

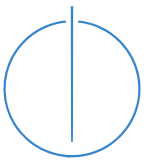
TUM SCHOOL OF COMPUTATION,
INFORMATION AND TECHNOLOGY

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Robotics, Cognition, Intelligence

Ultra-Efficient Resonance-Driven SCARA Robot

Theresa Gräbner





TUM SCHOOL OF COMPUTATION,
INFORMATION AND TECHNOLOGY

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Robotics, Cognition, Intelligence

Ultra-Efficient Resonance-Driven SCARA Robot

Hocheffizienter resonanzgetriebener SCARA-Roboter

Author:	Theresa Gräbner
Supervisor:	Prof. Dr. Alin Albu-Schäffer
Advisors:	Arne Sachtler, Prof. Dr. Manuel Keppler
Submission Date:	15 November 2026



Abstract

Pick-and-place is a common application of industrial robots, which consume substantial energy during acceleration and deceleration. Elastic elements can store kinetic as potential energy and return it, so that the robot oscillates at resonance between the pick and place positions while its motors only compensate for non-conservative forces. Since the robot has to perform different pick and place tasks consecutively, it should be able to adapt the properties of its elastic elements to oscillate freely between the varying pick and place positions. This thesis thus investigates the workspace regions in which a SCARA robot with clutched parallel elastic actuators can execute quasi-conservative pick-and-place sequences. When there are several suitable sequences for the same pick-and-place task set, the shortest sequence duration is considered. To this end, we simulate the robot as a conservative system, where the clutches engage the springs independently at selected times and thereby shift the spring equilibria. Around shared equilibria, which allow transitioning between pick-and-place motions through a torque pulse, we search for point-symmetric oscillations in joint space between Cartesian pick and place positions mirrored about the x-axis. As an example, a 3×3 task set of such x-axis mirrored position pairs spaced 6 cm apart can be placed and executed with quasi-conservative sequences anywhere in the workspace. For the considered planar 2-DoF SCARA robot with a maximum reach of 1 m, 3×3 sequence durations vary between 5 and 30 s. Most mirrored position pairs across the workspace offer several oscillations around the same equilibrium between the same joint configurations for other task sets to choose from, with 6.37 on average and up to 49. The results suggest that clutched parallel elastic actuators enable a SCARA robot to execute quasi-conservative sequences for various pick-and-place task sets and thus have great potential for energy-efficient manufacturing.

Acknowledgements

My heartfelt thanks go to my family, my boyfriend Markus Brunner and his family as well as my friends for their support. My boyfriend's family in particular helped me out by regularly lending me their car for the duration of this thesis.

On the professional side, I am grateful to the German Aerospace Center (DLR) for the opportunity to work on such an interesting topic in my master's thesis and to my advisors, who supported me along the way.

Munich — Theresa Gräbner

15 November 2026

Deutsches Zentrum
für Luft- und Raumfahrt
German Aerospace Center

Contents

1. Introduction **1**
- 1.1. Problem Statement and Research Question . . . 2
- 1.2. Contribution and Outline . . . 2

2. Related Work **4**
- 2.1. Energy Optimization Strategies for Industrial Robots . . . 4
- 2.2. Exploiting Natural Motion with Elastic Elements . . . 5
- 2.3. Clutched Elastic Actuators for Energy-Efficient Motion . . . 8
- 2.4. Research Gap . . . 8

3. Fundamentals **10**
- 3.1. Manipulator Kinematics . . . 10
- 3.2. Manipulator Dynamics . . . 13
- 3.3. Numerical Integration with Runge–Kutta 4 . . . 16
- 3.4. Covariance Matrix Adaptation Evolution Strategy . . . 17
- 3.5. Travelling Salesman Problem . . . 18

4. Approach and Concept **20**
- 4.1. Robot Model . . . 21
- 4.2. Role Division between Spring and Motor . . . 23
- 4.3. Progressive Problem Refinement for Single Tasks . . . 24
- 4.4. Conservative Pick-and-Place Sequence . . . 33

5. Implementation **37**
- 5.1. MuJoCo Modelling . . . 38
- 5.2. No-Spring Optimization . . . 43
- 5.3. Stiffness Optimization . . . 47
- 5.4. Clutch Switching Times Optimization . . . 56
- 5.5. Pick-and-Place Sequence Optimization . . . 62

6. Evaluation **64**
- 6.1. Workspace Coverage for Single Tasks . . . 64
- 6.2. Closed Conservative 3×3 Pick-and-Place Sequence . . . 75

7. Discussion **83**
- 7.1. Search Space . . . 83
- 7.2. Pick-and-Place Task Scope . . . 85
- 7.3. Transfer to a Realistic System . . . 86

8. Conclusion **88**

Bibliography **90**

List of Figures **97**

A. Appentix **I**
A.1. MJCF Model I
A.2. Radially Aligned Placement III

1 Introduction

In 2024, 4,281,585 industrial robot units operated worldwide. Their installations increased by 10% from 2023, suggesting a rapid increase in usage[46] and long-term growth[71]. However, large-scale robotization also increases energy consumption[24, 72], while the EU directive sets the target of consuming at least 11.7% less energy in 2030 than the 2020 reference projections[26, Art. 4]. As the industrial sector made up 23.9% of the EU's energy consumption in 2024, with electricity covering 33.3% of its demand[29], energy efficiency has become one of the central priorities and trends for industrial robots to reduce carbon emissions[72, 47, 24]. Across industries such as automotive, electronics, food, pharmaceuticals, cosmetics and machinery, pick-and-place is one of the elemental robot applications[50, 2, 67, 6].

In such tasks, robots grab objects in an initial pose and release them in a target pose[2, 50]. Several robotic tasks consist of a set of different pick-and-place tasks like loading and unloading quickly moving objects on belt conveyors, sorting, packing, changing machine tools, simple assembly tasks and palletizing[2, 50]. Parallel robots such as the Delta robot and the five-bar mechanism as well as the serial Selective Compliance Assembly Robot Arm (SCARA) are the most common robots performing pick-and-place tasks[67, 6, 43]. Although SCARA robots can also move along the vertical z-axis[15, 16], they are mostly built for repetitive, high-throughput tasks in the horizontal x-y plane[72]. This thesis therefore only considers pick-and-place motion in the horizontal x-y plane between the pick and place positions, for which a planar robot with two rotational joints (2-DoF) suffices. This has the advantage that gravity does not act in the direction of motion, simplifying modelling and simulation.

Since pick-and-place is a highly repetitive, high-precision and fast task[67, 6] for which energy consumption has rarely been a design priority[19, 72], it has high potential for reducing energy consumption in industrial sectors[72]. By accelerating and decelerating in fast repetitive motions, robots dissipate a lot of energy[4, 16]. Through the serial kinematic design of the SCARA robot, inertial effects accumulate and the robot thus requires higher torque, leading to increased energy consumption[72]. Ideally, the robot would therefore keep the kinetic energy of its links in its system instead of supplying and dissipating it anew in every repetition of the pick-and-place motions, so that its motors only compensate for the energy that friction and other non-conservative forces dissipate in a real system[6, 15, 84, 72]. Such a motion is conservative in a conservative system and is thus considered a quasi-conservative pick-and-place motion. The prospect of exploiting these quasi-conservative motions with a SCARA robot has the potential to make robot-based manufacturing significantly more sustainable[16].

Elastic elements enable free quasi-conservative oscillations between the pick and place position by storing the kinetic energy as potential energy while the robot's links decelerate and return the kinetic energy into the links while they accelerate[16, 5, 75]. Several works have already explored this principle of exploiting the free oscillation[84, 72]. Instead of relying on a pure free

oscillation, Vivas et al. [89] optimized the pick-and-place trajectory together with the stiffness of the elastic elements for a specific pick-and-place task of a planar serial 2-DoF robot, so that the motion exploits as much of the free oscillation as possible. Compared with rigid actuation, this reduced the energy consumption by up to 62%[89]. As in this case, most works tune their elastic elements to a single pick-and-place task[84], the so-called nominal pick-and-place task[64].

1.1. Problem Statement and Research Question

In practice, a robot rarely performs only its nominal task but has to perform different pick and place tasks[64, 7, 2] consecutively. With an elastic element designed for the nominal task, the free oscillation might not come to rest at pick and place positions other than those of the nominal task[64].

Many works, some of them beyond pick-and-place tasks, address such non-nominal tasks through optimized trajectories, shifted spring equilibria, variable stiffness actuators or a designed potential field theoretically realizable by nonlinear springs (Section 2.2.2). To the best of our knowledge, however, most works considering pick-and-place tasks supplement the free oscillation with actuator effort for non-nominal tasks to adapt the trajectory to stop in the non-nominal pick and place positions. How to modify the system to achieve quasi-conservative free oscillations between varying pick and place positions across the workspace has hardly been investigated.

Besides considering different pick-and-place tasks across the workspace, this thesis also focuses on combining these free oscillations into sequences. This way, the robot executes different pick-and-place tasks consecutively with quasi-conservative motions.

Since clutched parallel elastic actuators (CPEAs) can shift the equilibrium of their fixed-stiffness springs[77] by connecting and disconnecting the springs to the robot when needed, such actuators might give enough degrees of freedom to adapt the free quasi-conservative oscillation of the robot to execute quasi-conservative pick-and-place sequences. This leads to the following research question:

Research Question. *In which workspace regions can a SCARA robot with clutched parallel elastic actuators execute quasi-conservative pick-and-place sequences?*

Research Sub-Question. *If there are several suitable sequences for the same pick-and-place task set, what is the shortest sequence duration?*

1.2. Contribution and Outline

The core contribution of this thesis is a concept for quasi-conservative pick-and-place sequences, in which the clutches shape the free oscillations of different pick-and-place tasks such that they intersect at an equilibrium of the springs, where a short torque pulse of the motors would perform the transition to the next pick-and-place task. To find free oscillations between positions across the workspace, we develop an optimization pipeline that simulates the robot as a conservative system and collects several point-symmetric oscillations in joint space between pick and place positions mirrored about the x-axis. As an example of a quasi-conservative pick-and-place sequence, we consider a 3×3 grid of pick-and-place tasks whose pick and place positions are mirrored about the x-axis and analyse in which workspace regions it can be performed with quasi-conservative pick-and-place sequences.

In Chapter 2, the thesis first gives an overview of related work on energy-efficient robots. It starts with general strategies for the energy efficiency of industrial robots (Section 2.1) and then turns to works that exploit the robot's natural motion (Section 2.2). After a review of clutched elastic actuators for energy-efficient motion (Section 2.3), Section 2.4 points out the research gap that our work aims to close. Chapter 3 introduces the fundamentals that are essential for understanding this thesis, while Chapter 4 describes our concepts in detail. In particular, Section 4.3.2 shows and derives how our optimization can obtain point-symmetric oscillations in joint space and why such oscillations are advantageous. Chapter 5 then explains how our concepts are implemented. Afterwards, Chapter 6 presents the results and Chapter 7 answers the research question and discusses our work. Finally, Chapter 8 summarizes this thesis and gives an outlook on future research.

2
Related Work

To illustrate the research trend towards energy-efficient robots in more detail, Section 2.1 begins with a general overview of energy optimization strategies for industrial robots. Section 2.2 then defines natural motion and describes how the literature exploits a robot's dynamics to execute tasks energy-efficiently. As this work searches for quasi-conservative pick-and-place sequences with clutched parallel elastic actuators, Section 2.3 reviews how these clutched elastic actuators have been used for energy-efficient motions in robots so far. Based on Sections 2.2 and 2.3, Section 2.4 identifies the research gap that this thesis aims to close.

2.1. Energy Optimization Strategies for Industrial Robots

Muru and Rassõlkin [72] and Carabin, Wehrle, and Vidoni [19] divided the different strategies to increase energy efficiency into three main categories, namely purely software-based approaches, purely hardware-based approaches and hybrid approaches, which modify hardware as well as optimize software.

2.1.1. Software-Based Approaches

According to Muru and Rassõlkin [72] and Carabin, Wehrle, and Vidoni [19], the software-based approaches can again be categorized into task or operation scheduling, parameter optimization and trajectory optimization. In task scheduling, works optimize the order of the tasks for higher energy efficiency, which makes it a practical strategy for multi-robot production systems. Parameter optimization, in contrast, focuses on finding the most energy-efficient motion parameters such as speed, acceleration and jerk while considering mechanical influences such as payload and drive lubricant temperature [19, 72]. Works on trajectory optimization look into energy-efficient motions while considering task constraints, dynamic feasibility and limitations [72]. Here, the literature distinguishes between point-to-point polynomials [18] and multi-point trajectories, in which the points are connected by piecewise functions [11]. Wiriasto et al. [91] identified kinematic and dynamic analysis as the primary foundation for energy-conscious trajectories. According to them, machine learning and reinforcement learning as well as numerical and heuristic methods were used as complementary layers on top of this foundation. Software-based methods have improved the energy efficiency of pick-and-place tasks [87] and tasks composed of them. Palletizing [59], for example, consists of a series of pick-and-place tasks with the same pick position and varying place positions.

2.1.2. Hardware-Based Approaches

Hardware-based approaches redesign and optimize the robot's components and structure to better suit the robot's task [19]. Some works focus on making the moving components lighter so that the actuators need less effort and therefore less energy to move them [72]. Others focus on efficient robot reconfiguration, in which components are rearranged and changed to adapt to new circumstances [14]. Following Meike and Ribickis [62] and Garcia, Bittencourt, and Villani [34], there are also strategies that reduce the energy typically lost during idle phases by optimizing the energy usage in standby modes. In addition, regenerative drives reclaim the energy of braking phases so that it can be reused [62, 34]. Meike and Ribickis [63] designed a DC-bus architecture shared between robots that enables energy to flow between them when one of them decelerates. Energy can also be recovered and reused with elastic elements, which convert kinetic energy into elastic energy and back [72, 19].

2.1.3. Hybrid Approaches

Hybrid approaches combine hardware-based and software-based strategies[19, 72]. Some works add functional redundancy to the hardware so that it can later be exploited by software-based methods [72]. Others optimize hardware parameters together with the task and the motion[19, 72]. Fabris, Scalera, and Gasparetto [31] optimized the lower arm lengths, the task placement and the execution time of a pick-and-place task on a 3-DoF parallel robot. Combining elastic elements with the task motion, Bruzzone, Verotti, and Fanghella [15] investigated the vertical motion during pick-and-place tasks on an RR-4R-R SCARA-like robot. They compared exact balancing, which compensates for gravity in every static configuration, with natural balancing, which compensates for gravity only in the spring's equilibrium and uses the system's own dynamics for movements. Gravity balancing is only beneficial for slow motions, since the inertial effects dominate at higher speeds [5]. Natural balancing is thus more promising for fast tasks. Many other works use elastic elements and the system's dynamics in a similar way (Section 2.2).

2.2. Exploiting Natural Motion with Elastic Elements

Following [6, 15, 84, 72], motions that result from a system's own dynamics are generally known as natural motion. Elastic elements change these dynamics and thus the natural motion of a robot. Integrating them into the hardware and exploiting the resulting natural motion is a very promising strategy to reduce energy consumption in repeatedly executed tasks. When excited at its resonance, the perturbed system converts its kinetic energy into elastic energy while decelerating and stores this energy in its elastic element. The elastic element later releases this energy as kinetic energy back into the robot's links, which moves the system back to the equilibrium of the elastic element. The robot thus freely oscillates around this equilibrium. In the ideal case of a conservative system without damping, friction and other non-conservative forces such as disturbances, the actuator only has to provide the energy to perturb the system. In a more realistic scenario, the free oscillations decay over time. Instead of accelerating and decelerating the system, the actuator then only has to counteract the effect of the non-conservative forces to keep the system oscillating [6, 15, 84, 72]. This quasi-conservative free oscillation is the natural motion of the system with elastic elements. These free oscillations can also be characterized with nonlinear modal theory, which provides a way for controllers to effectively excite and sustain the oscillations [9, 25].

Several papers exploited the natural motion to improve energy efficiency for periodic motions

such as pick-and-place motions [6, 15, 84, 72] by tuning the properties of the elastic element, such as its equilibrium position and stiffness, to the desired motion characteristics and task. Scalera et al. [84] divided such methods into three main categories. The first category comprises papers that optimize the spring parameters to minimize the energy consumption along a predefined, mostly harmonic trajectory and exploit the natural motion of the system as well as possible. According to Palomba et al. [75], this method is beneficial when obstacle avoidance is necessary and the trajectory cannot be constrained to the system's dynamics. The second category comprises papers that optimize the trajectory concurrently with the parameters of the elastic element, using optimal control theory or a parametric representation of both. Since the elastic element and the trajectory are not independent of each other, the optimization becomes quite complex[84]. In contrast to the first two categories, papers using the free oscillation (free-vibration response) do not prescribe the shape of the trajectory, which is instead fully determined by the dynamics of the system. They only optimize the elastic element's properties such that the trajectory fulfils the task requirements [84]. This is done with simulators that mostly neglect dissipative forces, since the actuators compensate for them [67].

While this classification is referred to when discussing individual works, Sections 2.2.1 and 2.2.2 instead distinguish them by how the elastic element is mechanically integrated with the actuator and how they handle tasks for which the elastic element was not initially designed.

2.2.1. Task-Specific Elastic Elements

There are in general two ways in which the elastic element can be arranged, namely in series with or in parallel to the actuator. In series elastic actuators (SEAs), the spring is placed between the actuator and the link [84]. Both Kordik, Gattringer, and A. Müller [53] and Vivas et al. [89] used this actuator to exploit the natural motion with an optimized trajectory, simultaneously determining the trajectory and the elastic properties, namely stiffness and preload, for a specific pick-and-place task on a 2-DoF parallel planar robot and a 2-DoF serial planar robot, respectively. In contrast to SEAs, the spring in parallel elastic actuators (PEAs) is placed in parallel to the actuator and thus allows the required torque to be distributed between the spring and the actuator. This property makes PEAs attractive for energy-efficient motions [84, 4]. Palomba et al. [75] used both regenerative drives and PEAs to reduce the energy consumption of a parallel five-bar mechanism and a serial SCARA robot along a specific predefined trajectory for a pick-and-place task. The five-bar mechanism was also used by Mirz et al. [64], who concurrently optimized the trajectory of the pick-and-place task and the properties of the elastic elements with optimal control. Unlike these two works, which prescribe or optimize the trajectory, Barreto, Schöler, and Corves [6], Mora, Barreto, and Rodriguez [65] and Barreto and Corves [7] used the free oscillation to determine the spring's properties, such as stiffness and equilibrium position, for a specific pick-and-place task on the parallel five-bar mechanism and the Delta robot.

2.2.2. Adapting Elastic Elements to Varying Tasks

The specifications of pick-and-place tasks, such as start and target positions, cycle time and payload, often vary [64, 7]. Since the elastic element has been tuned to the task-specific dynamics [72] of the nominal task and the characteristics of regular springs cannot be adjusted [77], the trajectory required for a non-nominal task might not match the natural motion of the system as closely as intended for the nominal task [64]. The fixed stiffness restricts the actuator's ability to adapt its compliance to different tasks [5]. Since the equilibrium position of conventional PEAs is also fixed, the robot motions have to be planned around it to exploit the natural

motion effectively, which limits the flexibility in using the robot workspace [16]. Non-nominal tasks can thus be less energy-efficient than the nominal task [65]. This is usually addressed by optimizing the trajectory of non-nominal tasks for energy efficiency with purely software-based approaches [64, 7] and by varying the cycle time of the pick-and-place task instead of keeping it fixed to the task requirements [65].

Besides software-based approaches, the elastic element itself can be adapted to varying tasks through a nonlinear spring characteristic, a variable stiffness or an adjustable equilibrium position.

Nonlinear Elastic Elements

Sachtler and Albu-Schäffer [81] designed a potential field, in principle realizable by nonlinear elastic elements, such that every joint configuration of a 2-DoF serial manipulator within a predefined region lies on a strict normal mode passing through the common equilibrium. The system can thus freely oscillate along a variety of modes. These oscillations consist of rest-to-rest motions between two rest positions [81] and thus correspond to pick-and-place motions. However, only one of the two rest positions can be chosen within the predefined region[81], while the other position follows from it and the system's dynamics. These modes could therefore theoretically fulfil pick-and-place tasks between every joint configuration within the predefined region and the corresponding rest configuration that the system's dynamics determine on the same mode.

Variable Stiffness

Variable stiffness actuators have a spring whose stiffness can be controlled, for example by an additional actuator [49]. In the actuator of Jafari, Tsagarakis, and Caldwell [49], the variable stiffness spring acts as a series spring between the actuator and the link. They used a predefined trajectory of a 1-DoF arm to define the elastic element's properties and to tune the stiffness continuously when the frequency varies along the reference trajectory. According to Balderas Hill et al. [5], such an arrangement of the variable stiffness spring leads to uncontrolled deflections, which worsen the accuracy of the motion. In contrast, Goya et al. [37] used a variable stiffness spring in parallel to the actuator. They determined the stiffness based on a predefined, harmonic trajectory for pick-and-place tasks on a SCARA robot, with the equilibrium position in the middle between the start and end points.

Equilibrium Adjustment

Another way to adapt to non-nominal tasks is to adjust the equilibrium of the elastic element [16]. Hill et al. [43], Hill et al. [44], Balderas Hill et al. [4] and Balderas Hill et al. [5] improved the estimated energy efficiency of pick-and-place tasks on parallel robots such as the five-bar mechanism and the Delta robot. They mostly optimized the nominal elastic properties concurrently with the trajectory or used a predefined trajectory. To adjust to different tasks, they modified the springs' attachment points with an additional actuator, effectively changing the equilibrium of the parallel springs.

Mora, Rodriguez, and Corves [66] also used an extended parallel five-bar mechanism for pick-and-place tasks, with springs parallel to the actuators whose equilibria can be changed. The nominal spring parameters were calculated for a free oscillation between the nominal pick and place positions with the method of Mora, Barreto, and Rodriguez [65]. For non-nominal tasks, the equilibrium positions were changed while the robot followed a predefined trajectory [66].

Chen et al. [20] proposed a new mechanical design that does not shift the equilibrium of the

parallel spring with an additional motor, which would require extra motor effort for preloading the spring. Instead, they adjusted the equilibrium of the parallel spring similarly to SEAs and used a 3D-printed spring. They tested it with a passive prosthetic knee, where the adjustable-equilibrium parallel elastic actuator is in the hip joint and oscillates during walking. With this equilibrium adjustment, they were able to adapt to different tasks such as walking on inclined terrain and to different loads on the hip joint.

2.3. Clutched Elastic Actuators for Energy-Efficient Motion

Besides the mechanical solutions mentioned above for adapting to non-nominal tasks, one can also use clutched elastic actuators (CEAs), which have the advantage that the timing of the energy storage and release can be controlled and that the spring can be disengaged when it would exert unwanted torque on the joint [77]. CEAs are mostly used in exoskeletons, prostheses, robotic legs and arms to reduce energy consumption and to support jumping and walking motions [79]. Penzlin et al. [76] used a CEA in the hip joint for gait motions and were able to reduce the energy demand and peak loads. Wang et al. [90], in contrast, placed a CEA in the knee joint of their exoskeleton. Its spring stores energy during flexion and releases it during extension in the stance phase of walking, thus reducing the muscle activity of four participants. During the swing phase of walking, they disengage the clutch to preserve the natural human gait. Similarly, works on 1-DoF, 2-DoF or 3-DoF legged robots engage the CEA of the knee or ankle joint in the stance phase to reduce energy consumption and to preload the spring with high torque for a rapid energy release in jumping, hopping, walking or running motions, while keeping the CEA disengaged during the flight phase [38, 58, 57, 27]. Some papers even introduce CEAs with several springs, which provide further degrees of freedom for adjusting to a task [54]. In general, CEAs allow the equilibrium to be adjusted during operation [77]. Plooij, Wisse, and Vallery [78] used this to adapt the dynamics of a 1-DoF robot so that it oscillates between varying rest positions. They designed a CEA that keeps the spring deflected when the clutch disengages, which enables dwell times at the rest positions.

2.4. Research Gap

Table 2.1 lists by platform the reviewed works that use elastic elements to increase energy efficiency and indicates with ✓, (✓) or × whether a work meets each criterion fully, partly or not at all. The column “Multi-DoF” counts passive as well as active actuators as DoF.

As described in Sections 2.2.2 and 2.3, the works in the locomotion group consider hopping, jumping or walking motions rather than pick-and-place tasks. Many of the reviewed works address parallel robots for pick-and-place tasks, but hardly any work considers the free oscillation fulfilling non-nominal pick-and-place tasks. Some works use free oscillations only for their nominal task and adapt to varying tasks by other means (Section 2.2.2).

Furthermore, to the best of our knowledge, no work has yet used a CPEA for pick-and-place tasks on a robot with more than one degree of freedom. Among the works with a CPEA, only the single-link robot arm of Plooij, Wisse, and Vallery [78] performs rest-to-rest motions such as pick-and-place motions. When considering planar serial robots, to which the considered SCARA robot belongs, the work of Sachtler and Albu-Schäffer [81] is most closely related to this thesis. As explained in Section 2.2.2, however, their potential field allows only one of the two rest positions to be chosen, while the other one follows from the dynamics.

Platform	Source	Multi-DoF	Elastic Element		Considered Task		Free Oscillation		Adaptation to Varying Tasks
			Parallel to Actuator	CPEA	Cyclic	Pick-and-Place	Nominal Task	Non-nominal Task	
Locomotion	[27]	✓	✓	✓	✓	×	×	×	✓
	[57, 58]	✓	✓	✓	✓	×	(✓)	(✓)	✓
	[20]	✓	✓	×	✓	×	(✓)	(✓)	✓
	[38, 76]	×	✓	✓	✓	×	×	×	×
	[90]	×	✓	✓	✓	×	(✓)	(✓)	✓
Parallel Robot	[4, 5, 43, 44]	✓	✓	×	✓	✓	×	×	✓
	[66, 67]	✓	✓	×	✓	✓	✓	×	✓
	[53]	✓	×	×	✓	✓	×	×	×
	[6]	✓	✓	×	✓	✓	✓	×	×
	[7, 65]	✓	✓	×	✓	✓	✓	×	✓
	[64]	✓	✓	×	✓	✓	×	×	✓
	[75]	✓	✓	×	✓	✓	×	×	×
Single-Link Robot	[49]	×	×	×	×	×	×	×	(✓)
	[54]	×	✓	✓	✓	×	×	×	✓
	[78]	×	✓	✓	✓	(✓)	✓	✓	✓
Planar Serial Robot	[37]	✓	✓	×	✓	✓	×	×	✓
	[89]	✓	×	×	✓	✓	×	×	×
	[75]	✓	✓	×	✓	✓	×	×	×
	[81]	✓	✓	×	✓	(✓)	✓	✓	✓
	This thesis	✓	✓	✓	✓	✓	✓	✓	✓

Table 2.1.: Classification of related work using elastic elements for energy-efficient motion by platform.

This thesis considers the free oscillations of a 2-DoF planar SCARA robot with CPEAs in its joints for quasi-conservative pick-and-place sequences across its workspace and thus addresses the research gap.

3
Fundamentals

This chapter summarizes the fundamentals needed to understand this thesis, such as the derivation of conditions for point-symmetric oscillations in Section 4.3.2. Section 3.1 first considers the kinematics of our planar 2-DoF SCARA robot and Section 3.2 presents its dynamics. Section 3.3 then introduces the numerical integration method that the simulation in Section 5.1 uses. Section 3.4 briefly explains the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), which the stiffness optimization in Section 5.3 applies. Finally, Section 3.5 describes the generalized travelling salesman problem, which later determines the order of the task transitions in Section 4.4.

3.1. Manipulator Kinematics

In this work, we consider a planar 2-DoF robot, whose two revolute joints have the joint angles q_1 and q_2. Figure 3.1 shows a schematic drawing of the robot, in which the links are displayed as lines with the lengths l_1 and l_2.

