) + r_1 r_2 \sin(\theta(a - 1)) + al_0 r_2 \sin(a\theta) + ar_1 r_2 \sin(a\theta) - ar_1 r_2 \sin(\theta(a - 1))}{\sqrt{(l_0 + r_1 + r_2 - r_2 \cos(a\theta) - r_1 \cos(\theta))^2 + (r_2 \sin(a\theta) + r_1 \sin(\theta))^2}} \quad \text{(A.10)}
\end{align*}
Bibliography