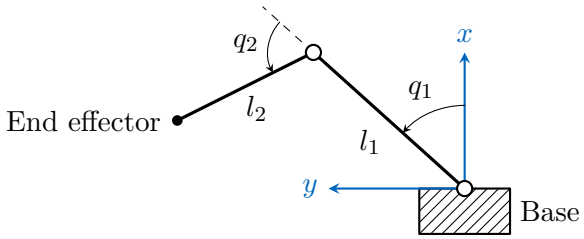


Figure 3.1.: Planar robot with two revolute joints (AI-generated illustration)

Following [60, chapter 2], the configuration of the robot is fully described by its two joint angles, which we collect in the joint vector $\boldsymbol{q} = (q_1, q_2)^\top$. Since a revolute joint returns to the same pose after a full revolution, each joint angle lies on a circle S^1. The configuration space of the robot is therefore not the plane $\mathbb{R}^2$ but the surface of a torus $T^2 = S^1 \times S^1$. Both are two-dimensional, but the plane extends infinitely, while the torus wraps around in both coordinates. A common choice of coordinates on T^2 is $[0, 2\pi) \times [0, 2\pi)$[60, chapter 2]. Equivalently, the joint angles can be given in $[-\pi, \pi) \times [-\pi, \pi)$, which is the convention used in this work.

3.1.1. Forward Kinematics

For a serial manipulator the forward kinematics map the joint variables to the pose of the end effector[85, Section 2.8]. This pose combines the position and the orientation of a frame attached to the end effector, both given relative to the base frame[85, Section 2.10]. During a motion, the

joint variables change over time and the end-effector position follows as $\boldsymbol{x}_{cart}(t) = f_{kin}(\boldsymbol{q}(t))$[60, Chapter 5].

Following [60, Chapter 6], we only consider the position $\boldsymbol{x}_{cart} = (p_x, p_y)^\top$ of the end effector in the plane for the planar 2-DoF robot. The first joint angle q_1 is measured from the x-axis of the base frame and the second joint angle q_2 relative to the first link, so that $\boldsymbol{q} = (0, 0)$ stretches the arm along the x-axis. Both joint angles are positive in the direction that rotates the x-axis towards the y-axis as shown in Figure 3.1. With the link lengths l_1 and l_2 the forward kinematics are (3.1).

$$\boldsymbol{x}_{cart} = f_{kin}(\boldsymbol{q}) = \begin{bmatrix} l_1 \cos q_1 + l_2 \cos(q_1 + q_2) \\ l_1 \sin q_1 + l_2 \sin(q_1 + q_2) \end{bmatrix} \tag{3.1}$$

Each set of joint angles $\boldsymbol{q}$ yields exactly one end-effector position[60, Chapter 6].

3.1.2. Inverse Kinematics

Following [60, Section 6], the inverse kinematics determine the joint angles $\boldsymbol{q}$ for a given end-effector position $\boldsymbol{x}_{cart}$. For the planar 2-DoF robot they can be solved geometrically with the two-argument arctangent $\operatorname{atan2}(p_y, p_x)$. It gives the angle between the x-axis and the line from the origin to the point (p_x, p_y). The law of cosines in the triangle spanned by the two links and the line from the origin to $\boldsymbol{x}_{cart}$ yields the auxiliary angles in (3.2).

$$\beta = \arccos\left(\frac{l_1^2 + l_2^2 - p_x^2 - p_y^2}{2 l_1 l_2}\right), \qquad \varphi = \arccos\left(\frac{p_x^2 + p_y^2 + l_1^2 - l_2^2}{2 l_1 \sqrt{p_x^2 + p_y^2}}\right), \qquad \gamma = \operatorname{atan2}(p_y, p_x) \tag{3.2}$$

With these angles the inverse kinematics yield the righty and the lefty solutions in (3.3).

$$q_1 = \gamma - \varphi, \quad q_2 = \pi - \beta \qquad \text{and} \qquad q_1 = \gamma + \varphi, \quad q_2 = \beta - \pi \tag{3.3}$$

Both solutions differ in the sign of the elbow angle q_2[60, Section 6].

Following [60, Chapter 6], all positions that the end effector can reach according to (3.1) form the workspace of the robot. For $l_1 \neq l_2$ it is the annular region between the circles of radius $|l_1 - l_2|$ and $l_1 + l_2$ around the base. Positions outside this region cannot be reached, positions on its two boundary circles are reached by exactly one joint configuration and positions in its interior by two[60, Chapter 6].

Following [23, Section 4.2], for $l_1 = l_2$ the inner radius vanishes and the workspace becomes a disc of radius $2l_1$. The positions at which the end effector can take every orientation form the dextrous workspace, which in this case consists only of the origin[23, Section 4.2]. Each orientation of the end effector in the origin belongs to a different joint configuration, so the origin is reached by infinitely many joint configurations.

3.1.3. Singularity

Following [60, Section 5.3], differentiating $\boldsymbol{x}_{cart}(t) = f_{kin}(\boldsymbol{q}(t))$ with respect to time yields $\dot{\boldsymbol{x}}_{cart} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}}$, where the Jacobian $\boldsymbol{J}(\boldsymbol{q}) = \partial f_{kin}(\boldsymbol{q}) / \partial \boldsymbol{q}$ maps the joint velocities to the end-effector velocity. With the Jacobian one can determine the configurations in which the end effector cannot move instantaneously in at least one direction. These configurations are called singularities[60, Section 5.3]. In such a configuration the Jacobian loses rank[85, Section 3.3].

For the planar 2-DoF robot, the Jacobian and its determinant are (3.4)[60, Chapter 5, 85, Section 3.3].

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{bmatrix} -l_1 \sin q_1 - l_2 \sin(q_1+q_2) & -l_2 \sin(q_1+q_2) \\ l_1 \cos q_1 + l_2 \cos(q_1+q_2) & l_2 \cos(q_1+q_2) \end{bmatrix}, \qquad \det \boldsymbol{J}(\boldsymbol{q}) = l_1 l_2 \sin q_2 \tag{3.4}$$

Following [85, Section 3.3], the determinant of the Jacobian vanishes exactly for (3.5),

$$q_2 = n_s \pi\,, \quad n_s \in \mathbb{Z} \tag{3.5}$$

which are therefore the singular configurations of the robot. For even n_s the arm is fully extended and the end effector lies on the outer boundary of the reachable workspace with $\|\boldsymbol{x}_{cart}\| = l_1 + l_2$. For odd n_s the arm is fully folded back and the end effector lies on the inner boundary with $\|\boldsymbol{x}_{cart}\| = |l_1 - l_2|$. They are therefore referred to as boundary singularities[85, Section 3.3]. In the unique representation within $[-\pi, \pi)$ these singularities are at $q_2 = 0$ and $q_2 = -\pi$.

On the inner and outer boundary the righty and the lefty inverse kinematics solutions coincide, since q_2 and $-q_2$ differ only by a multiple of 2π[60, Chapter 6]. A point on these boundaries therefore has only one inverse kinematics solution[60, Chapter 6]. The only exception is the inner boundary for $l_1 = l_2$. It consists only of the origin, which is reached by infinitely many configurations[23, Section 4.2] as discussed in Section 3.1.2.

3.1.4. From Cartesian to Joint-Space Tolerance

The maximum gain $\sigma_{\max}(\boldsymbol{q})$ of the mapping from joint to Cartesian displacements[85, Section 3.9, 36, Section 2.3, 2.4] bounds the Cartesian deviation $\Delta\boldsymbol{x}_{cart}$ caused by a joint-space deviation $\Delta\boldsymbol{q}$ in the configuration $\boldsymbol{q}$ by (3.6).

$$\|\Delta\boldsymbol{x}_{cart}\|_2 \leqslant \sigma_{\max}(\boldsymbol{q})\, \|\Delta\boldsymbol{q}\|_2 \tag{3.6}$$

Since $\sigma_{\max}(\boldsymbol{q})$ depends on the configuration, we replace it by its worst-case value over the workspace with (3.7).

$$\sigma_{\text{wc}} = \max_{\boldsymbol{q}}\, \sigma_{\max}(\boldsymbol{q}) \tag{3.7}$$

It satisfies $\sigma_{\max}(\boldsymbol{q}) \leqslant \sigma_{\text{wc}}$ in every configuration and is calculated in (3.17). A Cartesian tolerance ϵ_c is then guaranteed in every configuration if the joint-space deviation is bounded by (3.8),

$$\|\Delta\boldsymbol{q}\|_2 \leqslant \epsilon_q := \frac{\epsilon_c}{\sigma_{\text{wc}}} \tag{3.8}$$

since inserting (3.8) into (3.6) yields $\|\Delta\boldsymbol{x}_{cart}\|_2 \leqslant \sigma_{\max}(\boldsymbol{q})\, \epsilon_q \leqslant \sigma_{\text{wc}}\, \epsilon_q = \epsilon_c$.

To calculate $\sigma_{\max}(\boldsymbol{q})$ and its worst-case value σ_{wc}, we use (3.9)

$$\sigma_{\max}(\boldsymbol{q}) = \|\boldsymbol{J}(\boldsymbol{q})\|_2 = \sqrt{\lambda_{\max}\left(\boldsymbol{J}^\top(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})\right)} \tag{3.9}$$

where $\lambda_{\max}$ denotes the largest eigenvalue of $\boldsymbol{J}^\top(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})$[36, Section 2.3, 2.4, 85, Section 3.9]. For the Jacobian (3.4) of the planar 2-DoF robot the product $\boldsymbol{J}^\top(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})$ simplifies with $\sin^2 q_1 + \cos^2 q_1 = 1$ and $\sin^2(q_1+q_2) + \cos^2(q_1+q_2) = 1$[21, s.v. trigonometric function] as well as with the compound angle formula $\sin(q_1+q_2)\sin q_1 + \cos(q_1+q_2)\cos q_1 = \cos(q_1+q_2-q_1) = \cos q_2$[21, s.v. compound angle formulae] to (3.10)

$$\boldsymbol{J}^\top(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q}) = \begin{bmatrix} a(q_2) & b(q_2) \\ b(q_2) & c \end{bmatrix} \tag{3.10}$$

with the abbreviations (3.11).

$$a(q_2) = l_1^2 + l_2^2 + 2l_1 l_2 \cos q_2, \qquad b(q_2) = l_2^2 + l_1 l_2 \cos q_2, \qquad c = l_2^2 \tag{3.11}$$

Since $\boldsymbol{J}^\top(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})$ depends only on q_2, its eigenvalues also depend only on q_2. The eigenvalues are calculated with the characteristic equation (3.12).

$$\begin{aligned} &\det\big(\boldsymbol{J}^\top(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q}) - \lambda I\big) = \big(a(q_2) - \lambda\big)\big(c - \lambda\big) - b(q_2)^2 = 0 \\ \Leftrightarrow \quad &\lambda^2 - \big(a(q_2) + c\big)\,\lambda + \big(a(q_2)\,c - b(q_2)^2\big) = 0 \end{aligned} \tag{3.12}$$

Solving the quadratic equation (3.12)[21, s.v. quadratic equation] and retaining the larger root gives (3.13).

$$\lambda_{\max}(q_2) = \frac{1}{2}\left[\big(a(q_2) + c\big) + \sqrt{\big(a(q_2) + c\big)^2 - 4\big(a(q_2)\,c - b(q_2)^2\big)}\right] \tag{3.13}$$

Expanding the constant term of (3.12) with (3.11) yields (3.14).

$$\begin{aligned} a(q_2)\,c - b(q_2)^2 &= l_2^2\big(l_1^2 + l_2^2 + 2l_1 l_2 \cos q_2\big) - \big(l_2^2 + l_1 l_2 \cos q_2\big)^2 \\ &= l_1^2 l_2^2\big(1 - \cos^2 q_2\big) \end{aligned} \tag{3.14}$$

The constant term thus attains its minimum, namely zero, at $q_2 = 0$, whereas the coefficient (3.15)

$$a(q_2) + c = l_1^2 + 2l_2^2 + 2l_1 l_2 \cos q_2 \tag{3.15}$$

attains its maximum at the same configuration. Since $\lambda_{\max}$ in (3.13) increases monotonically with $a(q_2) + c$ and decreases monotonically with $a(q_2)c - b(q_2)^2$, it is largest at $q_2 = 0$. There the square root in (3.13) reduces to $a(0) + c$, which gives (3.16).

$$\lambda_{\max}\big|_{q_2=0} = a(0) + c = (l_1 + l_2)^2 + l_2^2 \tag{3.16}$$

The worst-case value (3.7) is therefore (3.17)

$$\sigma_{\text{wc}} = \sqrt{\lambda_{\max}\big|_{q_2=0}} = \sqrt{(l_1 + l_2)^2 + l_2^2} \tag{3.17}$$

attained in the fully extended configuration $q_2 = 0$, which is at the same time a singularity as identified in Section 3.1.3.

The joint-space tolerance can therefore be calculated from the Cartesian tolerance with (3.18).

$$\epsilon_q = \frac{\epsilon_c}{\sqrt{(l_1 + l_2)^2 + l_2^2}} \tag{3.18}$$

3.2. Manipulator Dynamics

Following [23, Section 6.8, 6.9], the equations of motion of a manipulator can be derived with the Newton–Euler formulation as a force-balance approach or with the Lagrangian formulation as an energy-based approach. Both yield the same equations of motion, which can be written in the form (3.19)

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + g(\boldsymbol{q}) = \tau \tag{3.19}$$

where τ is the vector of external torques at the joints[23, Section 6.8, 6.9].

3.2.1. Planar 2-DoF Robot

Craig [23, Section 6.7] considered the planar 2-DoF robot from Section 3.1 with the mass of each link modelled as a point mass m_i at its distal end. In their model, gravity acts in the negative y-direction of the base frame[23, Section 6.7]. As in [23, Section 6.8], we use the abbreviations $s_i = \sin q_i$, $c_i = \cos q_i$ and $c_{12} = \cos(q_1 + q_2)$ to display the terms of (3.19) for this robot. The mass matrix in (3.20)

$$\boldsymbol{M}(\boldsymbol{q}) = \begin{bmatrix} m_2 l_2^2 + 2m_2 l_1 l_2 c_2 + (m_1 + m_2) l_1^2 & m_2 l_2^2 + m_2 l_1 l_2 c_2 \\ m_2 l_2^2 + m_2 l_1 l_2 c_2 & m_2 l_2^2 \end{bmatrix} \tag{3.20}$$

contains all terms that multiply $\ddot{\boldsymbol{q}}$ and depends only on the joint position. It is symmetric and positive definite and therefore always invertible. The vector $\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in (3.21)

$$\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \begin{bmatrix} -m_2 l_1 l_2 s_2 \dot{q}_2^2 - 2m_2 l_1 l_2 s_2 \dot{q}_1 \dot{q}_2 \\ m_2 l_1 l_2 s_2 \dot{q}_1^2 \end{bmatrix} \tag{3.21}$$

contains all terms that depend on the joint velocity and is a function of both $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$. Squares $\dot{q}_i^2$ of a single joint velocity stem from the centrifugal force and products $\dot{q}_i \dot{q}_j$ of two different joint velocities from the Coriolis force[23, Section 6.8]. We split $\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ into two factors as in (3.22).

$$\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = m_2 l_1 l_2 s_2 \begin{bmatrix} -\dot{q}_2^2 - 2\dot{q}_1 \dot{q}_2 \\ \dot{q}_1^2 \end{bmatrix} = h(\boldsymbol{q}) \boldsymbol{\Phi}(\dot{\boldsymbol{q}}) \tag{3.22}$$

Here $h(\boldsymbol{q}) = m_2 l_1 l_2 s_2$ collects the configuration-dependent part and $\boldsymbol{\Phi}(\dot{\boldsymbol{q}})$ the velocity-dependent part. The vector $\boldsymbol{\Phi}(\dot{\boldsymbol{q}})$ is quadratic in the joint velocities and therefore an even function of $\dot{\boldsymbol{q}}$ [21, s.v. even function], as shown in (3.23).

$$\boldsymbol{\Phi}(-\dot{\boldsymbol{q}}) = \begin{bmatrix} -(-\dot{q}_2)^2 - 2(-\dot{q}_1)(-\dot{q}_2) \\ (-\dot{q}_1)^2 \end{bmatrix} = \begin{bmatrix} -\dot{q}_2^2 - 2\dot{q}_1 \dot{q}_2 \\ \dot{q}_1^2 \end{bmatrix} = \boldsymbol{\Phi}(\dot{\boldsymbol{q}}) \tag{3.23}$$

The gravity vector $g(\boldsymbol{q})$ in (3.24)

$$g(\boldsymbol{q}) = \begin{bmatrix} m_2 l_2 g c_{12} + (m_1 + m_2) l_1 g c_1 \\ m_2 l_2 g c_{12} \end{bmatrix} \tag{3.24}$$

contains all terms with the gravitational acceleration g and depends only on the joint position, not on its derivatives[23, Section 6.8].

Siciliano [85, Section 7.3.2] considered a more general version of the planar 2-DoF robot from Section 3.1 and drew the centre of mass (CoM) of each link i on the link axis in the x-y plane at the distance $l_{c,i}$ from the axis of joint i. Each link has the inertia tensor I_i about its CoM and its mass m_i is located at the CoM. For this robot model, the mass matrix still depends on $\boldsymbol{q}$ only through c_2. In the Coriolis and centrifugal vector, only l_2 is replaced by $l_{c,2}$, so that its split form (3.22) keeps the same $\boldsymbol{\Phi}(\dot{\boldsymbol{q}})$ with $h(\boldsymbol{q}) = m_2 l_1 l_{c,2} s_2$[85, Section 7.3.2]. Thus $h(\boldsymbol{q})$ is still proportional to s_2 and $\boldsymbol{\Phi}(\dot{\boldsymbol{q}})$ remains even in $\dot{\boldsymbol{q}}$ as in (3.23).

3.2.2. Kinetic Energy, Velocity and Momentum

The kinetic energy of the manipulator is given in (3.25)[23, Section 6.9].

$$E_{kin} = \frac{1}{2} \dot{\boldsymbol{q}}^\top \boldsymbol{M}(\boldsymbol{q}) \dot{\boldsymbol{q}} \tag{3.25}$$

As $\boldsymbol{M}(\boldsymbol{q})$ is a symmetric positive definite matrix, it can be represented in a unique Cholesky decomposition $\boldsymbol{M}(\boldsymbol{q}) = \boldsymbol{L}\boldsymbol{L}^\top$, where $\boldsymbol{L}$ is a lower triangular matrix with positive diagonal entries[36, Theorem 4.2.7]. Substituting the decomposition into (3.25) yields (3.26)

$$2E_{kin} = \dot{\boldsymbol{q}}^\top \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}^\top \boldsymbol{L}\boldsymbol{L}^\top \dot{\boldsymbol{q}} = (\boldsymbol{L}^\top \dot{\boldsymbol{q}})^\top (\boldsymbol{L}^\top \dot{\boldsymbol{q}}) = \|\dot{\boldsymbol{q}}_{\text{mass}}\|^2 \tag{3.26}$$

where $\|\cdot\|$ is the Euclidean norm and $\dot{\boldsymbol{q}}_{\text{mass}} = \boldsymbol{L}^\top \dot{\boldsymbol{q}}$ the mass-weighted velocity. Since the kinetic energy of the manipulator vanishes if and only if all joint velocities vanish due to $\boldsymbol{M}(\boldsymbol{q})$ being positive definite[60, Section 8.1.3, 23, Section 6.9], it also vanishes when the mass-weighted velocity vanishes.

Using the representation in (3.25) one can parametrize the joint velocity with the kinetic energy and a direction of motion. Let the joint velocity $\dot{\boldsymbol{q}}$ be parametrized in polar coordinates[21, s.v. polar coordinates] with (3.27),

$$\dot{\boldsymbol{q}} = \|\dot{\boldsymbol{q}}\|\, \boldsymbol{d}(\alpha), \qquad \boldsymbol{d}(\alpha) = \begin{bmatrix} \cos\alpha \\ \sin\alpha \end{bmatrix}, \qquad \|\dot{\boldsymbol{q}}\| \geqslant 0 \tag{3.27}$$

where the angle $\alpha \in [-\pi, \pi)$ defines the direction in joint space and $\|\dot{\boldsymbol{q}}\|$ the velocity magnitude. Inserting (3.27) into (3.26) gives $2E_{kin} = \|\dot{\boldsymbol{q}}\|^2(\boldsymbol{d}^\top(\alpha)\, \boldsymbol{M}(\boldsymbol{q})\, \boldsymbol{d}(\alpha))$. Since $\boldsymbol{M}(\boldsymbol{q})$ is positive definite and $\boldsymbol{d}(\alpha) \neq \boldsymbol{0}$ holds, the term $\boldsymbol{d}^\top(\alpha)\, \boldsymbol{M}(\boldsymbol{q})\, \boldsymbol{d}(\alpha)$ is positive and the magnitude follows as (3.28).

$$\|\dot{\boldsymbol{q}}\| = \sqrt{\frac{2E_{kin}}{\boldsymbol{d}^\top(\alpha)\, \boldsymbol{M}(\boldsymbol{q})\, \boldsymbol{d}(\alpha)}} \tag{3.28}$$

Hence, for a given configuration $\boldsymbol{q}$, every nonzero joint velocity corresponds to exactly one pair (E_{kin}, α) with $E_{kin} > 0$. For $E_{kin} = 0$, (3.28) yields $\|\dot{\boldsymbol{q}}\| = 0$ and thus $\dot{\boldsymbol{q}} = \boldsymbol{0}$, for which the angle α is undefined.

The momentum $\boldsymbol{\rho}$ of the manipulator is defined in (3.29) together with its correspondence to the joint velocity[92].

$$\boldsymbol{\rho} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}\,, \qquad \dot{\boldsymbol{q}} = \boldsymbol{M}^{-1}(\boldsymbol{q})\boldsymbol{\rho}\,, \qquad \boldsymbol{\rho}^\top = \dot{\boldsymbol{q}}^\top \boldsymbol{M}(\boldsymbol{q}) \tag{3.29}$$

3.2.3. Point-Symmetric Oscillation in Joint Space

For a system with time-reversal symmetry, the time-reversed trajectory $\boldsymbol{q}_T(t)$ in (3.30) of a trajectory $\boldsymbol{q}(t)$ in configuration space is again a trajectory of the system[55]. Following [83], if the manipulator of a conservative system comes to rest in two configurations $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$ at the instants t_A and t_B, the momentum in (3.29) vanishes at both instants and the motion is a periodic oscillation with period $T = 2|t_B - t_A|$ that comes to rest only in these two configurations. With the deviation $\boldsymbol{\delta}(t)$ from a point $\boldsymbol{c}_m$, a trajectory can be written as $\boldsymbol{q}(t) = \boldsymbol{c}_m + \boldsymbol{\delta}(t)$. Its reflection through $\boldsymbol{c}_m$ gives the point-reflected trajectory $\boldsymbol{q}_I(t)$ in (3.31). It is again a trajectory of the system if the mass matrix and the potential field are even-symmetric about $\boldsymbol{c}_m$ as in (3.32).

$$\boldsymbol{q}_T(t) = \boldsymbol{q}(-t)\,, \tag{3.30}$$

$$\boldsymbol{q}_I(t) = \boldsymbol{c}_m - \boldsymbol{\delta}(t) = 2\boldsymbol{c}_m - \boldsymbol{q}(t)\,, \tag{3.31}$$

$$\boldsymbol{M}(\boldsymbol{c}_m + \boldsymbol{\delta}) = \boldsymbol{M}(\boldsymbol{c}_m - \boldsymbol{\delta})\,, \qquad \boldsymbol{V}(\boldsymbol{c}_m + \boldsymbol{\delta}) = \boldsymbol{V}(\boldsymbol{c}_m - \boldsymbol{\delta}) \tag{3.32}$$

The rest configurations satisfy $\boldsymbol{q}_A = 2\boldsymbol{c}_m - \boldsymbol{q}_B$ and the oscillation is point-symmetric around $\boldsymbol{c}_m = \boldsymbol{q}(0)$. Its time-reversed and point-reflected trajectories coincide for all t as in (3.33).

$$\boldsymbol{q}_T(t) = \boldsymbol{q}_I(t) \qquad \forall t \tag{3.33}$$

Starting in $\boldsymbol{c}_m$ at $t = 0$, it passes the following configurations[83]:

- $t = \frac{T}{4}$: rest in $\boldsymbol{q}_B$
- $t = \frac{T}{2}$: $\boldsymbol{c}_m$ with the momentum reversed compared to the start in $\boldsymbol{c}_m$ at $t = 0$. Due to the same mass matrix $\boldsymbol{M}(\boldsymbol{c}_m)$, the joint velocity is also reversed.
- $t = \frac{3T}{4}$: rest in $\boldsymbol{q}_A$
- $t = T$: $\boldsymbol{c}_m$ with the initial joint velocity

3.3. Numerical Integration with Runge–Kutta 4

Solving (3.19) for the joint acceleration $\ddot{\boldsymbol{q}}$[23, Section 6.12] and introducing the state $\boldsymbol{y} = (\boldsymbol{q}, \dot{\boldsymbol{q}})^\top$ yields the first-order system (3.34) with the external torque τ.

$$\dot{\boldsymbol{y}} = \boldsymbol{f}_{rk}(t, \boldsymbol{y}) = \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{M}(\boldsymbol{q})^{-1}\big(\tau - \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - g(\boldsymbol{q})\big) \end{pmatrix} \tag{3.34}$$

Following [17, Section 5.4], the classical Runge–Kutta method of order four (RK4) evaluates $\boldsymbol{f}_{rk}$ four times per step at different points of the step without computing the derivatives of $\boldsymbol{f}_{rk}$. Starting from $\boldsymbol{w}_0 = \boldsymbol{y}(t_0)$, it computes the approximation $\boldsymbol{w}_{i+1} \approx \boldsymbol{y}(t_{i+1})$ with the step size Δt in the RK4 integration (3.35)[17, Section 5.4].

$$\begin{aligned} \boldsymbol{\kappa}_{rk,1} &= \Delta t\, \boldsymbol{f}_{rk}(t_i, \boldsymbol{w}_i)\,, \\ \boldsymbol{\kappa}_{rk,2} &= \Delta t\, \boldsymbol{f}_{rk}\big(t_i + \tfrac{\Delta t}{2}, \boldsymbol{w}_i + \tfrac{1}{2}\boldsymbol{\kappa}_{rk,1}\big)\,, \\ \boldsymbol{\kappa}_{rk,3} &= \Delta t\, \boldsymbol{f}_{rk}\big(t_i + \tfrac{\Delta t}{2}, \boldsymbol{w}_i + \tfrac{1}{2}\boldsymbol{\kappa}_{rk,2}\big)\,, \\ \boldsymbol{\kappa}_{rk,4} &= \Delta t\, \boldsymbol{f}_{rk}(t_{i+1}, \boldsymbol{w}_i + \boldsymbol{\kappa}_{rk,3})\,, \\ \boldsymbol{w}_{i+1} &= \boldsymbol{w}_i + \tfrac{1}{6}\big(\boldsymbol{\kappa}_{rk,1} + 2\boldsymbol{\kappa}_{rk,2} + 2\boldsymbol{\kappa}_{rk,3} + \boldsymbol{\kappa}_{rk,4}\big)\,, \\ t_{i+1} &= t_i + \Delta t \end{aligned} \tag{3.35}$$

We call one RK4 integration according to (3.35) a *full step* and each of the four evaluations of $\boldsymbol{f}_{rk}$ in $\boldsymbol{\kappa}_{rk,1}, \ldots, \boldsymbol{\kappa}_{rk,4}$ a *substep*, where we evaluate $\boldsymbol{f}_{rk}$ at t_i, $t_i + \frac{\Delta t}{2}$ and $t_i + \Delta t$.

For RK4 with order $p = 4$, $\boldsymbol{f}_{rk} \in C^4$ leads to a local error of $\mathcal{O}(\Delta t^5)$ per full step and together with a Lipschitz condition on $\boldsymbol{f}_{rk}$ to a global error of $\mathcal{O}(\Delta t^4)$[39, Chapter II.3]. To determine whether the manipulator dynamics satisfy these conditions, we look at the terms in $\boldsymbol{f}_{rk}$. The inverse of the mass matrix in (3.34) is given by (3.36), where m_{ij} denotes the entry of $\boldsymbol{M}(\boldsymbol{q})$ in row i and column j[21, s.v. inverse matrix].

$$\boldsymbol{M}(\boldsymbol{q})^{-1} = \frac{1}{m_{11}m_{22} - m_{12}m_{21}} \begin{bmatrix} m_{22} & -m_{12} \\ -m_{21} & m_{11} \end{bmatrix} \tag{3.36}$$

Inserting the entries of (3.20) and using $\sin^2 q_1 + \cos^2 q_1 = 1$[21, s.v. trigonometric function] yields the non-vanishing determinant in (3.37).

$$m_{11}m_{22} - m_{12}m_{21} = m_2 l_1^2 l_2^2 \big(m_1 + m_2 s_2^2\big) > 0 \tag{3.37}$$

Since additionally all entries of $\boldsymbol{M}(\boldsymbol{q})^{-1}$, $\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $g(\boldsymbol{q})$ in Section 3.2 are four times continuously differentiable, $\boldsymbol{f}_{rk} \in C^4$ holds if $\tau \in C^4$ holds with respect to t and $\boldsymbol{y}$. When $\boldsymbol{f}_{rk} \in C^4$, it is in particular continuously differentiable and therefore locally Lipschitz continuous[52, Lemma 3.2].

3.4. Covariance Matrix Adaptation Evolution Strategy

According to [40], Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is a randomized method for optimizing real-valued parameters in a continuous domain. The objective function does not have to be linear or convex. Even in a rugged search landscape, for example one with discontinuities, sharp ridges or local optima, the method remains robust [40].

As described in [40], CMA-ES minimizes an objective function $f_{cma}\colon \mathbb{R}^{n_{dim}} \to \mathbb{R}$ that it treats as a black box. Apart from the function values at the evaluated points, the method has no access to f_{cma}. Figure 3.2 shows the resulting loop of a randomized black-box search. At the start, the parameters $\theta^{(0)}$ of the search distribution are initialized. Each pass through the loop then consists of three steps. First, λ independent points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_\lambda$ are sampled from the search distribution $P(\boldsymbol{x} \mid \theta^{(g)})$. Second, the objective function f_{cma} is evaluated at these points. Third, the update rule F_θ in (3.38) determines the parameters of the next pass from the current parameters and the evaluated points.

$$\theta^{(g+1)} = F_\theta\big(\theta^{(g)}, (\boldsymbol{x}_1, f_{cma}(\boldsymbol{x}_1)), \ldots, (\boldsymbol{x}_\lambda, f_{cma}(\boldsymbol{x}_\lambda))\big) \tag{3.38}$$

One pass through these three steps is called a *generation* and the generations are numbered as $g = 0, 1, 2, \ldots$. After each update, a termination criterion is checked and the loop stops as soon as it is met [40].

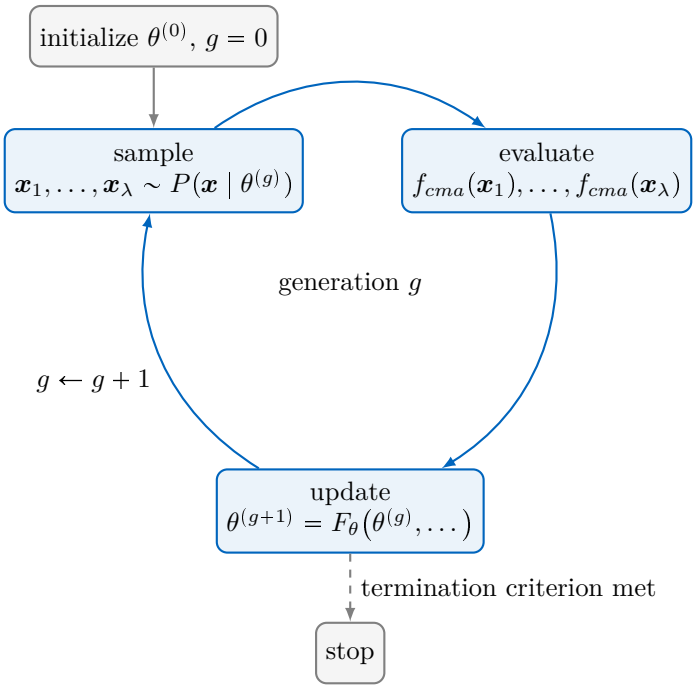


Figure 3.2.: Loop of a randomized black-box search as used by CMA-ES. After the initialization of $\theta^{(0)}$, each generation g samples λ points from the search distribution, evaluates them on f_{cma} and updates the distribution parameters according to (3.38). The loop stops once a termination criterion is met. (AI-generated illustration)

Following [40], the search distribution P of CMA-ES is a multivariate normal distribution. In generation g, it is described by the mean $\boldsymbol{m}^{(g)} \in \mathbb{R}^{n_{dim}}$, the covariance matrix $\boldsymbol{C}^{(g)} \in \mathbb{R}^{n_{dim} \times n_{dim}}$ and the overall standard deviation $\sigma^{(g)} > 0$, which is also called step size. From this distribution,

the search points of the next generation are drawn according to (3.39).

$$\boldsymbol{x}_k^{(g+1)} \sim \boldsymbol{m}^{(g)} + \sigma^{(g)}\mathcal{N}\left(\mathbf{0}, \boldsymbol{C}^{(g)}\right) \sim \mathcal{N}\left(\boldsymbol{m}^{(g)}, (\sigma^{(g)})^2\boldsymbol{C}^{(g)}\right), \quad k = 1, \ldots, \lambda \tag{3.39}$$

The λ search points $\boldsymbol{x}_k^{(g+1)} \in \mathbb{R}^{n_{dim}}$ together form the population of generation $g+1$ [40]. The Python implementation `pycma` sets the default population size to $\lambda = 4 + \lfloor 3 \ln n_{dim} \rfloor$ [74].

As explained in [40], the covariance matrix $\boldsymbol{C}$ generally has to be symmetric and positive definite. Geometrically, it corresponds to exactly one (hyper-)ellipsoid $\{\boldsymbol{x} \in \mathbb{R}^{n_{dim}} \mid \boldsymbol{x}^\top \boldsymbol{C}^{-1} \boldsymbol{x} = 1\}$, on which the density of the distribution is constant. In (3.39), the step size $\sigma^{(g)}$ scales this ellipsoid equally in all directions, while $\boldsymbol{C}^{(g)}$ determines its shape and orientation [40]. Here, `pycma` offers the option to provide one multiplier per coordinate for $\sigma^{(0)}$ through the array `CMA_stds`, so that users can scale the ellipsoid differently in each coordinate [74]. Figure 3.3 sketches such ellipses over several generations of a two-dimensional example, where one can see how the ellipse moves together with the mean to the minimum over the generations and contracts there.

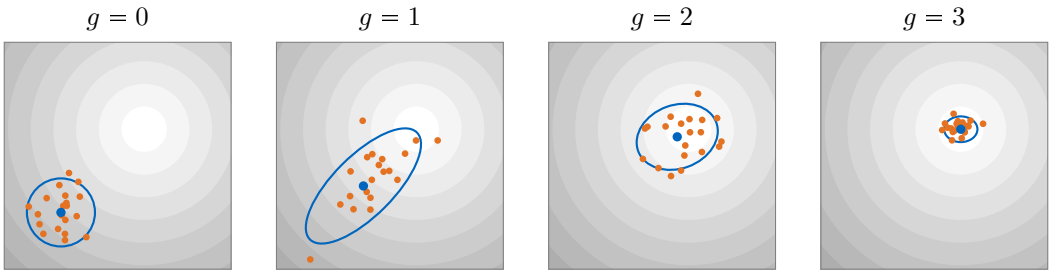


Figure 3.3.: Sketch of CMA-ES minimizing a two-dimensional function whose value decreases towards the light centre[73]. In each generation g, the blue dot marks the mean $\boldsymbol{m}^{(g)}$ and the blue ellipse the geometric representation of the covariance of the normal distribution[40]. The orange points are sampled from this distribution. The sketch is inspired by the illustration in [73]. (AI-generated illustration)

Following [22], `pycma` takes the initial mean `x0`$=\boldsymbol{m}^{(0)}$ and the initial step size `sigma0`$=\sigma^{(0)}$ before the first generation. A single initial step size for all coordinates presupposes that the variables are scaled comparably or will be properly scaled with `CMA_stds`. The value of $\sigma^{(0)}$ should then be chosen such that the optimum is expected to lie roughly within $\boldsymbol{m}^{(0)} \pm 3\sigma^{(0)}$ in each coordinate [22]. As a rule of thumb, $\sigma^{(0)}$ is about a quarter of the width of the search domain in which the optimum is expected [30].

3.5. Travelling Salesman Problem

Following [8], a salesman in the classical travelling salesman problem (TSP) leaves a base city, has to pass through each of the other cities exactly once and finally comes back to the base city. The question is which route keeps the total distance as short as possible. In the dynamic programming treatment of the TSP, the origin of the tour is fixed at one of the cities. If an optimal tour is cut at any intermediate city, the remaining part leads from this city through all cities not yet visited and back to the origin. This remaining part must itself be the shortest path through the remaining cities [8].

Following [80], the generalized travelling salesman problem (GTSP) extends the classical TSP. Its structure resembles a graph whose vertices are split into disjoint non-empty groups called clusters[80]. In this work, a cluster is called a *node* n and the vertices it contains are its *node*

members $s \in n$. Links between two members of the same node are *intra-node links*, while links between members of different nodes are *inter-node links*. Of the two versions of the GTSP classified in [80], we consider in this work the one in which a closed tour enters every node exactly once. Such a *generalized Hamiltonian tour* is sketched in Figure 3.4. The GTSP asks for the shortest tour of this kind. Dynamic programming approaches to the GTSP suffer mainly from a state space that grows exponentially in the number of nodes [80].

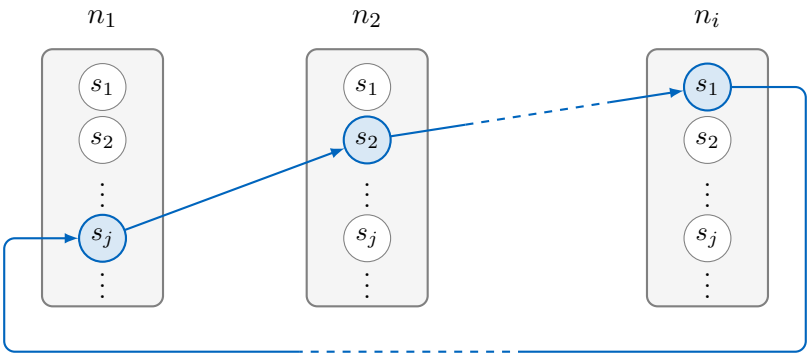


Figure 3.4.: Sketch of a generalized Hamiltonian tour that starts and ends at member s_j of node n_1. Dashed parts of a link stand for further nodes. The sketch is inspired by the illustration in [80]. (AI-generated illustration)

Pop et al. [80] report exact dynamic programming approaches for the GTSP that build on those for the classical TSP and describe a state by the nodes visited so far. We transfer the recursion of Held and Karp [42], in which we replace each city with a pair (n, s) of a node and one of its node members. They regard the TSP as the search for the cyclic order of all cities that gives the cheapest complete tour. As in Figure 3.4, the tour starts and ends at member s_j of node n_1. For a set $\mathcal{N}$ of nodes without n_1, $\mathrm{TSP}\big(\mathcal{N}, (n, s)\big)$ denotes the minimal cost of a path that starts at (n_1, s_j), passes through all nodes of $\mathcal{N}$ and ends at (n, s) with $n \in \mathcal{N}$. The recursion (3.40) splits this path at its last inter-node link into the cheapest path to a predecessor $(n_{\mathrm{prev}}, s_{\mathrm{prev}})$ and the cost of the link from there to (n, s).

$$\mathrm{TSP}\big(\mathcal{N}, (n, s)\big) = \min_{\substack{n_{\mathrm{prev}} \in \mathcal{N} \setminus \{n\} \\ s_{\mathrm{prev}} \in n_{\mathrm{prev}}}} \Big[\mathrm{TSP}\big(\mathcal{N} \setminus \{n\}, (n_{\mathrm{prev}}, s_{\mathrm{prev}})\big) + \mathrm{cost}\big((n_{\mathrm{prev}}, s_{\mathrm{prev}}), (n, s)\big) \Big] \tag{3.40}$$

If $\mathcal{N}$ contains all nodes except n_1, the link back to (n_1, s_j) closes the tour. Its minimal cost $\mathrm{TSP}(\mathcal{N})$ follows from (3.41).

$$\mathrm{TSP}(\mathcal{N}) = \min_{n \in \mathcal{N},\, s \in n} \Big[\mathrm{TSP}\big(\mathcal{N}, (n, s)\big) + \mathrm{cost}\big((n, s), (n_1, s_j)\big) \Big] \tag{3.41}$$

4

Approach and Concept

Since the research question in Section 1.1 considers a SCARA robot with clutched parallel elastic actuators (CPEAs), Section 4.1 models the 2-DoF SCARA robot with CPEAs inspired by Penzlin et al. [76] in each joint.

To answer this research question, we first look at a single pick-and-place task (Definition 1). The term task from here on always refers to a pick-and-place task and its two positions are referred to as A and B.

Definition 1 (Pick-and-Place Task)**.** *A pick-and-place task consists of a Cartesian pick position and a Cartesian place position at which the end effector of the robot has to stop to pick up or place an object.*

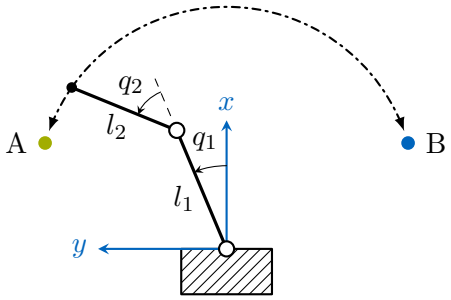


Figure 4.1.: Pick-and-place motion of a robot with link lengths l_1 and l_2 and joint angles q_1 and q_2 between Cartesian points A and B (AI-generated illustration)

The dash-dotted line in Figure 4.1 shows the (quasi-)conservative pick-and-place motion from Definition 2 between the task's positions A and B. Our goal is to let the 2-DoF planar robot swing freely back and forth between A and B, where it momentarily comes to rest at each position.

Definition 2 ((Quasi-)Conservative Pick-and-Place Motion)**.** *A (quasi-)Conservative pick-and-place motion is a motion of the robot that fulfils a task. The motion is conservative in a conservative system and quasi-conservative if the motion is applied in a non-conservative system.*

Section 4.2 further explains the role of the CPEA's spring in these free oscillations and Section 4.3 focuses on finding free oscillations between different pick and place positions A and B across the whole workspace.

Since we are looking for quasi-conservative pick-and-place sequences (Definition 4) in our research question, Section 4.2 explains the role of the motor for a pick-and-place task set (Definition 3).

Definition 3 (Pick-and-Place Task Set)**.** *A pick-and-place task set is a set of tasks that the robot has to fulfil one after the other in an arbitrary order.*

Definition 4 ((Quasi-)Conservative Pick-and-Place Sequence)**.** *A (quasi-)conservative pick-and-place sequence is a sequence of equal-energy-level (quasi-)conservative pick-and-place motions where each intersects the previous one in configuration space.*

In this work, we consider task sets that can be executed repeatedly with closed (quasi-) conservative pick-and-place sequences (Definition 5).

Definition 5 (Closed (Quasi-)Conservative Pick-and-Place Sequence)**.** *A closed (quasi-)conservative pick-and-place sequence is a (quasi-)conservative pick-and-place sequence whose last motion intersects the first one in configuration space, such that the sequence can be repeated indefinitely.*

As we are searching for free oscillations for different tasks, modelling the robot's behaviour in a conservative system suffices and free oscillations and their combination into sequences can then be considered as conservative pick-and-place motions and sequences.

Section 4.4 determines the momentum that the motor has to deliver within closed conservative pick-and-place sequences and defines an example task set for which we answer the research question.

To answer in which regions of its workspace the robot can execute closed conservative pick-and-place sequences, we only consider valid tasks and task sets (Definition 6) and determine their workspace coverage (Definition 7).

Definition 6 (Valid Task and Task Set)**.** *A task or task set is valid if all its pick and place positions lie in the workspace of the robot.*

Definition 7 (Workspace Coverage)**.** *The workspace coverage comprises all valid tasks or task sets for which a (quasi-)conservative pick-and-place motion or sequence was found.*

If several suitable conservative pick-and-place sequences were found for the same task set, the research sub-question adds the sequence duration (Definition 8) as a criterion to choose among them.

Definition 8 (Sequence Duration)**.** *The sequence duration is the time required to execute a (quasi-)conservative pick-and-place sequence.*

4.1. Robot Model

We consider the planar 2-DoF robot from Section 3.1 as shown in Figure 4.2, where its arm is stretched along the x-axis at $\boldsymbol{q} = (0, 0)$ and positive joint angles rotate from the x-axis towards the y-axis. The base frame $\{0\}$ and the frame $\{1\}$ of link 1 lie in joint 1 and the frame $\{2\}$ of link 2 lies in joint 2. The x-axis of each link frame points along its link, while all z-axes point

along the joint axes out of the plane of motion. As in the general case of Section 3.2, the CoM of each link i lies on its link axis, the x_i-axis, in the top view of the SCARA robot in Figure 4.2. The mass m_i of link i is located at its CoM, about which the link has the inertia tensor I_i. The gripper with the mass $m_{gripper}$ and the inertia tensor $I_{gripper}$ is rigidly attached to the end of link 2 on its x_2-axis. Considered as one rigid body, link 2 and the gripper have a mass, a CoM and an inertia tensor that differ from those of link 2 alone, while their common CoM still lies on the x_2-axis in the top view in Figure 4.2. In this work, we consider the x- and y-axes as the x_0- and y_0-axes of the base frame.

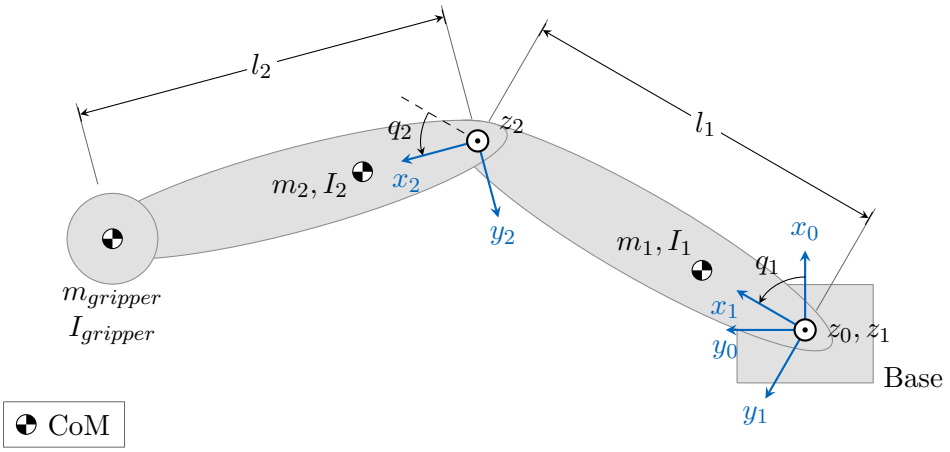


Figure 4.2.: Top view of the kinematic and dynamic model of the SCARA robot (AI-generated illustration)

For the clutched parallel elastic actuator in each joint, we adopt the spring model (4.1) of Penzlin et al. [76], in which a linear spring with stiffness k_i exerts a torque proportional to the deflection of joint i from the spring equilibrium $q_{\text{entry},i}$, which can be placed at an arbitrary angle.

$$\tau_{spring,i}(\boldsymbol{q}) = -k_i \left(q_i - q_{\text{entry},i}\right) \tag{4.1}$$

Figure 4.3 shows an overview of our SCARA manipulator with its two revolute joints. Figure 4.4 displays the joints with the torsional springs in detail, showing that the fixed-stiffness springs lie in parallel to the motors and can be connected to and disconnected from the joints by the clutches. For our conservative system, we follow Plooij, Wisse, and Vallery [78] in assuming that the clutches change their state without delay. As in their actuator, each joint passes through three phases when oscillating between two Cartesian positions. When the clutch of joint i engages, it stores the current angle of that joint as the constant value $q_{\text{entry},i}$ and keeps it as the equilibrium of its torsional spring until it disengages. Due to the potential field (4.2) created by the springs,

$$\boldsymbol{V}(\boldsymbol{q}) = \sum_{i=1}^{2} \frac{1}{2} k_i \left(q_i - q_{\text{entry},i}\right)^2 \tag{4.2}$$

the joint is decelerated until it comes to rest and then accelerated again. In the phase between accelerating and decelerating, the CPEA at joint i disengages its clutch and acts like an actuator without parallel elasticity[76]. With the clutch disengaged, no spring torque acts on joint i and $q_{\text{entry},i} = q_i(t)$ holds in (4.2) and (4.1). The spring at joint i has its equilibrium at every joint position within this phase, forming a region of spring equilibria. Since the two clutches switch their state independently of each other, the robot's equilibria lie within a joint-space region in which neither clutch is engaged.

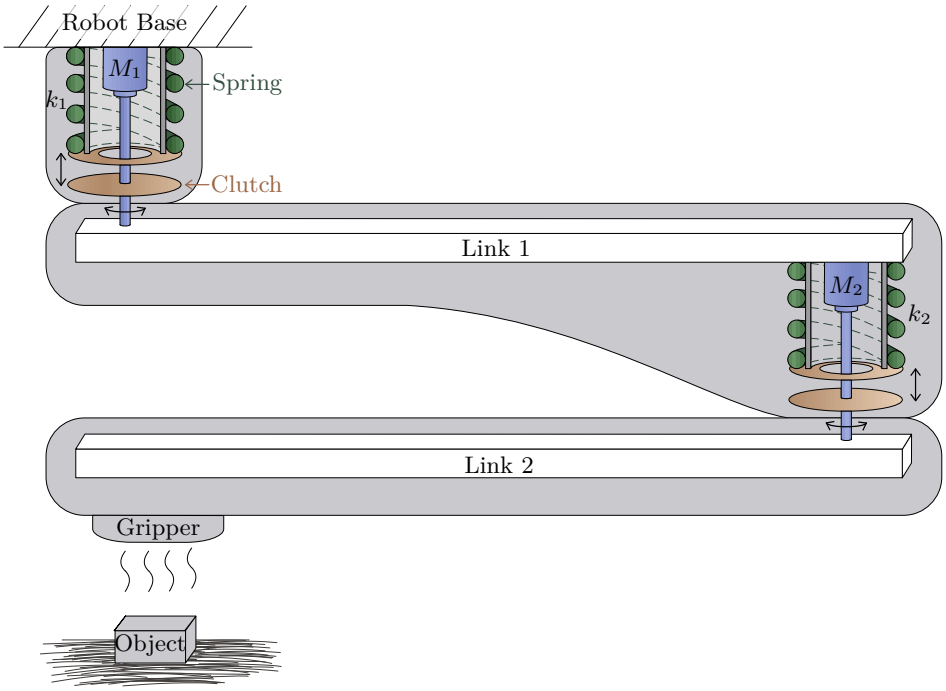


Figure 4.3.: SCARA manipulator with a torsional spring parallel to the motor and a clutch at each joint

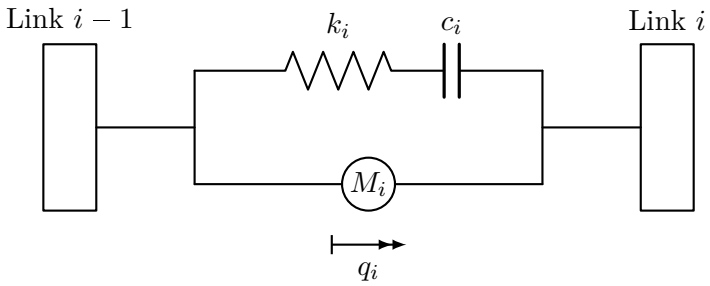


Figure 4.4.: Joint i with the motor M_i parallel to the spring k_i and the clutch c_i (AI-generated illustration)

4.2. Role Division between Spring and Motor

As already mentioned in Section 1.1 and Section 4, we want to exploit the natural motion of a planar SCARA robot and find oscillations between its pick and place positions. In the conservative system, no friction, contact or damping forces act on the manipulator dynamics. Since the manipulator moves horizontally, the gravity term $g(\boldsymbol{q})$ in (3.19) vanishes. The spring torque $\boldsymbol{\tau_{spring}}(\boldsymbol{q})$ of the clutch mechanism introduced in Section 4.1 and implemented in Section 5.1.2 is the only generalized force acting on the system and is derived from the negative gradient of the potential field $\boldsymbol{V}(\boldsymbol{q})$ [3, Theorem p.48]. The manipulator dynamics thus take the form (4.3).

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau_{spring}}(\boldsymbol{q}) = -\partial \boldsymbol{V}(\boldsymbol{q})/\partial \boldsymbol{q} \tag{4.3}$$

In a real system, the motor of the modelled CPEA in Section 4.1 would only have to compensate for the non-conservative forces not modelled in (4.3) as described in Section 2.2.

4.2.1. Spring Role

As explained in Section 2.2, elastic elements enable the robot dynamics to freely oscillate between the pick and place positions A and B. The role of the CPEAs' springs is thus to create a desired potential field such that the oscillation stops in the task's pick and place positions. In this work we only consider oscillations where both joints stop at the inverse kinematics solutions of the Cartesian positions A and B. At those two resting positions of the oscillation, the joint velocity and thus the robot's kinetic energy vanish. The total energy is then completely stored as potential energy in the springs. Since these free oscillations are periodic in a conservative system [83, Lemma 1], they can also be referred to as nonlinear normal modes [51].

For the pick-and-place task described in Section 4, we assume that the manipulator carries no payload and that no obstacles lie in its way. We further assume that the gripper grasps and releases the object instantly, so that the manipulator does not need to dwell at the pick and the place position. Possible ways to relax this assumption are discussed in Section 7.2.

4.2.2. Motor Role

The research question in Section 1.1 concerns not a single task but a task set, whose pick and place positions vary across an area within the workspace. The point-symmetric oscillations described in Section 4.3.2 share the same equilibrium, where no spring force acts and the clutches are disengaged. At this shared equilibrium the oscillations intersect and have $E_{kin} = E_{total}$. The motor's role is to deliver the impulsive transition (Definition 9) at this shared equilibrium to transition between the two motions fulfilling two tasks of the task set.

Definition 9 (Impulsive Transition)**.** *A transition between consecutive motions of a conservative pick-and-place sequence at an intersection point through an instantaneous change of momentum.*

4.3. Progressive Problem Refinement for Single Tasks

As introduced with Figure 4.1 and further refined in Section 4.2, a pick-and-place task is fulfilled by an oscillation between two points A and B with their cartesian position vectors $\boldsymbol{p}_A \neq \boldsymbol{p}_B$, at which both joints come to rest at the same time. Each of the two points can be the pick or the place position. To describe the oscillation in joint space, we first determine a unique representation of the joint angles to a reference configuration in Definition 10.

Definition 10 (Unique Configuration Representation)**.** *Extending the convention from Section 3.1, all joint angles of the configuration representation are given relative to a reference configuration* $\boldsymbol{q}_{\text{ref}}$ *with* $\boldsymbol{q} - \boldsymbol{q}_{\text{ref}} \in [-\pi, \pi) \times [-\pi, \pi)$*. This way every configuration of the manipulator has a unique representation relative to* $\boldsymbol{q}_{\text{ref}}$*. If no reference configuration is stated,* $\boldsymbol{q}_{\text{ref}} = (0, 0)$ *is used.*

The representation also determines the joint-space distance between different configurations in Definition 11.

Definition 11 (Joint-Space Distance)**.** *The joint-space distance between two configurations is the Euclidean distance between their joint angles in the given representation.*

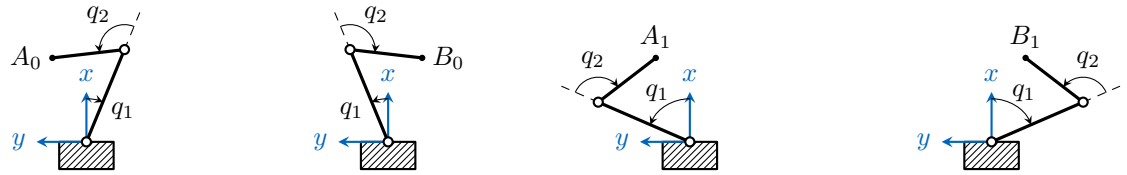
Since every joint angle in Definition 10 differs by at most π from $\boldsymbol{q}_{\text{ref}}$, the unique representation has the shortest joint-space distance to $\boldsymbol{q}_{\text{ref}}$ of all representations of a configuration. The

oscillation itself and its trajectory can be described by the turns of its joints along the path (Definition 12).

Definition 12 (Single-Turn and Multi-Turn)**.** *In a single-turn trajectory or oscillation, the largest and smallest angle of each joint on the trajectory differ by at most* 2π*. In a multi-turn trajectory or oscillation, they differ by more than* 2π *for at least one joint.*

Definition 13 then introduces a notation for the inverse kinematics solutions of an arbitrary Cartesian point P.

Definition 13 (Configuration Notation for a Cartesian Point P)**.** *Let* P *be a Cartesian point with the position vector* $\boldsymbol{p}_P = (p_{P,x}, p_{P,y})$*. Its configuration solutions are indexed in their unique representation relative to* $\boldsymbol{q}_{\text{ref}} = (0, 0)$ *from Definition 10. The index stays the same when a configuration is represented relative to another reference. In a singularity* (3.5)*,* P *has a single configuration solution* $\boldsymbol{q}_{P_0} = (q_{P_0,1}, q_{P_0,2})$*. The only exception is the origin for* $l_1 = l_2$*, which is reached by infinitely many inverse kinematics solutions (Section 3.1.2). Of these infinitely many configuration solutions, we define* $\boldsymbol{q}_{P_0} = (0, -\pi)$ *as the one with index* 0*. All other configurations of the origin are not uniquely identified by an index and are given without an index as* $\boldsymbol{q}_P = (q_{P,1}, -\pi)$*. Outside a singularity, applying the inverse kinematics* (3.3) *yields the two configuration solutions* $\boldsymbol{q}_{P_0} = (q_{P_0,1}, q_{P_0,2})$ *and* $\boldsymbol{q}_{P_1} = (q_{P_1,1}, q_{P_1,2})$ *with* $q_{P_1,2} = -q_{P_0,2}$ *(Section 3.1.2, Section 3.1.3). Index* 0 *denotes the solution with* $q_2 > 0$ *if* $p_{P,x}p_{P,y} > 0$ *or if* P *lies on the positive* x*- or* y*-axis. Otherwise index* 0 *denotes the solution with* $q_2 < 0$*. We further write* P_i *with its Cartesian vector* $\boldsymbol{p}_{P_i} = \boldsymbol{p}_P$ *for point* P *taken in configuration* $\boldsymbol{q}_{P_i}$ *with* $i \in \{0, 1\}$*. When no index* i *is given, the inverse kinematics solution is not specified.*



(a) A_0 with configuration index 0 as $q_{A_0,2} > 0$ (b) B_0 with configuration index 0 as $q_{B_0,2} < 0$ (c) A_1 with configuration index 1 as $q_{A_1,2} < 0$ (d) B_1 with configuration index 1 as $q_{B_1,2} > 0$

Figure 4.5.: Inverse kinematics solutions of a point A with $p_{A,x}p_{A,y} > 0$ and a point B with $p_{B,x}p_{B,y} < 0$ (AI-generated illustration)

Applied to A and B, Definition 13 yields the configurations $\boldsymbol{q}_{A_0}$, $\boldsymbol{q}_{A_1}$, $\boldsymbol{q}_{B_0}$ and $\boldsymbol{q}_{B_1}$. Figure 4.5 shows them for a point A with $p_{A,x}p_{A,y} > 0$ and a point B with $p_{B,x}p_{B,y} < 0$.

We solve the problem of finding oscillations between A and B in three stages. Each stage refines the problem formulation of the previous one and brings it closer to the target scenario. A stage's optimization result is successful if it solves the problem of this stage within numerical tolerances. The successful results then serve as initial guesses for the next stage. The first stage in Section 4.3.1 is the least restrictive formulation and only has to reach B for arbitrary points A and B. Afterwards, we reduce the problem to find oscillations between A and B in Section 4.3.2 to point-symmetric oscillations in joint space. The second stage in Section 4.3.3 searches for stiffness values for which joint-space point-symmetric oscillations can be found. Since the CPEAs in Section 4.1 have fixed-stiffness springs, the last stage in Section 4.3.4 keeps one chosen stiffness per spring for all tasks and adapts the oscillations to the tasks through the times at which the clutches engage and disengage. The successful results of the last stage are the oscillations that fulfil the tasks.

To keep the results comparable within and across the stages, their trajectories start in an initial configuration before swinging to B. Additionally, we hold the total energy of the conservative system defined in Section 4.2 constant across all optimizations, since in a nonlinear system the amplitude and the period T of an oscillation change with its energy [25, 82] and since this guarantees equal-energy-level motions for conservative pick-and-place sequences (Definition 4).

4.3.1. No-Spring Optimization

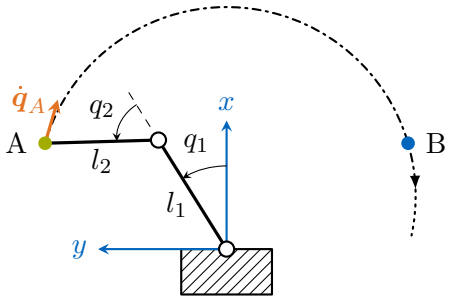


Figure 4.6.: Trajectory of a robot that starts in A with the initial velocity $\dot{\boldsymbol{q}}_A$ and passes through B without stopping (AI-generated illustration)

The first stage analyses the general dynamic behaviour of the system without a potential field, which means that all clutches stay disengaged. Unless stated otherwise, we use the starting configuration $\boldsymbol{q}_A = \boldsymbol{q}_{A_0}$ and the target configuration $\boldsymbol{q}_B = \boldsymbol{q}_{B_0}$. As the trajectory cannot stop in B without a potential field, the trajectory can only go through B as shown in Figure 4.6. We thus start the free-response shooting in $\boldsymbol{q}_A$ where we optimize for the initial velocity $\dot{\boldsymbol{q}}_A$ such that the trajectory reaches B. The robustness analysis in Section 5.2.5 shows that the trajectories toward B are robust enough for the successful results of this stage to serve as initial guesses for the optimization in Section 4.3.3.

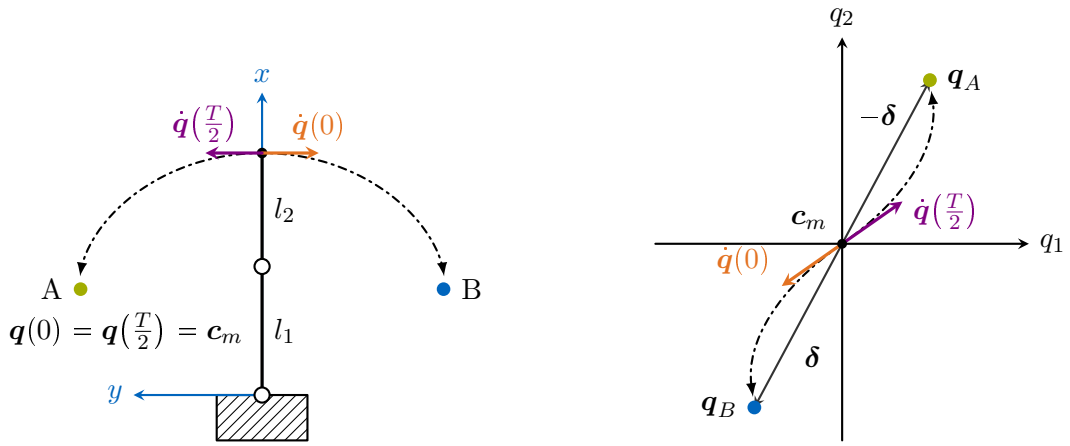
4.3.2. Joint-Space Point-Symmetric Oscillations

In this reduction, A and B are axis-symmetric with respect to the x-axis as shown in Figure 4.7a. To give the optimization a unique direction, A is the point with $q_{A_0,2} > 0$ according to Definition 13 and B is its mirror image, which the optimization aims at. Points on the x-axis form no pair, since they coincide with their mirror image. For these pairs, Theorem 1 gives the centre configurations from Definition 14.

Definition 14 (Centre Configuration)**.** *A centre configuration is one of the configurations* $\boldsymbol{c}_m \in \{(0,0),(0,-\pi),(-\pi,0),(-\pi,-\pi)\}$ *from Theorem 1, around which an oscillation between axis-symmetric points A and B is point-symmetric in joint space. For all four,* $\sin q_1 = 0$ *and* $\sin(q_1 + q_2) = 0$ *hold, so that the first row of the Jacobian* (3.4) *vanishes. They are thus singularities* (3.5)*, in which the arm is fully extended or folded back along the x-axis.*

Section 4.3.3 represents the configuration $\boldsymbol{q}_{B_i}$ relative to a centre configuration with $\boldsymbol{q}_{\text{ref}} = \boldsymbol{c}_m$ in Definition 10, while Section 4.3.4 represents $\boldsymbol{q}_{B_i}$ relative to the target configuration of the chosen initial guess. The configuration of A then follows directly as the reflected point $\boldsymbol{q}_{A_i} = 2\boldsymbol{c}_m - \boldsymbol{q}_{B_i} = \boldsymbol{c}_m - \boldsymbol{\delta}$ from (4.18) as shown in Figure 4.7b, where $\boldsymbol{\delta} = \boldsymbol{q}_{B_i} - \boldsymbol{c}_m$.

Initially, the design approach for the optimization was adopted as an educated guess and subsequently justified by the derivation in Section 4.3.2. As displayed in Figure 4.7a, we



(a) Cartesian space (b) Joint space

Figure 4.7.: Joint-space point-symmetric oscillation between the points A and B mirrored about the x-axis for $\boldsymbol{c}_m = (0, 0)$ (AI-generated illustration)

position the robot base in the origin of the cartesian coordinate system. With all CoMs of our robot model on the link axes in the top view of Figure 4.2 as in the general robot model in Section 3.2, the mass matrix $\boldsymbol{M}(\boldsymbol{q})$ depends only on c_2, $\boldsymbol{\Phi}(\dot{\boldsymbol{q}})$ is even for $\dot{\boldsymbol{q}}$ and $h(\boldsymbol{q}) = m_2 l_1 l_{c,2} s_2$ is proportional to s_2, where m_2 and $l_{c,2}$ belong to the combined rigid body of link 2 and the gripper. As the proofs in Sections 4.3.2 and 4.3.2 rely on exactly these properties, they still hold for our modelled robot in Section 4.1. Since we additionally create a potential field with the clutch mechanism introduced in Section 4.1 such that the assumptions of Theorem 1 are met, a trajectory starting in a centre configuration $\boldsymbol{c}_m$ and stopping in $\boldsymbol{q}_{B_0}$ or $\boldsymbol{q}_{B_1}$ also stops in $\boldsymbol{q}_{A_0}$ or $\boldsymbol{q}_{A_1}$ respectively. The optimization therefore only has to refine stage one with the starting configuration $\boldsymbol{c}_m$, use its successful result as initial guess and then search for trajectories stopping in the chosen joint configuration of the cartesian point B. This halves the criteria we optimize for, since we do not have to additionally optimize for a stop in A. The resulting oscillation is point-symmetric around $\boldsymbol{c}_m$ in joint space as shown in Figure 4.7b.

In general, an oscillation from $\boldsymbol{c}_m = \boldsymbol{q}(0)$ to B and on to A is also one from $\boldsymbol{c}_m = \boldsymbol{q}(\frac{T}{2})$ to A and on to B, since the oscillation $\boldsymbol{q}(t)$ returns to $\boldsymbol{c}_m$ after $\frac{T}{2}$ (Section 3.2.3). Only the initial velocity vector in $\boldsymbol{c}_m$ is negated as shown in Figure 4.7.

Conditions for Point-Symmetric Oscillations in Joint Space

Theorem 1 (Joint-Space Point-Symmetric Oscillations between Axis-Symmetric Task Positions)**.** *Let the planar 2-DoF robot from Section 3.1 have the forward kinematics* (3.1) *and the dynamics of the conservative system* (4.3) *with the terms* (3.20) *and* (3.21)*, let the Cartesian points $\boldsymbol{p}_A$ and $\boldsymbol{p}_B$ be axis-symmetric with respect to the x-axis and let the potential field be symmetric around a centre configuration $\boldsymbol{c}_m \in \{(0,0), (0,-\pi), (-\pi,0), (-\pi,-\pi)\}$. If a trajectory $\boldsymbol{q}(t)$ of* (4.3) *with $\boldsymbol{q}(0) = \boldsymbol{c}_m$ comes to rest at $\boldsymbol{p}_B$ in the configuration $\boldsymbol{q}_{B_i}$ with $i \in \{0,1\}$, then $\boldsymbol{q}(t)$ is a point-symmetric oscillation around $\boldsymbol{c}_m$ that also comes to rest at $\boldsymbol{p}_A$ in the configuration $\boldsymbol{q}_{A_i}$.*

The proof of Theorem 1 builds on two lemmas. Using the terms (3.20) and (3.21), Lemma 1 gives the second rest configuration $2\boldsymbol{c}_m - \boldsymbol{q}_B$ of the point-symmetric oscillation. Lemma 2 then shows with the forward kinematics (3.1) that this configuration belongs to the x-axis-symmetric Cartesian point.

Lemma 1 (Conditions for Joint-Space Point-Symmetric Oscillations)**.** *Let* $\boldsymbol{q}(t)$ *with* $\boldsymbol{c}_m = \boldsymbol{q}(0)$ *be a trajectory of the conservative system* (4.3) *with the terms* (3.20) *and* (3.21) *of the planar 2-DoF robot in Section 3.1, let the potential field be symmetric around* $\boldsymbol{c}_m$ *and let the centre configuration fulfill* $c_{m,2} = n_{c,2}\pi$ *with* $n_{c,2} \in \mathbb{Z}$*. If* $\boldsymbol{q}(t)$ *comes to rest in a joint configuration* $\boldsymbol{q}_B$ *at an instant* $t_B \neq 0$*, then it also comes to rest in* $\boldsymbol{q}_A = 2\boldsymbol{c}_m - \boldsymbol{q}_B$ *at the instant* $-t_B$*. With these two rest configurations,* $\boldsymbol{q}(t)$ *is an oscillation [83] that is point-symmetric around* $\boldsymbol{c}_m$ *as given in* (3.33)*.*

Proof of Lemma 1. We show that (i) $\boldsymbol{q}_T(t) = \boldsymbol{q}(-t)$ solves the SCARA dynamics and determine (ii) on which conditions $\boldsymbol{q}_I(t) = 2\boldsymbol{c}_m - \boldsymbol{q}(t)$ solves the manipulator dynamics. We then proceed in step (iii) by showing that both $\boldsymbol{q}_T(t)$ and $\boldsymbol{q}_I(t)$ have the same initial state at $t = 0$ and therefore prove that $\boldsymbol{q}_T(t) = \boldsymbol{q}_I(t) = 2\boldsymbol{c}_m - \boldsymbol{q}(t)$ $\forall t$ holds.

(i) The time-reversed trajectory $\boldsymbol{q}_T(t)$ must solve the same dynamic manipulator equation as $\boldsymbol{q}(t)$. For the following derivation we use the Coriolis and centrifugal vector from (3.21) in its split form (3.22).

Inserting $\boldsymbol{q}_T(t)$ and its derivatives calculated in (4.4) into (4.3)

$$\begin{aligned} \boldsymbol{q}_T(t) &= \boldsymbol{q}(-t)\,, \\ \dot{\boldsymbol{q}}_T(t) &= \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{q}(-t) = -\dot{\boldsymbol{q}}(-t)\,, \\ \ddot{\boldsymbol{q}}_T(t) &= \frac{\mathrm{d}}{\mathrm{d}t}\big(-\dot{\boldsymbol{q}}(-t)\big) = \ddot{\boldsymbol{q}}(-t) \end{aligned} \tag{4.4}$$

and using that $\boldsymbol{\Phi}$ is even in $\dot{\boldsymbol{q}}$ (3.23), we see in (4.5) that $\boldsymbol{q}_T(t)$ solves the same SCARA manipulator dynamics as $\boldsymbol{q}(-t)$.

$$\begin{aligned} \boldsymbol{M}(\boldsymbol{q}_T(t))\ddot{\boldsymbol{q}}_T(t) &= \boldsymbol{M}(\boldsymbol{q}(-t))\ddot{\boldsymbol{q}}(-t)\,, \\ h(\boldsymbol{q}_T(t))\boldsymbol{\Phi}(\dot{\boldsymbol{q}}_T(t)) &= h(\boldsymbol{q}(-t))\boldsymbol{\Phi}(-\dot{\boldsymbol{q}}(-t)) = h(\boldsymbol{q}(-t))\boldsymbol{\Phi}(\dot{\boldsymbol{q}}(-t))\,, \\ \boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{q}_T(t)) &= \boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{q}(-t)) \end{aligned} \tag{4.5}$$

(ii) We write each point of the trajectory as the offset to the centre of point symmetry, $\boldsymbol{\delta}(t) = \boldsymbol{q}(t) - \boldsymbol{c}_m$, so that $\boldsymbol{q}(t) = \boldsymbol{c}_m + \boldsymbol{\delta}(t)$. The point-reflected trajectory then reads as (4.6).

$$\boldsymbol{q}_I(t) = 2\boldsymbol{c}_m - \boldsymbol{q}(t) = \boldsymbol{c}_m - \boldsymbol{\delta}(t) \tag{4.6}$$

Since $\boldsymbol{q}_I(t)$ depends only on $\boldsymbol{q}(t)$, its position, velocity and acceleration follow from the corresponding state vectors of $\boldsymbol{q}$ at the same time t, as calculated in (4.7).

$$\begin{aligned} \boldsymbol{q}_I &= \boldsymbol{c}_m - \boldsymbol{\delta} = 2\boldsymbol{c}_m - \boldsymbol{q}\,, \\ \dot{\boldsymbol{q}}_I &= -\dot{\boldsymbol{q}}\,, \\ \ddot{\boldsymbol{q}}_I &= -\ddot{\boldsymbol{q}} \end{aligned} \tag{4.7}$$

For $\boldsymbol{q}_I$ to solve the same dynamics as $\boldsymbol{q}_T$ at an arbitrary time t, it has to fulfill (4.8a). Replacing its joint configurations with the formulas from (4.7) gives (4.8b) and multiplying each side with -1 as well as using (3.23) results in (4.8c).

$$\boldsymbol{M}(\boldsymbol{q}_I)\ddot{\boldsymbol{q}}_I + h(\boldsymbol{q}_I)\boldsymbol{\Phi}(\dot{\boldsymbol{q}}_I) = \boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{q}_I) = -\frac{\partial \boldsymbol{V}(\boldsymbol{q}_I)}{\partial \boldsymbol{q}_I}\,, \tag{4.8a}$$

$$-\boldsymbol{M}(\boldsymbol{c}_m - \boldsymbol{\delta})\ddot{\boldsymbol{q}} + h(\boldsymbol{c}_m - \boldsymbol{\delta})\boldsymbol{\Phi}(-\dot{\boldsymbol{q}}) = \boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{c}_m - \boldsymbol{\delta}) = -\frac{\partial \boldsymbol{V}(\boldsymbol{c}_m - \boldsymbol{\delta})}{\partial(\boldsymbol{c}_m - \boldsymbol{\delta})}\,, \tag{4.8b}$$

$$\boldsymbol{M}(\boldsymbol{c}_m - \boldsymbol{\delta})\ddot{\boldsymbol{q}} - h(\boldsymbol{c}_m - \boldsymbol{\delta})\boldsymbol{\Phi}(\dot{\boldsymbol{q}}) = -\boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{c}_m - \boldsymbol{\delta}) = \frac{\partial \boldsymbol{V}(\boldsymbol{c}_m - \boldsymbol{\delta})}{\partial(\boldsymbol{c}_m - \boldsymbol{\delta})}\,, \tag{4.8c}$$

$$\boldsymbol{M}(\boldsymbol{c}_m + \boldsymbol{\delta})\ddot{\boldsymbol{q}} + h(\boldsymbol{c}_m + \boldsymbol{\delta})\boldsymbol{\Phi}(\dot{\boldsymbol{q}}) = \boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{c}_m + \boldsymbol{\delta}) = -\frac{\partial \boldsymbol{V}(\boldsymbol{c}_m + \boldsymbol{\delta})}{\partial(\boldsymbol{c}_m + \boldsymbol{\delta})} \tag{4.8d}$$

Comparing (4.8c) with (4.8d) term by term gives the three conditions in (4.9). Condition (4.9b) and the symmetric potential field behind (4.9a) correspond to the even symmetry (3.32) in Section 3.2.3[83], which we derived here from the equations of motion.

$$\boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{c}_m - \boldsymbol{\delta}) = -\boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{c}_m + \boldsymbol{\delta})\,, \tag{4.9a}$$

$$\boldsymbol{M}(\boldsymbol{c}_m - \boldsymbol{\delta}) = \boldsymbol{M}(\boldsymbol{c}_m + \boldsymbol{\delta})\,, \tag{4.9b}$$

$$h(\boldsymbol{c}_m - \boldsymbol{\delta}) = -h(\boldsymbol{c}_m + \boldsymbol{\delta}) \tag{4.9c}$$

The spring torque in (4.8) comes from the potential field assumed to be symmetric around $\boldsymbol{c}_m$ in Lemma 1, which is characterized as (4.10a). In order to show that a symmetric potential field satisfies condition (4.9a), we derive both sides with respect to $\boldsymbol{q}$ getting (4.10c). When applying (4.10b) to (4.10c) we get (4.10d) which in turn proofs that (4.9a) is fulfilled when the potential field is symmetric since $\boldsymbol{q} = \boldsymbol{c}_m + \boldsymbol{\delta}$ holds per definition.

$$\boldsymbol{V}(2\boldsymbol{c}_m - \boldsymbol{q}) = \boldsymbol{V}(\boldsymbol{q})\ \forall \boldsymbol{q}\,, \tag{4.10a}$$

$$\boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{q}) = -\left.\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{q}}\right|_{\boldsymbol{q}}\,, \tag{4.10b}$$

$$-\left.\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{q}}\right|_{2\boldsymbol{c}_m - \boldsymbol{q}} = \left.\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{q}}\right|_{\boldsymbol{q}}\,, \tag{4.10c}$$

$$\boldsymbol{\tau}_{\boldsymbol{spring}}(2\boldsymbol{c}_m - \boldsymbol{q}) = -\boldsymbol{\tau}_{\boldsymbol{spring}}(\boldsymbol{q}) \tag{4.10d}$$

When looking into condition (4.9b), we look at the mass matrix derived in (3.20) and see that the only term depending on $\boldsymbol{q}$ is $c_2 = \cos(q_2)$. This means that we only need to look at the condition (4.11) and solve for the condition on q_2 by applying the addition theorem $\cos(c_{m,2} \mp \delta_2) = \cos(c_{m,2})\cos(\delta_2) \pm \sin(c_{m,2})\sin(\delta_2)$ [21, s.v. compound angle formulae].

$$\begin{aligned}
\cos(c_{m,2} - \delta_2) &= \cos(c_{m,2} + \delta_2)\,,\\
\cos(c_{m,2})\cos(\delta_2) + \sin(c_{m,2})\sin(\delta_2) &= \cos(c_{m,2})\cos(\delta_2) - \sin(c_{m,2})\sin(\delta_2)\,,\\
\sin(c_{m,2})\sin(\delta_2) &= -\sin(c_{m,2})\sin(\delta_2)\,,\\
0 &= 2\sin(c_{m,2})\sin(\delta_2)
\end{aligned} \tag{4.11}$$

The last equation in (4.11) can only be zero if $\sin(\delta_2) = \sin(q_2 - c_{m,2}) = 0$ or if $\sin(c_{m,2}) = 0$. This means the condition is fulfilled when $c_{m,2} = n_{c,2}\pi$ with $n_{c,2} \in \mathbb{Z}$, or if $q_2(t) = c_{m,2} + n_M\pi\ \forall t$ with $n_M \in \mathbb{Z}$.

The last condition is (4.9c). As defined in (3.22), $h(\boldsymbol{q})$ is only dependent on the term $\sin(q_2)$, so we look at the condition in (4.12) and solve for the condition on q_2 by applying the addition theorem $\sin(c_{m,2} \mp \delta_2) = \sin(c_{m,2})\cos(\delta_2) \mp \cos(c_{m,2})\sin(\delta_2)$[21, s.v. compound angle formulae].

$$\begin{aligned} \sin(c_{m,2} - \delta_2) &= -\sin(c_{m,2} + \delta_2)\,, \\ \sin(c_{m,2})\cos(\delta_2) - \cos(c_{m,2})\sin(\delta_2) &= -\big(\sin(c_{m,2})\cos(\delta_2) + \cos(c_{m,2})\sin(\delta_2)\big)\,, \\ \sin(c_{m,2})\cos(\delta_2) &= -\sin(c_{m,2})\cos(\delta_2)\,, \\ 0 &= 2\sin(c_{m,2})\cos(\delta_2) \end{aligned} \tag{4.12}$$

The last equation in (4.12) can only be zero if $\cos(\delta_2) = \cos(q_2 - c_{m,2}) = 0$ or if $\sin(c_{m,2}) = 0$. This means the condition is fulfilled when $c_{m,2} = n_{c,2}\pi$ with $n_{c,2} \in \mathbb{Z}$, or if $q_2(t) = c_{m,2} + (2n_h + 1)\frac{\pi}{2}\ \forall t$ with $n_h \in \mathbb{Z}$.

Each of the two conditions (4.9b) and (4.9c) is fulfilled either by $\sin(c_{m,2}) = 0$ or by a condition on the trajectory that has to hold for any arbitrary time t. The trajectory condition of the mass matrix places $\delta_2 = q_2 - c_{m,2}$ on the multiples of π and the one of the centrifugal term on the odd multiples of $\frac{\pi}{2}$. These two sets are disjoint, so both trajectory conditions can never hold at the same time and the only remaining option is $\sin(c_{m,2}) = 0$. Since all three conditions (4.9a), (4.9b) and (4.9c) must hold simultaneously, a symmetric potential field is required and the point symmetry centre must satisfy (4.13).

$$c_{m,2} = n_{c,2}\pi\,, \quad n_{c,2} \in \mathbb{Z} \tag{4.13}$$

(iii) If the three conditions in (4.9) are fulfilled, both $\boldsymbol{q}_T(t)$ and $\boldsymbol{q}_I(t)$ solve the SCARA manipulator dynamics. At $t = 0$ they have the same configuration and the same velocity, which follows from (4.4) and (4.7) as shown in (4.14). The dynamic manipulator equation is locally Lipschitz continuous as shown in Section 3.3 and continuous at all times t. This holds for a continuously differentiable spring torque $\boldsymbol{\tau_{spring}} \in C^1$. The solution for a given initial state in (4.3) is then unique for all t[52, Theroem 3.2]. Both functions therefore coincide with $\boldsymbol{q}_T(t) = \boldsymbol{q}_I(t)\ \forall t$ and the trajectory is point-symmetric around $\boldsymbol{c}_m$.

$$\begin{aligned} \boldsymbol{q}_T(0) &= \boldsymbol{q}(0) = \boldsymbol{c}_m\,, \\ \boldsymbol{q}_I(0) &= 2\boldsymbol{c}_m - \boldsymbol{q}(0) = \boldsymbol{c}_m\,, \\ \dot{\boldsymbol{q}}_T(0) &= -\dot{\boldsymbol{q}}(0)\,, \\ \dot{\boldsymbol{q}}_I(0) &= -\dot{\boldsymbol{q}}(0) \end{aligned} \tag{4.14}$$

Since $\boldsymbol{q}_T(t) = \boldsymbol{q}_I(t)\ \forall t$ holds, their velocities and accelerations from (4.4) and (4.7) coincide as well. Due to (4.15c), the acceleration at $t = 0$ thus vanishes (4.15d). When we stop in $\boldsymbol{q}_B$ with $\dot{\boldsymbol{q}}(t_B) = \boldsymbol{0}$, (4.15a) and (4.15b) give $\boldsymbol{q}(-t_B) = 2\boldsymbol{c}_m - \boldsymbol{q}_B = \boldsymbol{q}_A$ and $\dot{\boldsymbol{q}}(-t_B) = \boldsymbol{0}$. The trajectory therefore also comes to rest in $\boldsymbol{q}_A$ as stated in Lemma 1. With the two rest configurations $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$, it is an oscillation[83].

$$\boldsymbol{q}(-t) = 2\boldsymbol{c}_m - \boldsymbol{q}(t)\ \forall t\,, \tag{4.15a}$$
$$\dot{\boldsymbol{q}}(-t) = \dot{\boldsymbol{q}}(t)\ \forall t\,, \tag{4.15b}$$
$$\ddot{\boldsymbol{q}}(-t) = -\ddot{\boldsymbol{q}}(t)\ \forall t\,, \tag{4.15c}$$
$$\ddot{\boldsymbol{q}}(0) = \boldsymbol{0} \tag{4.15d}$$

□

Lemma 2 (Mirrored Configuration)**.** *Let the robot base lie on the x-axis. If the forward kinematics of* $\boldsymbol{q}$ *gives the Cartesian point* (x, y)*, then the forward kinematics of* $-\boldsymbol{q}$ *gives its mirror image* $(x', y') = (x, -y)$*.*

Proof of Lemma 2. When inserting $-\boldsymbol{q} = (-q_1, -q_2)$ into the forward kinematics (3.1) of the SCARA robot in (4.16), we can reformulate the equation with the symmetry properties $\cos(-\theta) = \cos(\theta)$ and $\sin(-\theta) = -\sin(\theta)$ [21, s.v. trigonometric function] and prove that $x' = x$ and $y' = -y$.

$$\begin{aligned} x' &= l_1 \cos(-q_1) + l_2 \cos(-q_1 - q_2) = l_1 \cos(-q_1) + l_2 \cos(-(q_1 + q_2)) \\ &= l_1 \cos(q_1) + l_2 \cos(q_1 + q_2) = x\,, \\ y' &= l_1 \sin(-q_1) + l_2 \sin(-q_1 - q_2) = l_1 \sin(-q_1) + l_2 \sin(-(q_1 + q_2)) \\ &= -\big(l_1 \sin(q_1) + l_2 \sin(q_1 + q_2)\big) = -y \end{aligned} \tag{4.16}$$

□

Proof of Theorem 1. By Lemma 2 the configuration $-\boldsymbol{q}_{B_i}$ gives $\boldsymbol{p}_A$ since $\boldsymbol{p}_A$ and $\boldsymbol{p}_B$ are x-axis-symmetric. As the forward kinematics (3.1) are 2π-periodic in each joint angle, every configuration $-\boldsymbol{q}_{B_i} + 2\pi\boldsymbol{n}_c$ with $\boldsymbol{n}_c \in \mathbb{Z}^2$ gives $\boldsymbol{p}_A$ as well. The midpoints between $\boldsymbol{q}_{B_i}$ and these configurations are given in (4.17).

$$\boldsymbol{c}_m = \frac{\boldsymbol{q}_{B_i} + \big(-\boldsymbol{q}_{B_i} + 2\pi\boldsymbol{n}_c\big)}{2} = \pi\boldsymbol{n}_c = (n_{c,1}\pi, n_{c,2}\pi)\,, \quad n_{c,1}, n_{c,2} \in \mathbb{Z} \tag{4.17}$$

With $\boldsymbol{c}_m$ uniquely represented by $\boldsymbol{q}_{ref} = (0, 0)$ in Definition 10, (4.17) gives exactly the four centre configurations $\boldsymbol{c}_m \in \{(0,0), (0,-\pi), (-\pi, 0), (-\pi, -\pi)\}$ of Theorem 1. Since all of them fulfill (4.13) and the potential field is symmetric around $\boldsymbol{c}_m$, Lemma 1 applies to $\boldsymbol{q}(t)$ with $\boldsymbol{q}_B = \boldsymbol{q}_{B_i}$. The trajectory is therefore a point-symmetric oscillation around $\boldsymbol{c}_m$ that also comes to rest in the configuration $\boldsymbol{q}_A$ given in (4.18).

$$\boldsymbol{q}_A = 2\boldsymbol{c}_m - \boldsymbol{q}_{B_i} \tag{4.18}$$

Represented in the interval of Definition 10, its second joint angle has the opposite sign of $q_{B_i,2}$. With the sign convention of Definition 13, $\boldsymbol{q}_A$ is therefore the configuration $\boldsymbol{q}_{A_i}$ in point A_i. In a singularity only $\boldsymbol{q}_{A_0}$ and $\boldsymbol{q}_{B_0}$ exist and $\boldsymbol{q}_A = \boldsymbol{q}_{A_0}$ follows directly.

□

Workspace Reduction

To find the Cartesian point pairs with point-symmetric oscillations in joint space, we sample the workspace. Since an oscillation to B and on to A is also an oscillation to A and on to B (Section 4.3.2), it suffices to sample the points A and thus half of the workspace. The rotational invariance about q_1 in Lemma 3 halves the sampled region once more.

Lemma 3 (Rotational Invariance)**.** *Let the planar 2-DoF robot from Section 3.1 have the forward kinematics* (3.1) *and the dynamics of the conservative system* (4.3) *with the terms* (3.20) *and* (3.21)*. Let* $\boldsymbol{q}(t)$ *solve the manipulator dynamics with the potential field* $\boldsymbol{V}$*. Then* $\boldsymbol{q}'(t) = \boldsymbol{q}(t) + (\psi, 0)$ *solves the manipulator dynamics with the rotated potential field* $\boldsymbol{V}'$ *in* (4.20a)*.*

Proof of Lemma 3. Inserting the rotated trajectory into the manipulator dynamics gives (4.19a). The mass matrix depends on $\boldsymbol{q}$ only through $\cos(q_2)$ as used in (4.11) and $h(\boldsymbol{q})$ only through $\sin(q_2)$ as given in (3.22), so a constant shift of q_1 leaves the left-hand side unchanged and (4.19a) becomes (4.19b). Comparing it with (4.19c) leaves (4.19d) as the condition on the rotated spring torque.

$$\boldsymbol{M}(\boldsymbol{q}')\ddot{\boldsymbol{q}}' + h(\boldsymbol{q}')\boldsymbol{\Phi}(\dot{\boldsymbol{q}}') = \boldsymbol{\tau}_{spring}{}'(\boldsymbol{q}')\,, \tag{4.19a}$$

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + h(\boldsymbol{q})\boldsymbol{\Phi}(\dot{\boldsymbol{q}}) = \boldsymbol{\tau}_{spring}{}'\big(\boldsymbol{q} + (\psi, 0)\big)\,, \tag{4.19b}$$

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + h(\boldsymbol{q})\boldsymbol{\Phi}(\dot{\boldsymbol{q}}) = \boldsymbol{\tau}_{spring}(\boldsymbol{q})\,, \tag{4.19c}$$

$$\boldsymbol{\tau}_{spring}{}'\big(\boldsymbol{q} + (\psi, 0)\big) = \boldsymbol{\tau}_{spring}(\boldsymbol{q}) \tag{4.19d}$$

In order to show that the rotated potential field fulfills (4.19d), we derive the rotated spring torque from its representation (4.20a) and get (4.20b). Substituting the rotated configuration then gives (4.20c) proving the lemma.

$$\boldsymbol{V}'(\boldsymbol{q}) = \boldsymbol{V}\big(\boldsymbol{q} - (\psi, 0)\big)\,, \tag{4.20a}$$

$$\boldsymbol{\tau}_{spring}{}'(\boldsymbol{q}) = -\left.\frac{\partial \boldsymbol{V}'}{\partial \boldsymbol{q}}\right|_{\boldsymbol{q}} = -\left.\frac{\partial \boldsymbol{V}}{\partial \boldsymbol{q}}\right|_{\boldsymbol{q}-(\psi,0)} = \boldsymbol{\tau}_{spring}\big(\boldsymbol{q} - (\psi, 0)\big)\,, \tag{4.20b}$$

$$\boldsymbol{\tau}_{spring}{}'\big(\boldsymbol{q} + (\psi, 0)\big) = \boldsymbol{\tau}_{spring}\big(\boldsymbol{q} + (\psi, 0) - (\psi, 0)\big) = \boldsymbol{\tau}_{spring}(\boldsymbol{q}) \tag{4.20c}$$

□

For $\psi = \pi$ the rotation maps the x-axis onto itself. The rotated points $\boldsymbol{p}'_A = (-p_{A,x}, -p_{A,y})$ and $\boldsymbol{p}'_B = (-p_{A,x}, p_{A,y})$ are therefore again axis-symmetric with respect to the x-axis and $\boldsymbol{c}'_m = \boldsymbol{c}_m + (\pi, 0)$ is again one of the four centre configurations of Theorem 1. A point-symmetric oscillation around $\boldsymbol{c}_m$ thus gives a point-symmetric oscillation around $\boldsymbol{c}'_m$ with the same joint velocities. With $p_{A,x}p_{A,y} > 0$, the sign of $p_x p_y$ is unchanged after rotation. Since additionally q_2 stays the same, the configuration labels of Definition 13 carry over to the rotated points. Points A on the positive y-axis are not rotated, since they are already part of the sampled region. Sampling the points A with $p_{A,x} \geqslant 0$ and $p_{A,y} > 0$, a quarter of the workspace, therefore yields oscillations of the whole workspace.

4.3.3. Stiffness Optimization

The second stage solves the reduced problem from Section 4.3.2 and additionally optimizes the stiffness values of the torsional springs introduced in Section 4.1. These values stay constant along each trajectory and differ only between different results. Optimizing the stiffness values gives additional degrees of freedom to find point-symmetric oscillations.

Only after analysing the occurrence of these oscillations across the workspace and the necessary stiffness values can we select one suitable fixed stiffness per spring in Section 4.3.4. For this analysis, it suffices to sample a quarter of the workspace as derived in Section 4.3.2.

4.3.4. Clutch Switching Times Optimization

The last stage no longer optimizes the stiffness values. It fixes one reasonably good stiffness per spring chosen based on the successful results of Section 4.3.3 and examines how far the clutches can adapt the oscillations to different tasks. The remaining freedom of the mechanism then lies

in the times at which each of the two clutches engages and disengages, independently of the other. As in the second stage, we search for point-symmetric oscillations in joint space between the axis-symmetric points A and B under the conditions of Section 4.3.2.

To see for which pick and place positions such oscillations can be found, we sample the workspace as a grid of Cartesian positions for A with their mirror images B. Since sharing the configuration pair and $\boldsymbol{c}_m$ most likely yields similar oscillations and Section 4.4 needs the shared centre configuration $\boldsymbol{c}_m$ to transition between the found oscillations, we define a raster (Definition 15).

Definition 15 (Raster)**.** *A raster consists of a grid of Cartesian positions for A and their mirror images B, where the optimization results aim for oscillations starting in a specified centre configuration $\boldsymbol{c}_m$ and stopping in A_i and B_i with a specified index i (Definition 13).*

Each raster yields raster nodes (Definition 16).

Definition 16 (Raster Node)**.** *A raster node n_{raster} is defined by the Cartesian position of A, also called raster position. Additionally, it is identified by the configuration index i from Definition 13 and the centre configuration $\boldsymbol{c}_m$. Its members $s_{raster} \in n_{raster}$ are this stage's optimization results in joint space around $\boldsymbol{c}_m$ between $\boldsymbol{q}_{A_i}$ and $\boldsymbol{q}_{B_i}$. A successful member is the successful result of this stage, namely the point-symmetric oscillation in joint space.*

Such a raster node can have several members or none. With the two configuration indexes and the four centre configurations $\boldsymbol{c}_m$ in Theorem 1, there are in total eight different rasters for which raster node's members are searched for.

4.4. Conservative Pick-and-Place Sequence

Following the problem statement in Section 1.1 and the motor's role in Section 4.2.2, this section considers the execution of a conservative pick-and-place sequence. Section 4.4.1 defines the transition cost (Definition 17) of the impulsive transition at the shared equilibrium $\boldsymbol{c}_m$ and Section 4.4.2 applies this cost to an example task set of a 3×3 grid of Cartesian points A with their counterparts B mirrored about the x-axis. In such a task sequence, we define B as the pick position and A as the place position. Starting at the centre configuration $\boldsymbol{c}_m$, the robot swings to B to pick up the object and then through $\boldsymbol{c}_m$ to A to place it. When it passes $\boldsymbol{c}_m$ again toward B, the motor transitions to the oscillation of the next task, which picks up the next object at its point B.

4.4.1. Transition Cost

We want to define the transition cost between two pick-and-place motions fulfilling two different pick-and-place tasks. These motions are in our case the point-symmetric oscillations in joint space from Section 4.3.2. Since the only criterion for the impulsive transition is that the oscillations intersect, one can transition between two tasks with different configuration indexes. The point where the oscillations intersect is the centre configuration $\boldsymbol{c}_m$, which is also the shared equilibrium of the oscillations.

Figure 4.8 displays the two oscillations between which the motor performs the impulsive transition at $\boldsymbol{c}_m$ with $\Delta t = t_2 - t_1 \to 0$. After its period T, the current oscillation returns to $\boldsymbol{c}_m = \boldsymbol{q}_{curr}(0) = \boldsymbol{q}_{curr}(T)$ with its initial velocity $\dot{\boldsymbol{q}}_{curr}(T) = \dot{\boldsymbol{q}}_{curr}(0)$ toward $\boldsymbol{q}_{B,curr}$ (Section 3.2.3). The transition then starts the next oscillation from $\boldsymbol{c}_m = \boldsymbol{q}_{next}(0)$ with the initial velocity $\dot{\boldsymbol{q}}_{next}(0)$ toward $\boldsymbol{q}_{B,next}$ as shown in Figure 4.8.

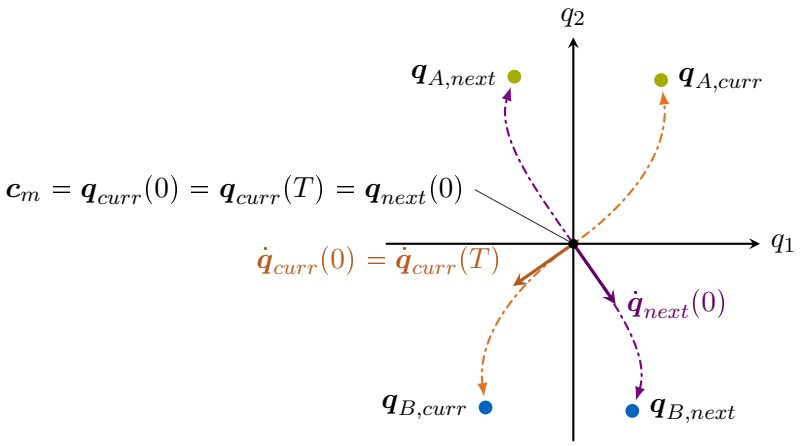


Figure 4.8.: Point-symmetric oscillations of the current (orange) and the next task (violet) in joint space with their shared centre configuration $\boldsymbol{c}_m = (0, 0)$ (AI-generated illustration)

The torque pulse of the impulsive transition has to redirect the joint velocity at $\boldsymbol{c}_m$ from $\dot{\boldsymbol{q}}_{curr}(T)$ to $\dot{\boldsymbol{q}}_{next}(0)$. Since the transition cost should rate the contribution of the motors, we do not consider the change of the joint velocity but the difference $\Delta\boldsymbol{\rho}$ of the momentum (3.29) between the current and the next oscillation. For $\Delta t \to 0$, the joint state stays approximately at $\boldsymbol{c}_m$ during the pulse and $\Delta\boldsymbol{\rho}$ approximately equals the time integral of the motor torque $\boldsymbol{\tau}_m$ over the pulse in (4.21)[32, Section 11.7].

$$\Delta\boldsymbol{\rho} \approx \int_{t_1}^{t_2} \boldsymbol{\tau}_m(t)\,\mathrm{d}t \tag{4.21}$$

Each component of $\Delta\boldsymbol{\rho}$ thus directly gives the torque impulse of one motor and displays its contribution. Inserting (3.29) into the momentum difference $\Delta\boldsymbol{\rho} = \boldsymbol{\rho}(t_2) - \boldsymbol{\rho}(t_1)$ gives (4.22).

$$\Delta\boldsymbol{\rho} = \boldsymbol{\rho}(t_2) - \boldsymbol{\rho}(t_1) = \boldsymbol{M}(\boldsymbol{q}(t_2))\,\dot{\boldsymbol{q}}(t_2) - \boldsymbol{M}(\boldsymbol{q}(t_1))\,\dot{\boldsymbol{q}}(t_1) \tag{4.22}$$

Since the joint state stays at $\boldsymbol{c}_m$, the mass matrix $\boldsymbol{M}(\boldsymbol{c}_m)$ is constant and $\Delta\boldsymbol{\rho}$ in (4.23) only depends on the joint velocities, defined by (3.27) and (3.28).

$$\Delta\boldsymbol{\rho} = \boldsymbol{M}(\boldsymbol{c}_m)\big(\dot{\boldsymbol{q}}(t_2) - \dot{\boldsymbol{q}}(t_1)\big) \tag{4.23}$$

As the disengaged clutches store no potential energy at $\boldsymbol{c}_m$, the kinetic energy there equals the total energy E_{total} for all oscillations (Section 4.3). Only the angles α_{curr} of the current and α_{next} of the next task remain in (4.24) and thus define the value of the momentum difference.

$$\begin{aligned}\Delta\boldsymbol{\rho} = \boldsymbol{M}(\boldsymbol{c}_m)\Bigg(&\sqrt{\frac{2E_{total}}{\boldsymbol{d}^\top(\alpha_{next})\,\boldsymbol{M}(\boldsymbol{c}_m)\,\boldsymbol{d}(\alpha_{next})}}\,\boldsymbol{d}(\alpha_{next}) \\ &- \sqrt{\frac{2E_{total}}{\boldsymbol{d}^\top(\alpha_{curr})\,\boldsymbol{M}(\boldsymbol{c}_m)\,\boldsymbol{d}(\alpha_{curr})}}\,\boldsymbol{d}(\alpha_{curr})\Bigg)\end{aligned} \tag{4.24}$$

Since the kinetic energy equals E_{total} before and after the pulse within the conservative system, a regenerative drive without losses only redistributes the energy between the joints to match the momentum of the next task. The transition is thus conservative for every momentum difference.

The transition cost in Definition 17 can be calculated with the momentum difference (4.24) where the 1-norm sums the magnitudes of the torque impulses of both motors in (4.25). It thus has a direct physical interpretation as the total torque impulse the motors apply for the transition.

Definition 17 (Transition Cost)**.** *1-norm[36, Section 2.2.1] of the instantaneous change of momentum of an impulsive transition*

$$\|\Delta\boldsymbol{\rho}\|_1 = |\Delta\rho_1| + |\Delta\rho_2| \tag{4.25}$$

4.4.2. 3×3 Task Set

As an example of a conservative pick-and-place sequence in the conservative system, we consider a 3×3 task set shown in Figure 4.9, which can be executed repeatedly. Its place positions A lie on a Cartesian 3×3 grid and its pick positions B on the grid mirrored about the x-axis. Each task of the task set consists of a place position A and its own mirror image B as pick position. We are searching for closed 3×3 pick-and-place sequences that execute the pick-and-place tasks of the 3×3 task set in arbitrary order. Since the motor delivers the impulsive transition at $\boldsymbol{c}_m$ (Section 4.4.1) and the oscillations performing the tasks intersect at a centre configuration, all oscillations considered for a sequence must share the same centre configuration $\boldsymbol{c}_m$. We therefore represent each task by a GTSP node of the GTSP in Section 3.5 that fixes $\boldsymbol{c}_m$ but not the configuration index (Definition 18).

Definition 18 (GTSP Node)**.** *A GTSP node n is defined by the Cartesian position of A and the centre configuration $\boldsymbol{c}_m$. Its members $s \in n$ are the point-symmetric oscillations in joint space around $\boldsymbol{c}_m$ between $\boldsymbol{q}_{A_i}$ and $\boldsymbol{q}_{B_i}$ of both configuration indexes i, where B is the mirror image of A about the x-axis.*

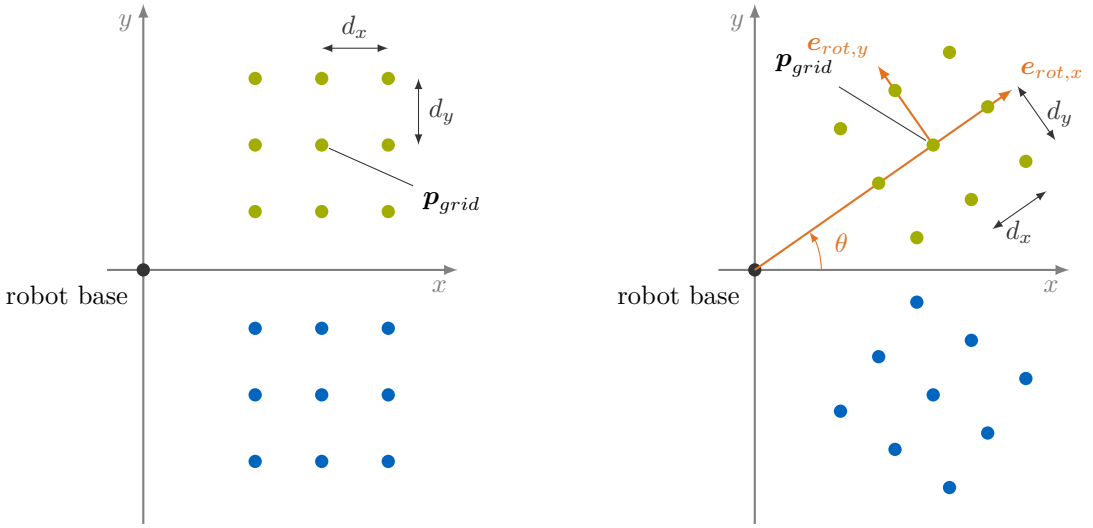
If A lies on a raster position, a GTSP node merges two raster nodes of Section 4.3.4. These raster nodes share the same raster position and centre configuration $\boldsymbol{c}_m$ but differ in the configuration index. The GTSP members $s \in n$ are then the successful members s_{raster} of the merged raster nodes. If A lies off the raster, there are no raster nodes and the members are searched for separately in Section 5.5.

Figure 4.9 shows the 3×3 task set for two placement modes. The different placements of the 3×3 task set are identified by the grid centre $\boldsymbol{p}_{grid}$ of its place positions. Each pick position is the counterpart of its place position mirrored about the x-axis. The pick B and place A positions thus fulfil the axis symmetry required by Theorem 1. Starting from the centre $\boldsymbol{p}_{grid}$, the other grid points of the place positions A lie at the distances $d_x = d_y = 0.06\,\mathrm{m}$ along two orthogonal axes $\boldsymbol{e}_{grid,x}$ and $\boldsymbol{e}_{grid,y}$. The nine grid points thus form the set $\mathcal{P}_{grid}$ in (4.26).

$$\mathcal{P}_{grid} = \left\{ \boldsymbol{p}_{grid} + u\,\boldsymbol{e}_{grid,x} + v\,\boldsymbol{e}_{grid,y} \;\middle|\; u \in \{-d_x, 0, d_x\},\ v \in \{-d_y, 0, d_y\} \right\} \tag{4.26}$$

Since $d_x = d_y$, the set $\mathcal{P}_{grid}$ of a grid rotated by 90° about its centre contains the same points and thus describes the same task set. In the axis-aligned placement in Figure 4.9a, $\boldsymbol{e}_{grid,x}$ and $\boldsymbol{e}_{grid,y}$ point along the x- and y-axis, so that the grid moves across the workspace with $\boldsymbol{p}_{grid}$ without rotating. In the radially aligned placement in Figure 4.9b, they are the axes $\boldsymbol{e}_{rot,x}$ and $\boldsymbol{e}_{rot,y}$ of a coordinate system rotated by the polar angle θ of the grid centre about the robot base at the origin in (4.27). Its axis $\boldsymbol{e}_{rot,x}$ thus points from the robot base towards the grid centre.

$$\boldsymbol{e}_{rot,x} = (\cos\theta, \sin\theta)^\top, \quad \boldsymbol{e}_{rot,y} = (-\sin\theta, \cos\theta)^\top \tag{4.27}$$



(a) Axis-Aligned Placement

(b) Radially Aligned Placement

Figure 4.9.: Placement modes of the 3×3 task set with the points A (green) and their mirror images B (blue) about the x-axis. (AI-generated illustration)

A task set placement with $\boldsymbol{p}_{grid}$ is only valid when all points of $\mathcal{P}_{grid}$ lie in the quarter workspace with $x \geqslant 0$ and $y > 0$, so that their mirror images B lie on the other side of the x-axis.

An inter-node link between members of two different GTSP nodes corresponds to a transition. Its link cost $\|\Delta\boldsymbol{\rho}\|_1$ is the transition cost (4.25) to minimize the motors' contribution to the sequence. As optimization method, we choose the GTSP from Section 3.5, which searches for the generalized Hamiltonian tour with the lowest sum of link costs. It determines the order in which the tasks of the task set are performed and which members of the GTSP nodes are selected as pick-and-place motions. Since we are searching for closed conservative pick-and-place sequences, the sequence cost (Definition 19) also includes the transition cost from the last to the first oscillation of the sequence.

Definition 19 (Sequence Cost)**.** *Sum of the transition costs of a conservative pick-and-place sequence. For a closed sequence, it includes the transition cost between the last and first motion.*

Since all motions of the closed sequence are oscillations and we transition between them after one period, the sequence duration is the sum of the period times T of the oscillations. For one placement, there might be several closed sequences with the same sequence cost for the same $\boldsymbol{c}_m$. In such a case, we choose the closed sequence with the shortest sequence duration since that is our point of interest in the research sub-question.

5 Implementation

We implement the methods introduced in Chapter 4 on the computer described in Table 5.1 with the software versions listed in Table 5.2.

Component	Specification
Processor	Intel Core i5-13500
Cores / threads	14 / 20
Memory	16 GB

Table 5.1.: Computer hardware

Software	Version
Operating system	Ubuntu 24.04.5 LTS
Linux kernel	7.0.0-28-generic
Python	3.12.3
MuJoCo [86]	3.3.7
NumPy [41]	1.26.4
SciPy [88]	1.11.4
pycma [40, 30]	4.4.4
pandas [61]	2.1.4
Matplotlib [45]	3.6.3

Table 5.2.: Installed software

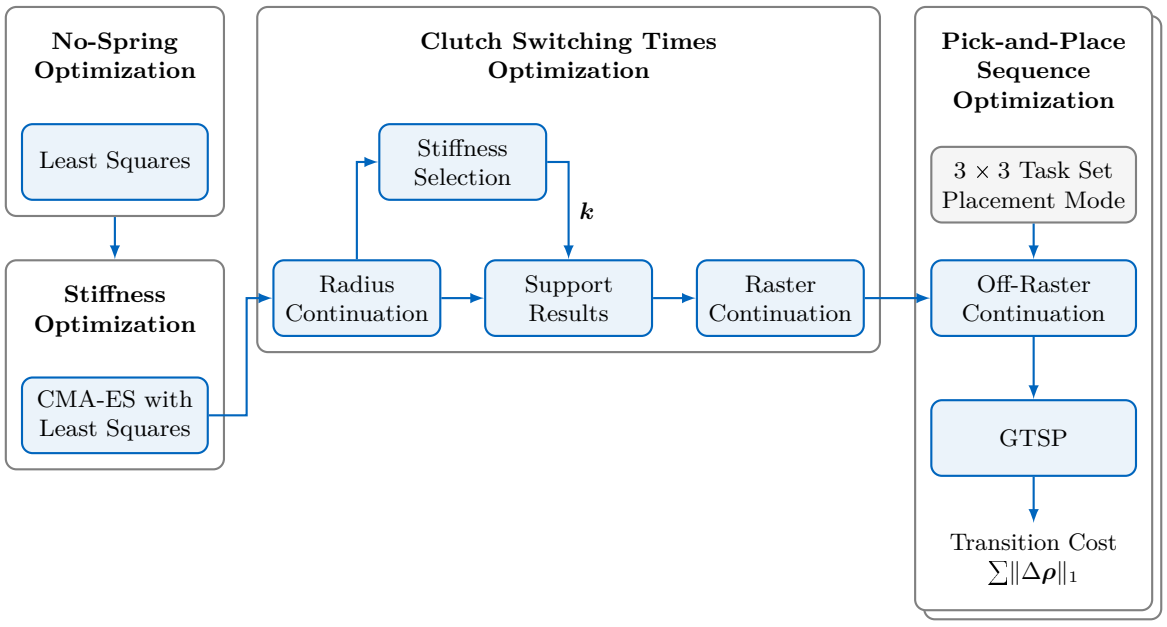


Figure 5.1.: Optimization pipeline from the no-spring optimization to the transition cost of the pick-and-place sequences (AI-generated illustration)

All optimizations in Figure 5.1 simulate the robot in MuJoCo as described in Section 5.1 and

use free-response shooting, a common method to find a path between two configurations[1]. Before each shot, the optimizer determines the system parameters, such as the stiffness values and the clutch switching events, as well as the initial velocity, calculated from the initial kinetic energy $E_{kin} = 5\,\mathrm{J}$ and the angle in joint space with (3.27) and (3.28). Figure 5.1 gives an overview of the implemented methods from Chapter 4 and the relations between the sections in this chapter.

In Section 5.2, the no-spring optimization implements the first stage from Section 4.3.1, where it optimizes to reach B and provides the initial guesses for the stiffness optimization in Section 5.3. The successful results of the stiffness optimization serve as a source for creating initial guesses in the clutch switching times optimization in Section 5.4. Section 5.5 then optimizes the necessary off-raster oscillations required by the radially aligned task set placement and calculates the sequence cost of the pick-and-place sequences.

5.1. MuJoCo Modelling

We model the SCARA robot dynamics in a conservative system with the proposed clutch mechanism from Section 4.1. The MuJoCo model described in Section 5.1.1 internally models the left-hand side of the previously defined manipulator equation (4.3) as $\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = 0$. It therefore only considers the dynamics of the SCARA robot without the clutch mechanism and models only the kinetic energy E_{kin} of the system. The spring torque and therefore the potential field and energy E_{pot} of the clutch mechanism are modelled separately as described in Section 5.1.2 and added as an external torque to the robot dynamics in MuJoCo. Section 5.1.3 discusses the numerical energy consistency $E_{total} = E_{pot} + E_{kin}$ of the complete system and our goal to keep it conservative with $E_{total} = 5\,\mathrm{J}$.

5.1.1. MJCF Model Description

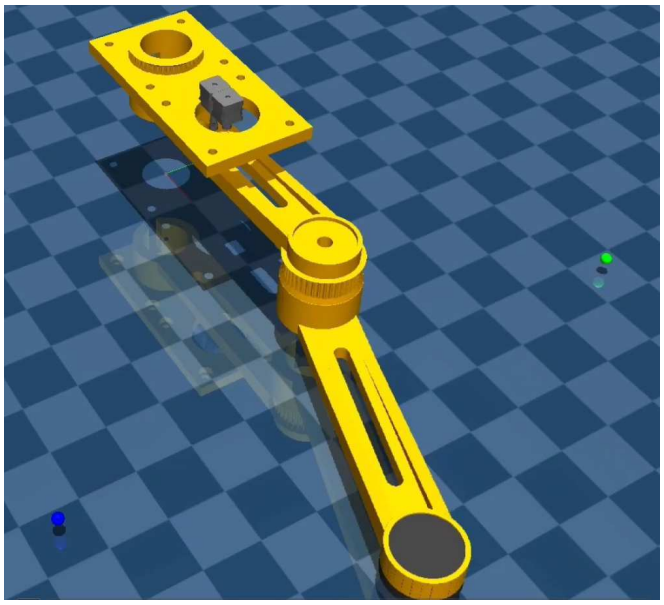


Figure 5.2.: MuJoCo model of the SCARA robot with the points A (green) and B (blue)

The MJCF model in Appendix A.1 contains the two links and the gripper of the SCARA robot from Figure 4.2, where both links are identical in length, mass and inertia. Figure 5.2 shows the

model in MuJoCo. Table 5.3 lists their parameters together with those of the gripper. We give the CoMs and the inertia tensors in the frame $\{i\}$ of link i from Figure 4.2. Since the gravity

Element	Parameter	Value
Link $i = 1, 2$	length l_i	$0.5\,\mathrm{m}$
	mass m_i	$2\,\mathrm{kg}$
	CoM in frame $\{i\}$	$(\approx 1.576\text{e-}1, \approx 2.689\text{e-}8, \approx 4.829\text{e-}2)^\top\,\mathrm{m}$
	inertia tensor I_i about the CoM	$\begin{bmatrix} \approx 3.82\text{e-}3 & \approx 8.948\text{e-}9 & \approx 6.234\text{e-}3 \\ \approx 8.948\text{e-}9 & \approx 6.662\text{e-}2 & \approx -8.842\text{e-}10 \\ \approx 6.234\text{e-}3 & \approx -8.842\text{e-}10 & \approx 6.629\text{e-}2 \end{bmatrix}\,\mathrm{kg\,m^2}$
Gripper	mass $m_{gripper}$	$0.6\,\mathrm{kg}$
	CoM in frame $\{2\}$	$(5\text{e-}1, 0, 1.5\text{e-}2)^\top\,\mathrm{m}$
	inertia tensor $I_{gripper}$ about the CoM	$\begin{bmatrix} \approx 7.25\text{e-}4 & 0 & 0 \\ 0 & \approx 7.25\text{e-}4 & 0 \\ 0 & 0 & \approx 1.08\text{e-}3 \end{bmatrix}\,\mathrm{kg\,m^2}$
Joint q_i	type	revolute
	damping, friction, armature	0
	range	unlimited
Simulation	time step dt	$10^{-3}\,\mathrm{s}$
	integrator	RK4
	contacts	disabled
	actuators	none

Table 5.3.: Parameters of the MuJoCo model

term vanishes for the horizontally moving manipulator (Section 4.2), setting gravity to zero in the MJCF model in Listing A.1 does not change the simulated dynamics. Contacts are disabled, since MuJoCo only models $\boldsymbol{M}(\boldsymbol{q})$ and $\boldsymbol{c}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in (4.3). The joints also have no damping, friction loss or armature. The model does not contain actuators, since our optimizations only consider the free response of the system. We do not simulate the impulsive transitions with the actuators but only calculate their cost. As Section 5.1.3 explains, we use RK4 as an integrator with the time step $dt = 10^{-3}\,\mathrm{s}$ as listed in Table 5.3.

5.1.2. Clutches

We neglect the actuation required to switch the clutch states introduced in Section 4.1 and the associated energy losses, assuming ideal switching so that the modelled system remains conservative. Only the spring torques and their corresponding potential field and energy E_{pot} are modelled and added to the dynamics of the robot. To keep the spring model open to later modifications, the springs are modelled externally and only the resulting torques are applied to the MuJoCo system as external torques. The clutch model remembers the joint

position $q_{\text{entry},i}$ at which clutch i engaged as the equilibrium of its spring (Section 4.1) until the clutch disengages again. As described in Sections 4.3.3 and 4.3.4, we are optimizing for point-symmetric oscillations in Sections 5.3 and 5.4 respectively and thus require the torque field to be point-symmetric around $\boldsymbol{c}_m$ in joint space. Between two switching events where we either engage or disengage the clutch, the spring torque is a well-defined function depending solely on the joint configuration $\boldsymbol{q}$ since $q_{\text{entry},i} = \text{const.}$ or $q_{\text{entry},i} = q_i$. We define a switching circle in joint space around $\boldsymbol{c}_m$ whose radius r_{c_i} is the joint-space distance from the circle to its centre $\boldsymbol{c}_m$ (Definition 11) and whose crossing triggers a switching event. Inside the circle with r_{c_i}, clutch i is disengaged and $\tau_{spring,i}(\boldsymbol{q}) = 0$ with $q_{\text{entry},i} = q_i$ holds. Outside the circle, a potential field acts and $q_{\text{entry},i}$ takes on the value $q_i(t_{\text{entry},i}) = q_{c_1,i}$ with $t_{\text{entry},i}$ being the time when the trajectory last crossed the respective circle from the inside to the outside at the crossing configuration $q_{c_1,i}$. Since the circle itself is point-symmetric around $\boldsymbol{c}_m$, the point-reflected trajectory $\boldsymbol{q}_I(t) = 2\boldsymbol{c}_m - \boldsymbol{q}(t)$ (3.31) of any trajectory $\boldsymbol{q}(t)$ has the reflected spring equilibria $2c_{m,i} - q_{\text{entry},i}$. Along the trajectory, the clutches therefore produce a potential field symmetric around $\boldsymbol{c}_m$ as assumed in Theorem 1 and Lemma 1. Figure 5.3 shows the first quarter of a point-symmetric oscillation, where the oscillation starts in $\boldsymbol{c}_m = (0, 0)$ and stops in $\boldsymbol{q}_{B_0}$. The dashed line shows the point-reflected trajectory $\boldsymbol{q}_I(t) = 2\boldsymbol{c}_m - \boldsymbol{q}(t)$ of the first quarter of the oscillation. The potential field acting outside the switching circle is displayed in Figure 5.3 as its torque vectors pointing towards $\boldsymbol{q}_{\text{entry}} = (q_{c_1,1}, q_{c_2,2})^\top$. In this case, $r_{c_1} = r_{c_2}$ holds and thus $\boldsymbol{q}_{\text{entry}} = \boldsymbol{q}_{c_1} = \boldsymbol{q}_{c_2}$. Let clutch i be engaged with the stored equilibrium of the

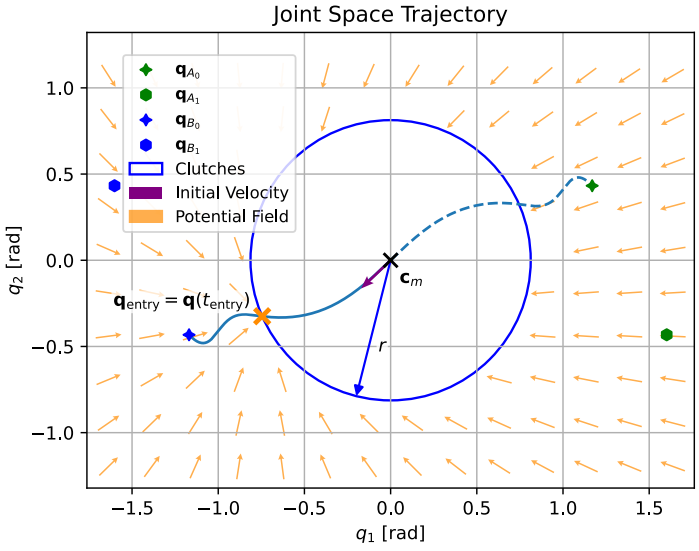
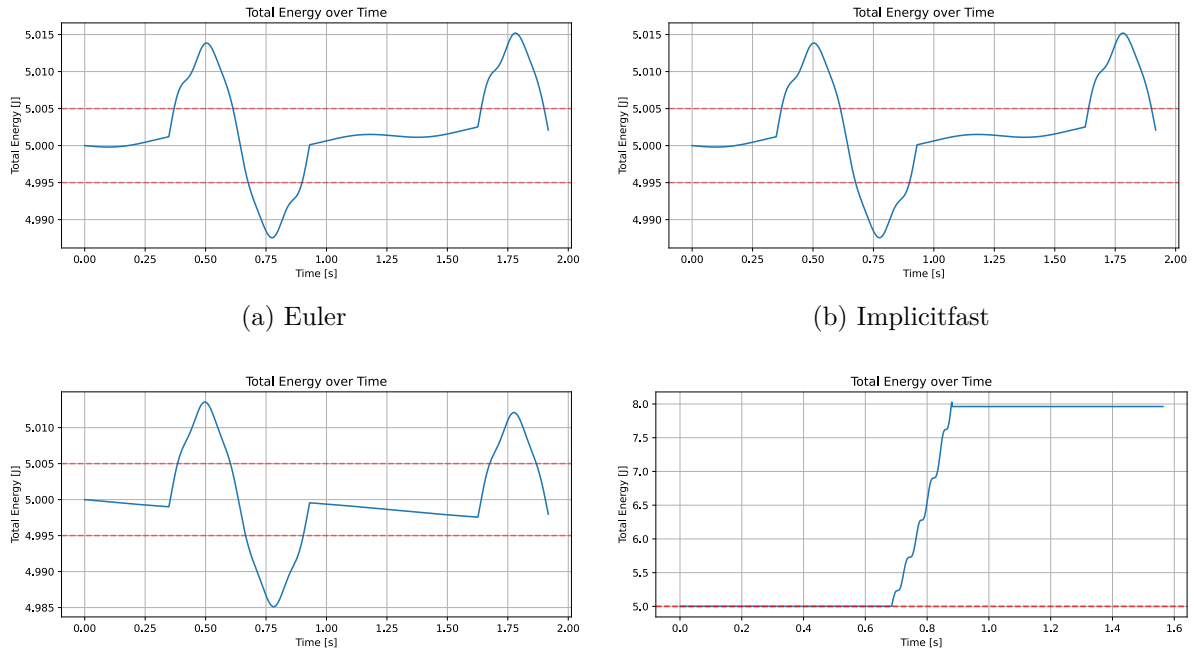


Figure 5.3.: Potential field of the clutches in joint space for a trajectory from $\boldsymbol{c}_m = (0, 0)$ to $\boldsymbol{q}_{B_0}$.

spring $q_{\text{entry},i} = q_i(t_{\text{entry},i})$ from the crossing point into the potential field. When the robot re-enters the circle at a different crossing point than the internally stored $q_{\text{entry},i}$, the spring is still deflected. The clutches disengage and the remaining spring energy is lost. Due to this, E_{total} decreases and the system is not conservative anymore. This behaviour is intentional since it reproduces the behaviour of a real clutch releasing a deflected spring. In the case of our desired point-symmetric oscillation defined in Section 4.3, the robot comes to rest inside the potential field with $E_{kin} = 0$, where time reversibility forces the trajectory to retrace its incoming trajectory and exit the potential field at the same crossing point $q_{\text{entry},i}$ where it entered it. The spring at this instant is therefore fully relaxed at disengagement and all stored energy is returned to the robot, keeping the system conservative.

5.1.3. Numerical Energy Consistency

If a trajectory enters and exits the potential field at the same point, we expect that the total energy of the internally modelled robot and the externally modelled clutch mechanism is numerically conservative with $E_{total} = E_{pot} + E_{kin}$. This means MuJoCo's integrator needs to be energy-consistent within our tolerance of $0.1\% \cdot E_{total}$ when external torques are applied. Figures 5.4c, 5.4a and 5.4b show the total energy over time during a period where we engage



(a) Euler

(b) Implicitfast

(c) Implicit

(d) RK4

Figure 5.4.: Total energy over time for different integrators of MuJoCo.

the clutch after roughly 0.3 s, disengage it and then engage it again. The oscillations around E_{total} for Euler and Implicitfast have a smaller amplitude when no external spring force acts on the system, though one can observe a slight drift increasing E_{total} over time. Even though Implicit also oscillates when the clutches are engaged, its curve looks more like a line indicating that it loses energy over time. The three figures thus show that none of these integrators is energy-consistent, since their total energy does not stay within our tolerance.

Considering Figure 5.4d, one can see that RK4 drifts a lot during clutch engagement, but stays at the same level when no external force acts on the system. According to MuJoCo [69], RK4 is qualitatively better in stability and accuracy for energy-conserving systems when there are no large velocity-dependent forces. Since the applied spring torque is position dependent, we chose RK4 as an integrator and prevented the drift during engaged clutches by calculating the spring torques in a user-defined control function that MuJoCo calls during the simulation (`mjcb_control`). The control function adds the spring torques as external joint torques (`qfrc_applied`) to MuJoCo. Since RK4 regularly calls this control function, the external torque is updated whenever RK4 changes the joint position (Section 3.3, [70]). This way, we are within our tolerances around the total energy, as can be seen in Figure 5.5.

Though the energy is within tolerances, there are still noticeable jumps when we engage and disengage the clutch. Since engaging should not cost energy in our simulation, those jumps come from numerical inaccuracies near the switching events. Since MuJoCo samples the trajectory at discrete time steps, we can only estimate the exact point where the trajectory crosses the

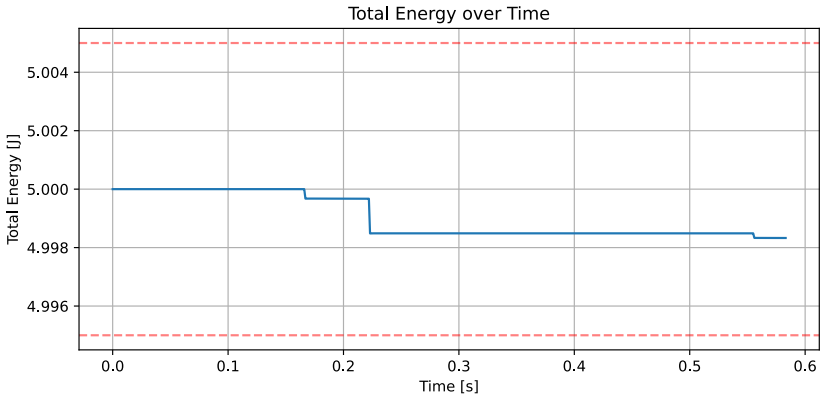


Figure 5.5.: Total energy over time with RK4 and the spring torque applied in the control function.

circle in our external model. As we already discussed in Section 3.3, RK4 works best with differential equations that are continuous in their fourth derivative. Our modelled switching system is a hybrid system that piecewise fulfils the condition for good RK4 integration and is position dependent except at the times when we switch the system. Globally, the spring torque depends on the history of $\boldsymbol{q}(t)$. This is usually solved by integrating such that RK4 stops exactly at the time when we would switch the clutches' state and within one RK4 integration we use either $q_{\text{entry},i} = q_i(t_{\text{entry},i})$ or $q_{\text{entry},i} = q_i$ depending on the clutch state [39, Chapter II.6 Discontinuous Equation]. This way, we can use $q_{\text{entry},i} = q_i(t_{\text{entry},i})$ when we are within the potential field and use $q_{\text{entry},i} = q_i$ when we are outside the potential field. To implement this in our simulation, we use the so-called Event Location method to determine when one of the clutches would switch [39, Chapter II.6 Event Location]. After every RK4 integration,

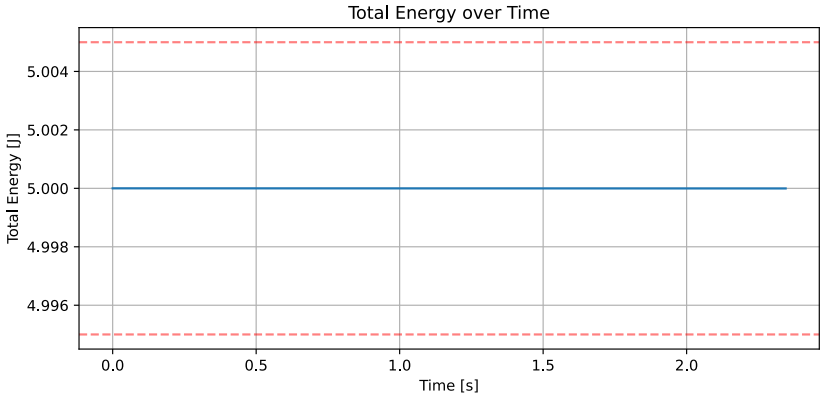


Figure 5.6.: Total energy over time with RK4 and event location of the switching events.

we determine if we crossed one of the circles by using cubic Hermite interpolation with the given position and velocity at the current and previous sampled positions from MuJoCo and estimate the time when the first event occurred between those samples. Afterwards, we reset our position to the previous one and reduce our MuJoCo step size such that RK4 integrates to the time that we estimated. At this point, we again check when the first event occurred and repeat the process until we either have tried more than 10 times to find the exact crossing point or we are within the tolerance of 10^{-12} rad away from the circle with radius r_{c_i}. For the event that engages the clutch, we aim to be between the two circles r_{c_i} and $r_{c_i} + 10^{-12}$, while

we aim for the disengaging event to be between the circles r_{c_i} and $r_{c_i} - 10^{-12}$. This way, there are no visible jumps and drifts in RK4 anymore when engaging or disengaging the clutch, as Figure 5.6 illustrates. We can thus assume in the following optimization problems that MuJoCo models the system as conservative for point-symmetric oscillations in joint space.

5.2. No-Spring Optimization

The no-spring optimization implements the first stage from Section 4.3.1. It considers arbitrary points A_i and B_j with $i, j \in \{0, 1\}$ and optimizes for a trajectory from $\boldsymbol{q}_{A_i}$ to $\boldsymbol{q}_{B_j}$ in the conservative system. Since all clutches stay disengaged, no potential field acts. The optimization result is successful if the trajectory reaches the Cartesian point B in the configuration $\boldsymbol{q}_{B_j}$ within 1 mm.

5.2.1. Decision Variables

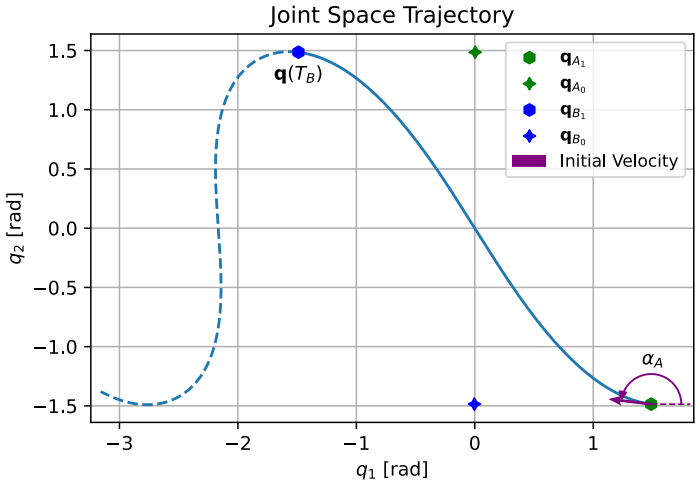


Figure 5.7.: Joint-space trajectory of the no-spring optimization from $\boldsymbol{q}_{A_1}$ to $\boldsymbol{q}_{B_1}$.

Figure 5.7 shows a successful result of the no-spring optimization where the joint-space trajectory goes from $\boldsymbol{q}_{A_1}$ to $\boldsymbol{q}_{B_1}$. The dashed line is the continuation of this trajectory for $t > T_B$. The figure also displays the decision variables, namely the time $T_B = T_{B_j}$ needed to reach $B = B_j$ and the angle of the initial velocity (3.27), whose magnitude follows from $E_{total} = 5\,\mathrm{J}$ with (3.28). Since the no-spring optimization starts its trajectories at $\boldsymbol{q}_{A_i} = \boldsymbol{q}_A = \boldsymbol{q}(0)$, we denote this angle as α_A. We only need T_B as a decision variable to select the joint state for which we calculate the cost. This way, the cost function needs no minimum in its calculation and is differentiable, as can be seen in Section 5.2.4. Furthermore, T_B needs no bound against negative times, since the time reversibility in (4.4) allows us to replace a negative T_B by the positive time $|T_B|$ with the angle α_A rotated by π, effectively reversing the initial velocity.

5.2.2. Initial Guess

To avoid trajectories that turn the joints several times as in Figure 5.8, we aim for single-turn trajectories. We therefore set $\boldsymbol{q}_B$ to the representation of $\boldsymbol{q}_{B_j}$ relative to $\boldsymbol{q}_{\mathrm{ref}} = \boldsymbol{q}_{A_i}$ from Definition 10, which has the shortest joint-space distance (Definition 11) to $\boldsymbol{q}_A = \boldsymbol{q}_{A_i}$. On the direct path to this representation, every joint turns by at most π, so that single-turn trajectories

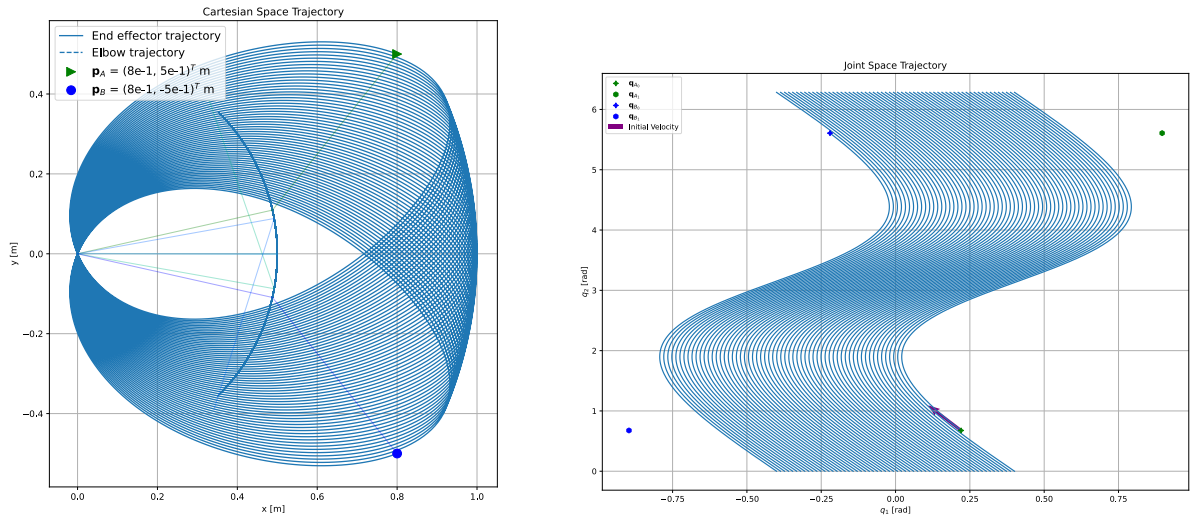
can reach it. A representation that lies more than 2π away from $\boldsymbol{q}_A$ in one joint can instead only be reached by multi-turn trajectories. We initialize T_B with 0 s, since we want to find the fastest way towards $\boldsymbol{q}_B$ and therefore start from the shortest possible time. The initial angle α_A is the angle of the direction vector $\boldsymbol{q}_B - \boldsymbol{q}_A$, so that the first free-response shot points towards $\boldsymbol{q}_B$.

5.2.3. Optimization Algorithm

To solve this optimization problem, we use the Trust Region Reflective algorithm [12] implemented in SciPy's `least_squares` [88]. The solver rescales the decision variables in every iteration with the inverse column norms of the Jacobian matrix [68]. All other settings remain at their default values. Starting from the initial guess of Section 5.2.2, this local gradient-based optimizer proved sufficient to find a suitable result.

5.2.4. Cost Function

Our cost function $\text{cost}_{rigid} = 0.5 \left\| \boldsymbol{s}_{rigid} \right\|^2$ is differentiable and consists of the residual vector $\boldsymbol{s}_{rigid} = \boldsymbol{q}_B(T_B, \alpha_A) - \boldsymbol{q}_B$ in rad. Since $\boldsymbol{s}_{rigid}$ compares the joint angles directly, the residual only vanishes when the trajectory reaches exactly the representation $\boldsymbol{q}_B$ from Section 5.2.2. A cost function that treats joint angles differing by multiples of 2π (Section 3.1) as equal would instead vanish for every representation of $\boldsymbol{q}_{B_j}$. It could then lead the optimization to a multi-turn trajectory as in Figure 5.8, which reaches B only after about 54.58 s. Figure 5.8a shows the oscillations along the trajectory in Cartesian space, while Figure 5.8b represents them in joint space. The implemented cost function controls which representation of $\boldsymbol{q}_B$ and thus how many turns per joint between $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$ the optimization aims at.



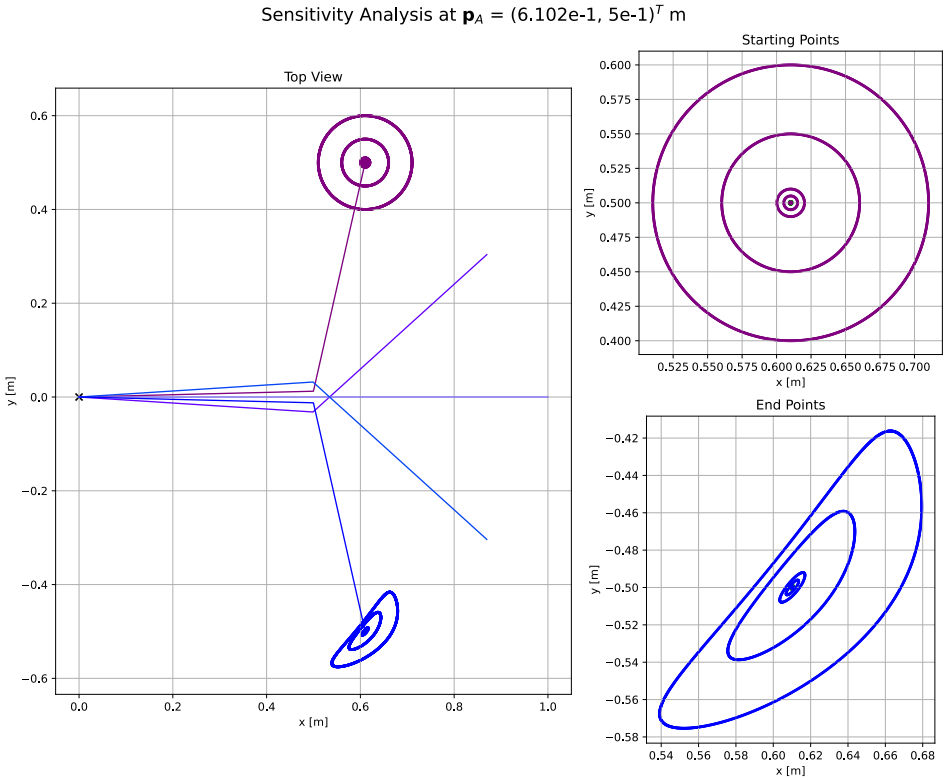
(a) Cartesian Space

(b) Joint Space

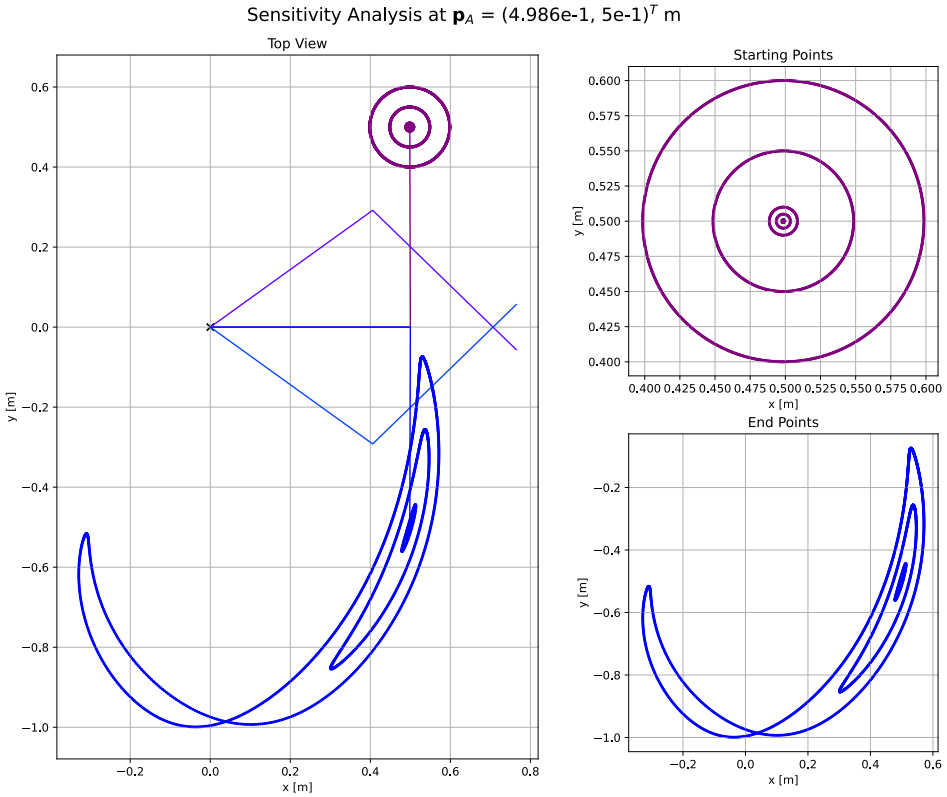
Figure 5.8.: Multi-turn trajectory from $\boldsymbol{q}_{A_0}$ to $\boldsymbol{q}_{B_0}$ for $\boldsymbol{p}_A = (0.8, 0.5)^\top$ m and $\boldsymbol{p}_B = (0.8, -0.5)^\top$ m.

5.2.5. Robustness Analysis

To analyse how robust the system is at $E_{total} = 5\,\text{J}$ against deviations of A, we optimize the trajectory from $\boldsymbol{q}_{A_0}$ to $\boldsymbol{q}_{B_0}$ with $i = j = 0$. Around the starting position A, we place five circles

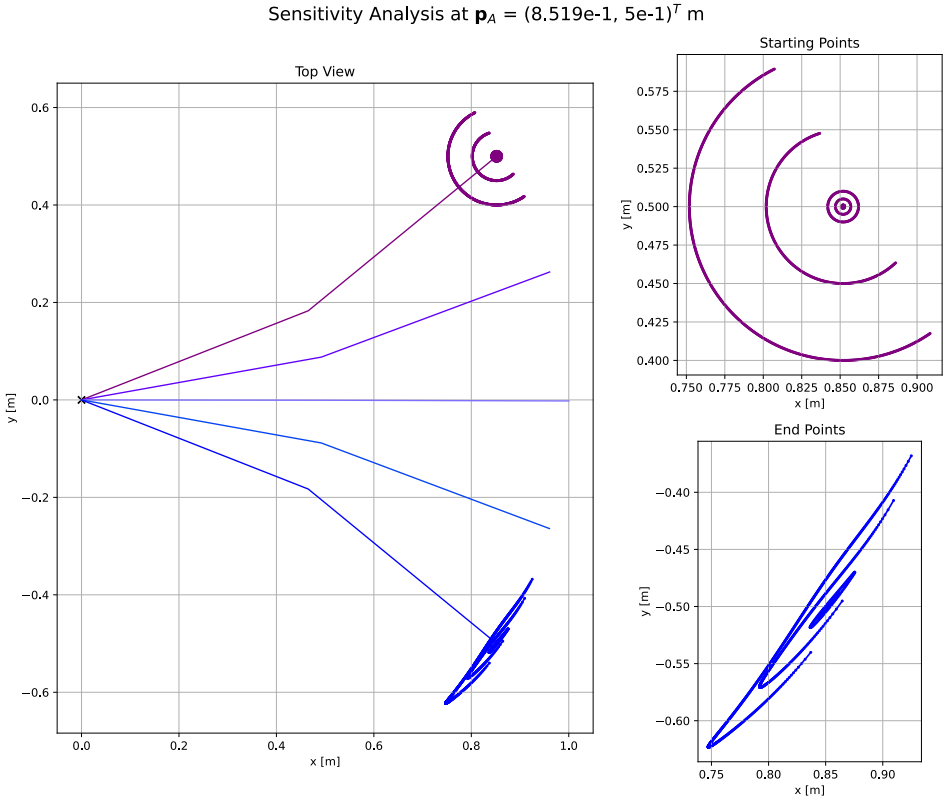


(a) Stretched Arm, $\boldsymbol{p}_A = (0.6102, 0.5)^\top$ m



(b) Folded Arm, $\boldsymbol{p}_A = (0.4986, 0.5)^\top$ m

Figure 5.9.: Start points sampled around $\boldsymbol{p}_A$ and the corresponding end points at T_B^*.



(c) Boundary Singularity, $\boldsymbol{p}_A = (0.8519, 0.5)^\top$ m

Figure 5.9.: Start points sampled around $\boldsymbol{p}_A$ and the corresponding end points at T_B^*.

centred at $\boldsymbol{p}_A$ with the radii $r_{robust} \in \{10^{-3}, 5 \cdot 10^{-3}, 10^{-2}, 5 \cdot 10^{-2}, 10^{-1}\}$ m. On each circle, we sample Cartesian points $\hat{\boldsymbol{p}_A}$ with a distance of $5 \cdot 10^{-4}$ m between neighbouring samples. Each sample starts a simulation in its configuration $\hat{\boldsymbol{q}_{A_0}}$ with the decision variables α_A^* and T_B^* of the successful optimization result. Figure 5.9 shows the end positions reached at T_B^* for the various deviations from A. In each subfigure, the upper right plot shows the sampled points $\hat{\boldsymbol{p}_A}$ on the circles and the lower right plot the corresponding Cartesian points at which the simulations end at T_B^*. In addition to both point sets, the left plot displays five joint configurations of the optimized trajectory by the positions of its links in Cartesian space. Besides the start and end configuration, these include three intermediate configurations chosen such that the same number of simulation steps lies between each two consecutive configurations.

Figure 5.9a shows that the circles are only slightly distorted when the trajectory swings over the stretched arm lying on the x-axis. If the trajectory swings over the arm folded back on itself along the x-axis as in Figure 5.9b, the circles with larger radii are much more distorted than for the stretched arm. Figure 5.9c further shows that the circles are also distorted when A and B lie close to a singularity. Overall, these figures still show that the trajectory reaches the target position B for small deviations of A.

5.3. Stiffness Optimization

The stiffness optimization implements the second stage from Section 4.3.3 and searches for point-symmetric oscillations in joint space for each configuration pair A_i/B_i and each centre configuration $\boldsymbol{c}_m$. Following Theorem 1, the least squares optimizations only aim at stopping in $\boldsymbol{q}_{B_i}$, while the success criteria in Section 5.3.2 select the point-symmetric oscillations. Each simulation starts in $\boldsymbol{c}_m$, which is the shared equilibrium of both clutches from Section 4.3.2.

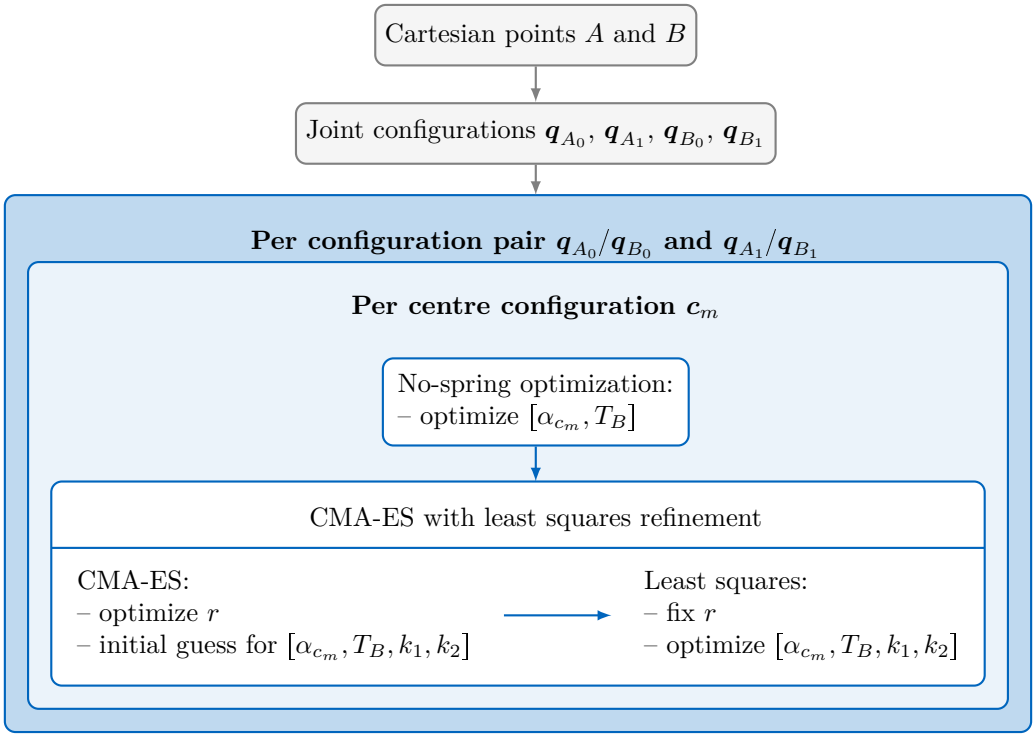


Figure 5.10.: Stiffness optimization per configuration pair and centre configuration $\boldsymbol{c}_m$ (AI-generated illustration)

As the abstract overview in Figure 5.10 displays, for each configuration pair, the optimizations for the four centre configurations $\boldsymbol{c}_m$ run in parallel. The no-spring optimization from Section 5.2 provides the initial guess for the angle α_{c_m} of the initial velocity and T_B, the time at which the trajectory stops in B. CMA-ES from `pycma` [40, 30] then optimizes the radius r of the switching circles from Section 5.1.2 and provides the least squares optimizations with initial guesses for α_{c_m}, T_B and the stiffness values k_1 and k_2. Section 5.3.1 defines these decision variables with their bounds.

5.3.1. Decision Variables

The decision variables of the stiffness optimization form the vector $\boldsymbol{z}_{stiff}$ in (5.1) and are displayed in Figure 5.11, which shows the first quarter of an oscillation and its continuation as a dashed line. Similar to Section 5.2.1, T_B is only there to select the joint state for the cost function to evaluate if we stopped in B and it has no bounds. In the shared equilibrium $\boldsymbol{c}_m$, the kinetic energy equals E_{total} and gives the magnitude of the initial velocity (3.28), so that only its angle α_{c_m} remains to be optimized. The stiffness values k_1 and k_2 of the torsional springs (4.1) produce the potential field shown in Figure 5.11 when the clutches are engaged. The switching circles of the clutches coincide as $r_{c_1} = r_{c_2} = r$ holds for the stiffness optimization.

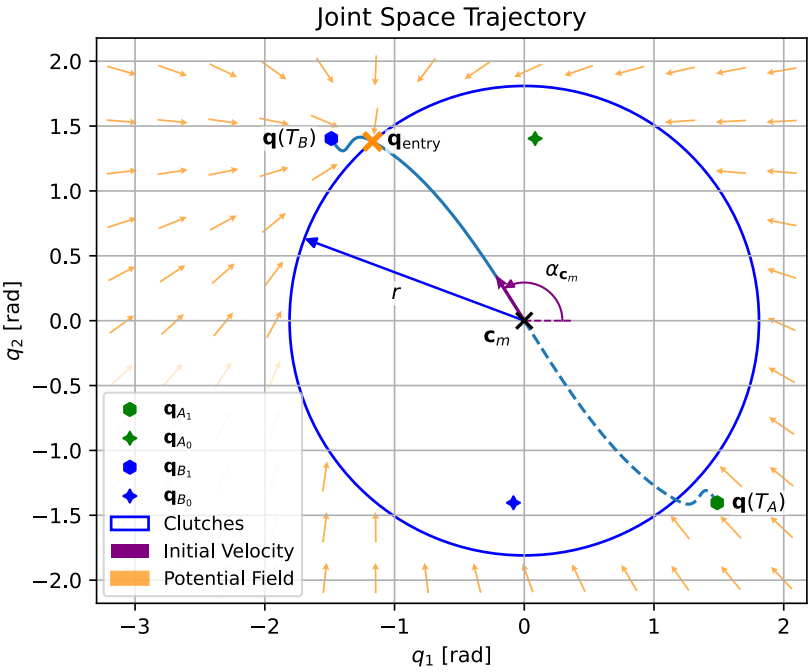


Figure 5.11.: Joint-space trajectory of the stiffness optimization for $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ with the common switching circle and the potential field

The times of the switching events do not have to be known in advance, since the clutch model from Section 5.1.2 switches the clutches and updates $\boldsymbol{q}_{\text{entry}}$ during the simulation.

$$\begin{aligned} \boldsymbol{z}_{stiff} &= (\alpha_{c_m}, T_B, r, k_1, k_2)^\top , \\ \alpha_{c_m} &\in [\alpha_{c_m,init} - \pi, \alpha_{c_m,init} + \pi]\,\text{rad}\,, \qquad T_B \in \mathbb{R}\,, \\ r &\in [10^{-3}, \|\boldsymbol{q}_B - \boldsymbol{c}_m\| - 10^{-5}]\,\text{rad}\,, \qquad k_1, k_2 \in [50, 1000]\,\text{N m/rad} \end{aligned} \tag{5.1}$$

The angle α_{c_m} is bounded to one turn around its initial guess from Section 5.3.3, so that every initial direction is represented uniquely. In contrast, the bounds on the stiffness values in (5.1) are only there to keep them realistic, since an infinite stiffness would be a trivial result of the optimization, for which the trajectory only has to reach B to stop there. With $\boldsymbol{q}_B$ represented relative to $\boldsymbol{c}_m$, the upper bound of r guarantees that $\boldsymbol{q}_B$ is always outside the switching circle, so that both clutches are engaged when the trajectory stops there. The optimization algorithm

Optimizer	$\boldsymbol{\alpha_{c_m}}$	$\boldsymbol{T_B}$	$\boldsymbol{r}$	**Stiffness Values**
CMA-ES	optimized	optimized	optimized	$\phi \in [0, 1]$
LS1_{stiff}	fixed	$T_B \geqslant T_{B,min}$	fixed	fixed
LS2_{stiff}	fixed	$T_B \geqslant T_{B,min}$	fixed	$\kappa \in [\kappa_{min}, \kappa_{max}]$
LS4_{stiff}	optimized	optimized	fixed	k_1, k_2

Table 5.4.: Decision variables of the optimizers in the stiffness optimization

in Section 5.3.4 combines CMA-ES from Section 3.4 with the least squares optimizations LS1_{stiff}, LS2_{stiff} and LS4_{stiff}, whose index gives the number of decision variables and whose subscript distinguishes them from the least squares optimizations of the clutch switching times optimization in Section 5.4. CMA-ES optimizes all entries of $\boldsymbol{z}_{stiff}$, while each least squares

optimizes only the subset listed in Table 5.4 and keeps the remaining entries fixed. For all optimizers, the cost function in Section 5.3.5 evaluates the simulation with the complete vector $\boldsymbol{z}_{stiff}$. LS2_{stiff} and CMA-ES represent the stiffness values in $\boldsymbol{z}_{stiff}$ by a single variable, the stiffness ratio κ and its logarithmic position $\phi \in [0,1]$ in the interval $[\kappa_{min}, \kappa_{max}]$, respectively. These single variables, defined in (5.2), use that $E_{kin} = 0$ and $E_{total} = E_{pot}$ hold when the trajectory stops in $\boldsymbol{q}_B$, effectively reducing the dimension of the decision variables.

$$\begin{aligned} \kappa &= \frac{k_1}{k_2} = \kappa_{min} \left(\frac{\kappa_{max}}{\kappa_{min}} \right)^{\phi} , \\ E_{total} = E_{pot} &= k_1 V_1 + k_2 V_2 \quad \text{with} \quad V_j = \frac{1}{2} \left(q_{B,j} - q_{\text{entry},j} \right)^2 , \quad j \in \{1,2\} , \\ \Rightarrow \quad k_2 &= \frac{E_{total}}{\kappa V_1 + V_2} , \quad k_1 = \kappa k_2 \end{aligned} \tag{5.2}$$

The interval $[\kappa_{min}, \kappa_{max}]$ contains all ratios for which k_1 and k_2 from (5.2) lie within their bounds in (5.1). Since V_j depends on $\boldsymbol{q}_{\text{entry}}$ and thus on α_{c_m} and r, this interval does not exist for some values of α_{c_m} and r. Since LS4_{stiff} changes α_{c_m} in its optimization steps, it optimizes for k_1 and k_2 directly. LS4_{stiff} keeps the radius r fixed, since the stiffness values and r correlate. Due to the shared radius r, both clutches engage simultaneously for the first time since $t_0 = 0\,\text{s}$ at t_{both}. t_{both} only depends on α_{c_m} and r, since there is no potential field for $t_0 < t < t_{both}$. As every successful result stops in $\boldsymbol{q}_B$ with both clutches engaged, LS1_{stiff} and LS2_{stiff} bound T_B from below by $T_{B,min} = t_{both} + dt$ with the time step dt from Table 5.3.

5.3.2. Success Criteria

Following Section 3.2.3, a point-symmetric oscillation stops in B at $T_B = \frac{T}{4}$ and in A at $3T_B = \frac{3T}{4}$. As explained in Section 5.1.2, a trajectory can stop in B but lose energy on its way when a clutch releases a deflected spring and then no longer stop in A. An optimization result $\boldsymbol{z}_{stiff}$ is therefore only successful if it stops in both B and A within the tolerances in (5.3), where f_{kin} is the forward kinematics from (3.1).

$$\|\boldsymbol{p}_B - f_{kin}(\boldsymbol{q}_B(\boldsymbol{z}_{stiff}))\| < 10^{-3}\,\text{m} , \qquad E_{kin}(\boldsymbol{z}_{stiff}) < 5 \cdot 10^{-5}\,\text{J} , \tag{5.3a}$$

$$\|\boldsymbol{p}_A - f_{kin}(\boldsymbol{q}_A(3T_B, \boldsymbol{z}_{stiff} \backslash \{T_B\}))\| < 10^{-3}\,\text{m} , \quad E_{kin}(3T_B, \boldsymbol{z}_{stiff} \backslash \{T_B\}) < 5 \cdot 10^{-5}\,\text{J} \tag{5.3b}$$

In this stage, we do not check whether the oscillation reaches $\boldsymbol{c}_m$ after $t = T$, since the results of this stage only serve as initial guesses for the clutch switching times optimization.

5.3.3. Initial Guess

Each of the four parallel optimizations per $\boldsymbol{c}_m$ from Section 5.3 needs an initial guess of $\boldsymbol{z}_{stiff}$ and ϕ. The no-spring optimization from Section 5.2 with the start configuration $\boldsymbol{c}_m$ and the target configuration $\boldsymbol{q}_B$, represented relative to $\boldsymbol{q}_{\text{ref}} = \boldsymbol{c}_m$, yields the angle α_{c_m} and the time T_B as shown in Figure 5.12.

Since the optimized trajectory still reaches B for small deviations of its start position (Section 5.2.5), we consider the potential field of the clutches as a small deviation from the trajectory and use α_{c_m} and T_B from the successful result as suitable initial guesses. We choose the initial values for r and ϕ to be the middle of their bounds in (5.1) and Table 5.4 so that CMA-ES centres its sample distribution in those values.

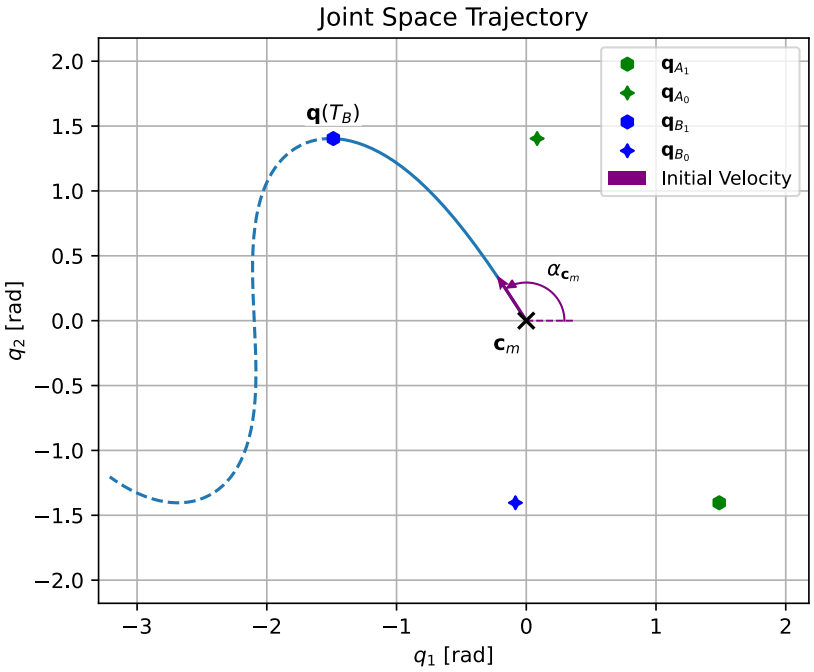


Figure 5.12.: Joint-space trajectory of the no-spring optimization from $\boldsymbol{c}_m$ to $\boldsymbol{q}_{B_1}$.

5.3.4. Optimization Algorithm

Since a single successful result from Section 5.3.2 for one $\boldsymbol{c}_m$ is sufficient, the first of the four parallel optimizations that finds one ends all of the ongoing optimizations. Each optimization then returns the results it has found until then. To prefer oscillations with a shorter distance between A and B, only the centre configurations $\boldsymbol{c}_m$ with the two shortest joint-space distances between $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$ can end the search during the first hour. Each optimization runs CMA-ES

Option	Value
`sigma0`	$1/4$
`CMA_stds`	$\left(2\pi,\ 1.5\max(\lvert T_{B,init}\rvert, 10\,dt),\ (\lVert\boldsymbol{q}_B - \boldsymbol{c}_m\rVert - 10^{-5}) - 10^{-3},\ 1\right)$
`popsize`	$3\lambda + 2 = 26$
`tolfun`	10^{-5}
`tolx`	10^{-4}
`tolfunhist`	10^{-5}
`tolstagnation`	2
`maxfevals`	5000
`restarts`	9
`incpopsize`	4
`restart_from_best`	False
`bipop`	True

Table 5.5.: Options of the algorithm `fmin2` from `pycma` [74] in the stiffness optimization

with restarts with the algorithm `fmin2` from `pycma` [40, 30] with the options in Table 5.5 and samples $\boldsymbol{x} = (\alpha_{c_m}, T_B, r, \phi)$ within the bounds from Section 5.3.1. Following the rule of thumb (Section 3.4), $\sigma^{(0)} = 1/4$ together with `CMA_stds` gives an initial step size of a quarter of the bound width of each parameter. Table 5.5 displays the bound widths from `CMA_stds` in the

same order as the decision variables are stored in $\boldsymbol{x} = (\alpha_{c_m}, T_B, r, \phi)$. Only T_B has no bounds and is therefore scaled with its initial guess instead. We use three times the default population size λ from Section 3.4 for the four entries of $\boldsymbol{x}$ and add the two samples injected after each generation to the population size. With the restart options, CMA-ES serves as a global sampler that restarts with a new population instead of restarting from its best sample. Before the first generation, the initial mean is evaluated and after each generation, the steps in Figure 5.14 run. In the evaluation of each sample and in the steps in Figure 5.14, LS1_{stiff}, LS2_{stiff} and LS4_{stiff} use the same least squares algorithm implemented in SciPy as the no-spring optimization in Section 5.2.3, but with different decision variables (Table 5.4) and a different cost function (Section 5.3.5).

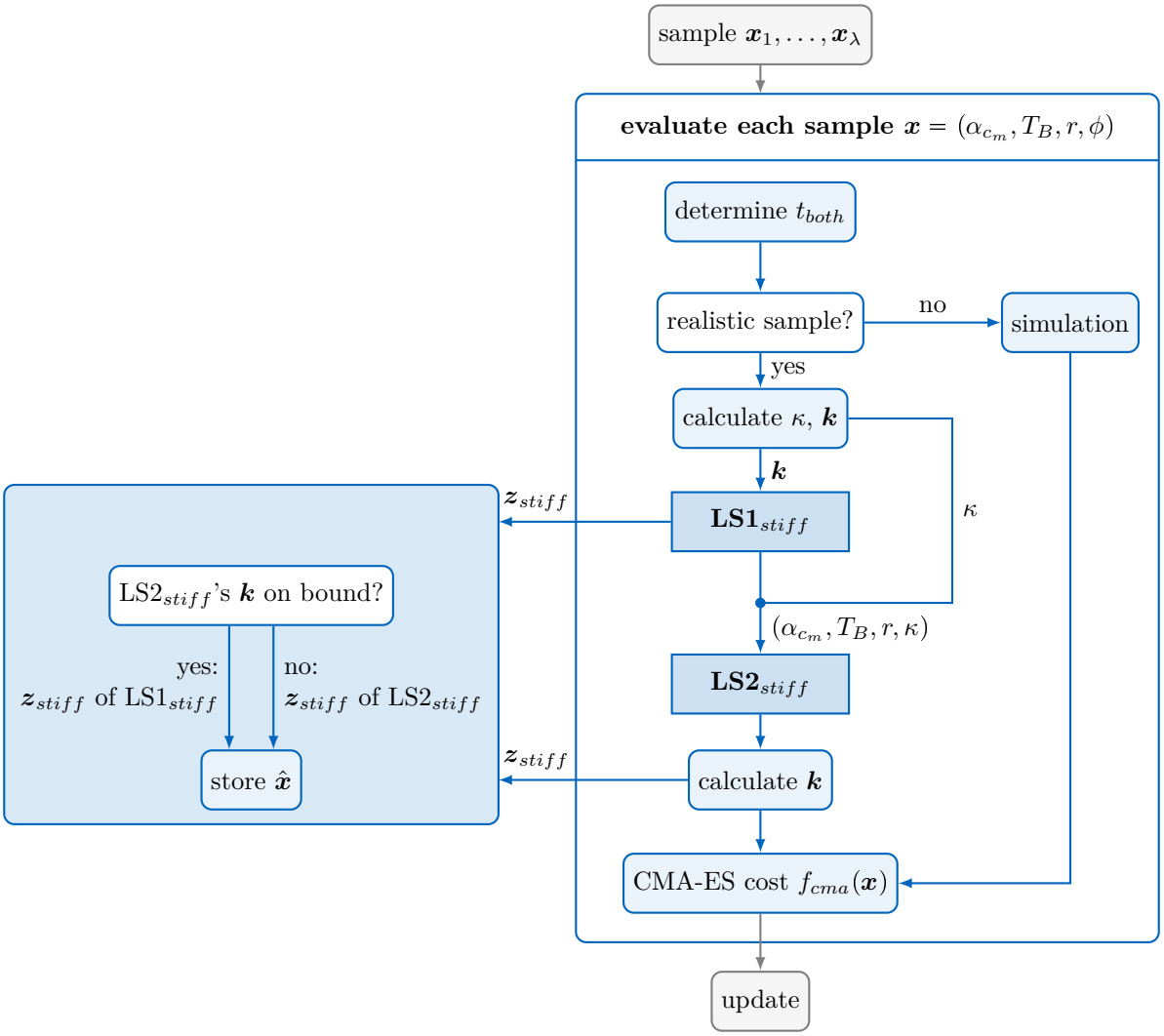


Figure 5.13.: Evaluation of a CMA-ES sample in the stiffness optimization (AI-generated illustration)

Figure 5.13 shows how the objective function evaluates each sample $\boldsymbol{x}$. A simulation determines t_{both} and $\boldsymbol{q}_{\text{entry}}$ and thus the interval $[\kappa_{min}, \kappa_{max}]$ from Section 5.3.1. If the trajectory enters the switching circle within $10\,T_B$ and the interval exists, the sample is realistic and ϕ gives κ and the stiffness values $\boldsymbol{k} = (k_1, k_2)$ with (5.2). LS1_{stiff} optimizes T_B with these stiffness values and passes it together with κ to LS2_{stiff}, whose result gives the cost $f_{cma}(\boldsymbol{x})$ from Section 5.3.5 for CMA-ES. LS2_{stiff} is limited to 100 function evaluations with termination tolerances of

10^{-5} for the change of the cost function and of its decision variables T_B and κ, since it only checks whether the sample is worth refining with LS4_{stiff} later in the steps of Figure 5.14. The block on the left in Figure 5.13 stores the sample as $\hat{\boldsymbol{x}}$ together with $\boldsymbol{z}_{stiff}$, the cost of LS2_{stiff} and whether the optimized trajectory stops in B within (5.3a). If a stiffness of LS2_{stiff} lies within 0.4 N m/rad of its bound values, $\hat{\boldsymbol{x}}$ keeps $\boldsymbol{z}_{stiff}$ of LS1_{stiff} instead, since LS2_{stiff} then got stuck at the stiffness bounds and would give LS4_{stiff} a poor initial guess.

A sample that is not realistic skips LS1_{stiff} and LS2_{stiff} and is not stored as displayed in Figure 5.13. If its trajectory does not enter the switching circle, ϕ maps linearly onto the stiffness bounds with $k_1 = k_2$. Otherwise, ϕ gives κ with (5.2) in the widest interval $[\frac{k_{min}}{k_{max}}, \frac{k_{max}}{k_{min}}] = [0.05, 20]$ that the stiffness bounds k_{min} and k_{max} from (5.1) allow. Afterwards, the stiffness values resulting from applying ϕ to the bounds are smoothly clamped into the bounds of k_1 and k_2 with the shifted log-sum-exp smooth minimum and maximum [35, 10]. A single simulation with these stiffness values then gives $f_{cma}(\boldsymbol{x})$ with the cost function from Section 5.3.5.

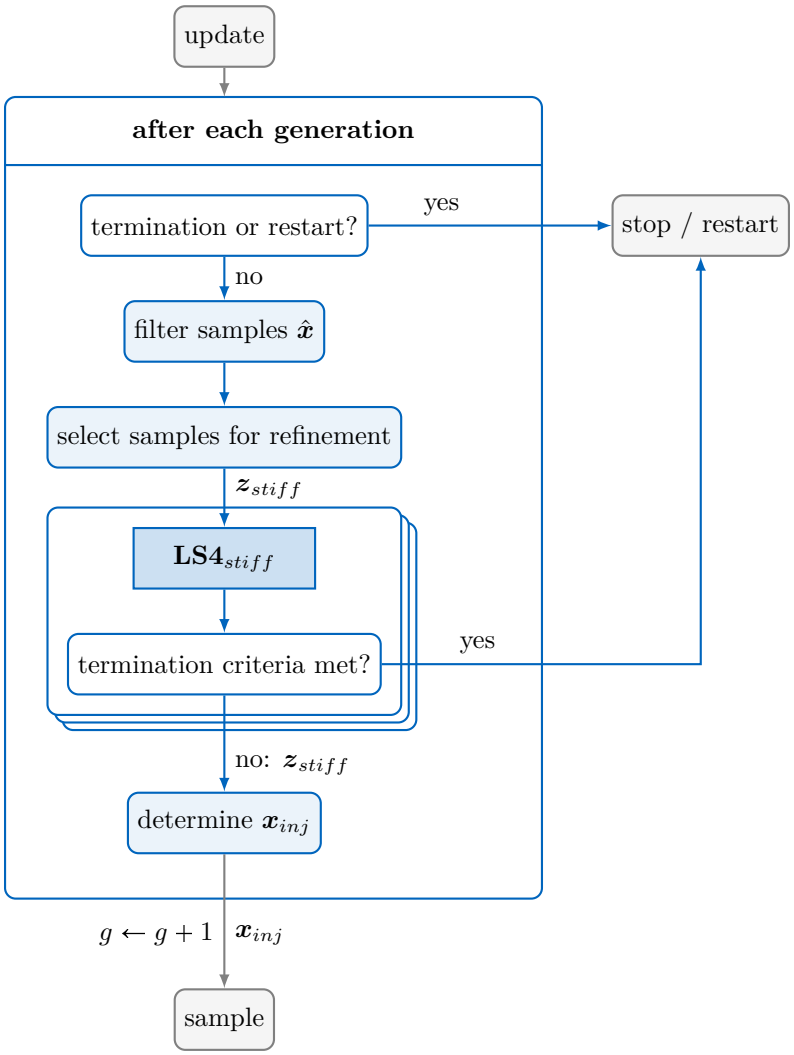


Figure 5.14.: Steps of the stiffness optimization after each CMA-ES generation (AI-generated illustration)

Figure 5.14 shows the steps after each generation. CMA-ES stops if another of the parallel

optimizations has ended the search or if a result meets one of two termination criteria. The termination criteria are met when the result either stops in B with numerical accuracy, with a distance below 10^{-8} m and $E_{kin} < 10^{-15}$ J, or it fulfils (5.3) after more than 50 least squares calls. In addition, CMA-ES stops after two hours on the computer from Table 5.1 and leaves its centre configuration and configuration pair unsolved, since a better initial guess from neighbouring Cartesian and joint-space points could still reach a successful result if desired. In general, CMA-ES restarts if the best result has stayed at the stiffness bounds within 0.4 N m/rad for 60 minutes or has improved neither its distance to B nor its kinetic energy by more than 1 % within 70 minutes.

If neither a restart nor the termination is triggered, the filter drops each sample $\hat{\boldsymbol{x}}$ whose distance to one of the previously injected points $\boldsymbol{x}_{inj}$, scaled with `CMA_stds`, lies below 10^{-6}. This is done, since such a sample is close to an injected one and has already been refined by LS4$_{stiff}$. From the remaining samples, all that stop in B within (5.3a) are selected for the LS4$_{stiff}$ refinement. If there are fewer than three stopping in B, the samples with the lowest cost of LS2$_{stiff}$ are used until we have three selected for LS4$_{stiff}$ refinement. In addition, one randomly chosen sample from the remaining samples is refined. The selected samples for refinement give their stored $\boldsymbol{z}_{stiff}$ to LS4$_{stiff}$ as initial guesses.

To prevent a vanishing gradient with respect to the stiffness values, each simulation of LS4$_{stiff}$ replaces the fixed r by the shifted log-sum-exp smooth minimum [35, 10] of r and $r_{sim} = \|\boldsymbol{q}(T_B - 0.999\,dt) - \boldsymbol{c}_m\|$ with the sharpness $\beta = 30/r$, where r_{sim} is the joint-space distance to $\boldsymbol{c}_m$ that the trajectory without springs reaches shortly before T_B. This way, both clutches engage before T_B and the resulting minimum r_{eff} is a smooth function of α_{c_m} and T_B within the LS4$_{stiff}$ optimization steps, ensuring that LS4$_{stiff}$ has smooth gradients. After each LS4$_{stiff}$, the termination and restart criteria from above are checked again.

If none of the termination and restart criteria are met after the last LS4$_{stiff}$, the final step in Figure 5.14 determines the injected points $\boldsymbol{x}_{inj}$ for the next generation. Only the results of LS4$_{stiff}$ are injected, since CMA-ES does not yet know these samples and their cost internally. Results from LS4$_{stiff}$ whose stiffness values lie within 0.4 N m/rad of their bounds are excluded. For each remaining result, ϕ follows from $\kappa = \frac{k_1}{k_2}$ with (5.2). For this, the interval $[\kappa_{min}, \kappa_{max}]$ is recomputed with the entry configuration $\boldsymbol{q}_{\text{entry}}$ of the trajectory that the LS4$_{stiff}$ result $\boldsymbol{z}_{stiff}$ produces. Since $E_{total} = E_{pot}$ in (5.2) only holds exactly when this trajectory stops in $\boldsymbol{q}_B$, the interval may not exist for results that do not stop in B. Results without a valid interval are thus skipped and not chosen for $\boldsymbol{x}_{inj}$. The two results with the lowest cost become the points $\boldsymbol{x}_{inj}$, which CMA-ES receives as injected external samples, implemented with `inject` from `pycma`, after transforming them into its internal coordinates [22]. By default, CMA-ES only uses each injected sample as a direction from the mean of the distribution. If the best result of the whole optimization algorithm so far lies at the stiffness bounds, we inject the points unscaled [22] such that CMA-ES gets forced samples $\boldsymbol{x}_{inj}$ that may be outside its distribution but help move the mean in the direction of samples with lower cost.

5.3.5. Cost Functions

LS1$_{stiff}$, LS2$_{stiff}$ and LS4$_{stiff}$ minimize cost$_{stiff}$ in (5.4). The residual vector $\boldsymbol{s}_B$ takes over the position residual of the no-spring optimization from Section 5.2.4 with $\boldsymbol{q}_B$ represented relative to $\boldsymbol{c}_m$, so that the stiffness optimization aims at oscillations with the shortest joint-space distance between $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$ and thus at single-turn oscillations. In addition, $\boldsymbol{s}_B$ contains the mass-weighted velocity $\dot{\boldsymbol{q}}_{\text{mass}}$ from (3.26), such that the cost only vanishes when the trajectory stops in $\boldsymbol{q}_B$. CMA-ES receives $f_{cma}(\boldsymbol{x})$, which adds the residual $\boldsymbol{s}_A$ of the stop in A from (5.3b).

Its weight 10^{-6} keeps the stop in B dominant for the optimization.

$$\begin{aligned}
\boldsymbol{s}_B &= \left(\frac{\boldsymbol{q}_B(\boldsymbol{z}_{stiff}) - \boldsymbol{q}_B}{\epsilon_q}, \frac{\dot{\boldsymbol{q}}_{\text{mass}}(\boldsymbol{z}_{stiff})}{\sqrt{2\epsilon_E}} \right)^\top, \\
\boldsymbol{s}_A &= \left(\frac{\boldsymbol{q}_A(3T_B, \boldsymbol{z}_{stiff} \backslash \{T_B\}) - \boldsymbol{q}_A}{\epsilon_q}, \frac{\dot{\boldsymbol{q}}_{\text{mass}}(3T_B, \boldsymbol{z}_{stiff} \backslash \{T_B\})}{\sqrt{2\epsilon_E}} \right)^\top, \\
\text{cost}_{stiff} &= \frac{1}{2}\|\boldsymbol{s}_B\|^2 = \frac{1}{2}\left(\frac{\|\boldsymbol{q}_B(\boldsymbol{z}_{stiff}) - \boldsymbol{q}_B\|^2}{\epsilon_q^2} + \frac{2E_{kin}(\boldsymbol{z}_{stiff})}{2\epsilon_E} \right), \\
f_{cma}(\boldsymbol{x}) &= \text{cost}_{stiff} + 10^{-6}\,\frac{1}{2}\|\boldsymbol{s}_A\|^2
\end{aligned} \tag{5.4}$$

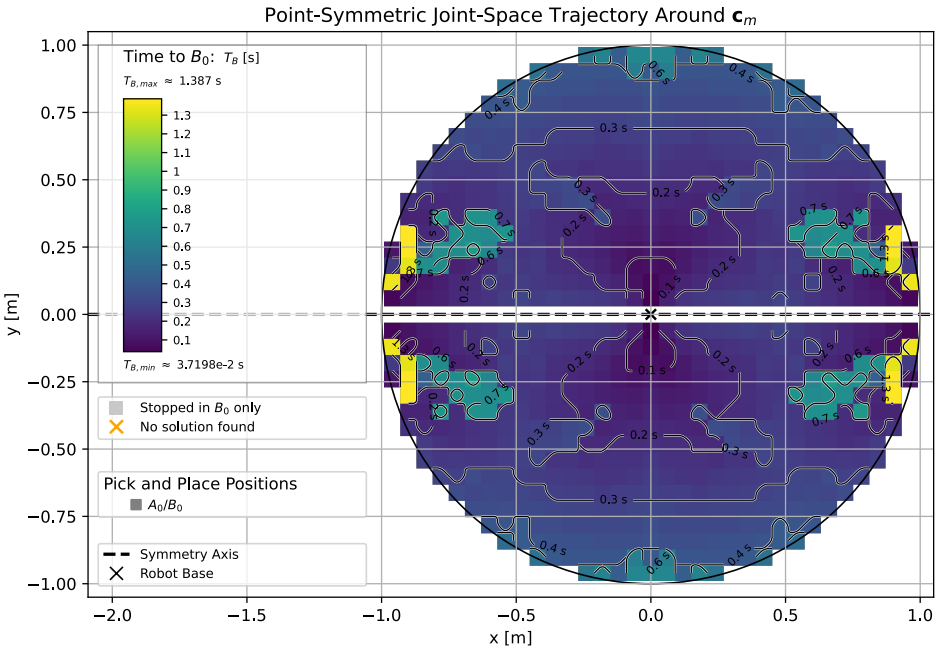
Both parts of the residuals are scaled with the tolerances of (5.3), namely the energy tolerance $\epsilon_E = 5 \cdot 10^{-5}\,\text{J}$ and the joint-space tolerance ϵ_q from (3.18). The joint-space tolerance thus guarantees the Cartesian tolerance $\epsilon_c = 10^{-3}\,\text{m}$ if the joint-space distance is within the tolerance. Each term of $\|\boldsymbol{s}_B\|^2$ and $\|\boldsymbol{s}_A\|^2$ stays below 1 as long as the joint deviation lies within ϵ_q and the kinetic energy below ϵ_E.

5.3.6. Workspace Coverage

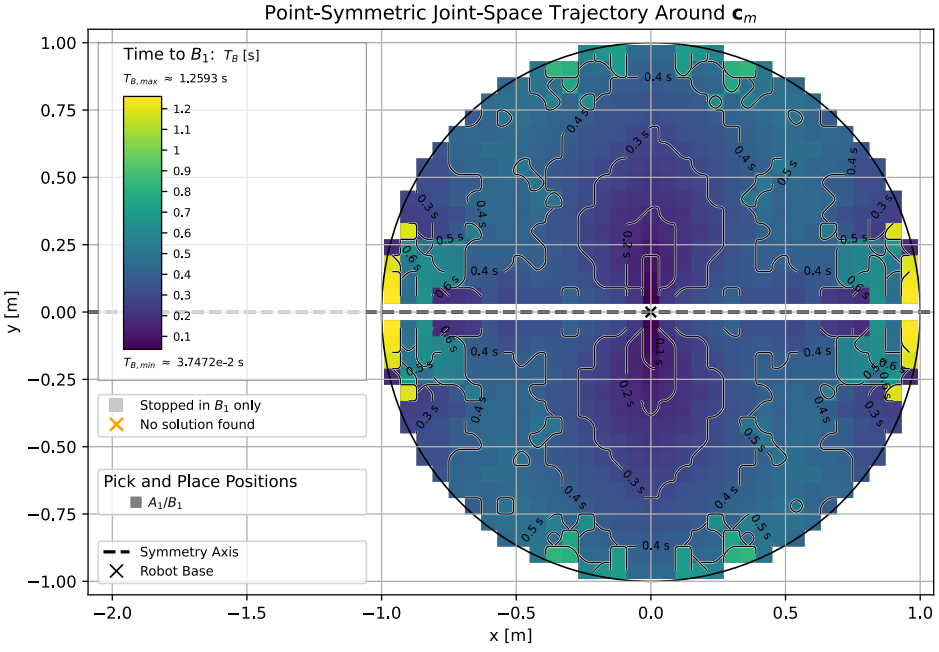
We sample the points A_i with $p_{A_i,x} \geqslant 0$ and $p_{A_i,y} > 0$ in the quarter of the workspace from Section 4.3.2, with B_i as their mirror image about the x-axis. For every sampled position A, the optimization algorithm in Section 5.3.4 finds a point-symmetric trajectory in joint space that stops in both A and its mirrored position B within (5.3). The workspace coverage for these trajectories is shown in Figure 5.15a for the positions A_0 and B_0 while Figure 5.15b shows the successful results for A_1 and B_1.

Figure 5.15 shows the time T_B of the fastest successful result per position A by its colour. Between neighbouring positions A, T_B mostly changes smoothly and only some points stand out with abrupt changes. There, presumably a different $\boldsymbol{c}_m$ than at the neighbouring positions A ended the search first, or the optimization result is an oscillation that oscillates several times along its trajectory. Section 6.1.1 shows such a case.

With an optimization result for every sampled A, the stiffness optimization fulfils its purpose from Section 4.3.3. It shows where point-symmetric oscillations exist, namely everywhere, and provides initial guesses for the clutch switching times optimization in Section 5.4. Initial guesses from neighbouring positions A, as in the raster continuation of Section 5.4.3, could extend this search per centre configuration and configuration pair to get more successful results in this stage. These additional successful results could replace the results chosen in Figure 5.15. This way, abrupt changes of T_B in Figure 5.15 may be replaced by smoother changes in T_B.



(a) A_0/B_0



(b) A_1/B_1

Figure 5.15.: Time T_B to B of the fastest successful result per position A of the sampled workspace for both configuration pairs. Position B is again the mirror image of A about the x-axis.

5.4. Clutch Switching Times Optimization

The clutch switching times optimization implements the third stage from Section 4.3.4 and keeps the stiffness values fixed at $\boldsymbol{k}_{fix}$, which is chosen in Section 5.4.3. To cover as much of the workspace as possible with oscillations for the pick-and-place sequences in Section 5.5, the raster continuation in Section 5.4.3 runs separately for each of the eight rasters. The raster continuation thus has a fixed configuration index for the pick and place positions A_i/B_i and one centre configuration $\boldsymbol{c}_m$ from the set in Theorem 1 throughout the continuation. To regain the freedom lost by not optimizing k_1 and k_2 anymore, the two switching circles from Section 5.1.2 get separate radii r_{c_1} and r_{c_2}, so that the clutches engage and disengage at different times and crossing points $\boldsymbol{q}_{c_i}$. As in Section 5.3, each simulation starts in $\boldsymbol{c}_m$ and the least squares optimizations only aim at stopping in $\boldsymbol{q}_{B_i}$ to find possible point-symmetric oscillations in joint space.

5.4.1. Decision Variables

Figure 5.16 shows the first quarter of an oscillation between A_i and B_i and its continuation. It also displays the decision variables used in the clutch switching times optimization. The dashed circle corresponds to the switching circle for clutch 1. When the trajectory crosses the dashed circle in $\boldsymbol{q}_{c_1}$, only clutch 1 engages or disengages while clutch 2 only engages or disengages when the trajectory crosses the other circle in $\boldsymbol{q}_{c_2}$ as shown in Figure 5.16.

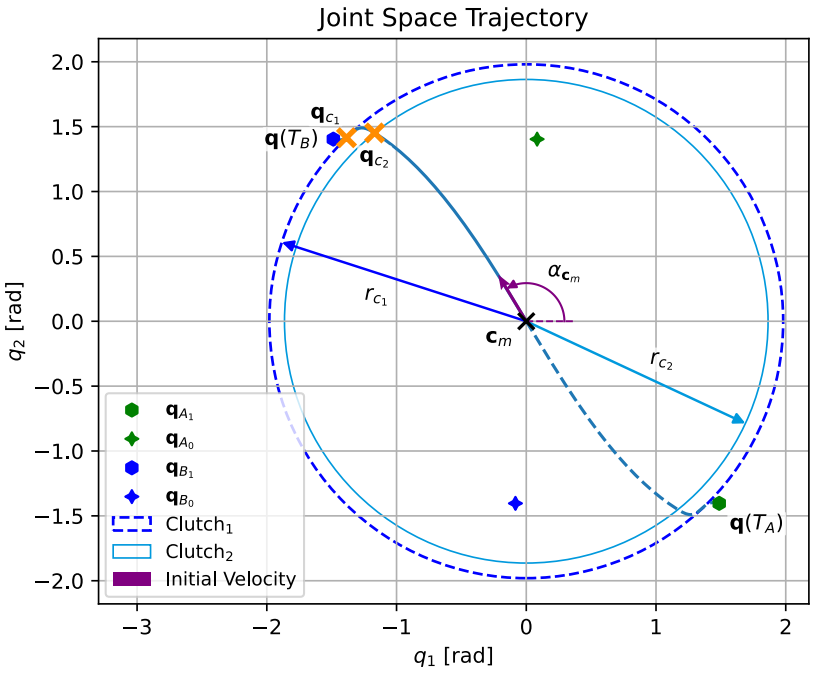


Figure 5.16.: Joint-space trajectory of the clutch switching times optimization for $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ with the switching circles of radii r_{c_1} and r_{c_2}.

As in Section 5.3.1, α_{c_m} from the decision variables $\boldsymbol{z}_{switch}$ in (5.5) is bounded to one turn around its initial guess while T_B only selects the joint state for the cost function as introduced in Sections 5.2.1 and 5.3.1. This time, T_B is bounded to be positive, since the initial guesses

already stop in or near B with a positive T_B.

$$\begin{aligned} \boldsymbol{z}_{switch} &= (\alpha_{c_m}, T_B, r_{c_1}, r_{c_2})^\top , \\ \alpha_{c_m} &\in [\alpha_{c_m,init} - \pi, \alpha_{c_m,init} + \pi]\,\mathrm{rad}\,, \qquad T_B > 0\,, \\ r_{c_1}, r_{c_2} &\in [10^{-3}, \|\boldsymbol{q}_B - \boldsymbol{c}_m\| - 10^{-15}]\,\mathrm{rad} \end{aligned} \tag{5.5}$$

The upper bound r_{max} of r_{c_1} and r_{c_2} keeps $\boldsymbol{q}_B$ outside both switching circles such that at $\boldsymbol{q}_B$ both clutches are engaged. To find more successful results, we tentatively increased r_{max} by reducing its margin to $\|\boldsymbol{q}_B - \boldsymbol{c}_m\|$ from 10^{-5} rad in (5.1) to 10^{-15} rad.

Since the initial guesses from Section 5.4.3 are already good enough, a global sampler is not needed. Instead, the least squares optimizer $\mathrm{LS4}_{switch}$ directly optimizes all four entries of $\boldsymbol{z}_{switch}$. The index of $\mathrm{LS4}_{switch}$ gives the number of decision variables as in Section 5.3.1 and the subscript marks the least squares optimizations of the clutch switching times optimization. In contrast, $\mathrm{LS1}_{switch}$ optimizes T_B within the computed bounds with fixed α_{c_m}, r_{c_1} and r_{c_2}.

5.4.2. Success Criteria

A result $\boldsymbol{z}_{switch}$ is successful if it stops in B and A within the tolerances of (5.3) with $\boldsymbol{z}_{switch}$ in place of $\boldsymbol{z}_{stiff}$, but with the energy tolerance $0.1\,\% E_{total} = 5 \cdot 10^{-3}$ J instead of $5 \cdot 10^{-5}$ J. In addition, a successful result has to return to $\boldsymbol{c}_m$ after one period $T = 4T_B$ within the estimated sensor accuracy of 10^{-4} rad. Otherwise, the motor would first have to bring the robot back to $\boldsymbol{c}_m$ before the impulsive transition between two oscillations can be performed and the transition cost in Section 4.4.1 would neglect this additional motor effort.

5.4.3. Initial Guess

Figure 5.1 shows the four steps that transform the stiffness optimization results into initial guesses for the clutch switching times optimization for fixed stiffness values. The radius continuation in Section 5.4.3 exploits the correlation between the radius r and the stiffness values k_1 and k_2. The stiffness optimization results are continued along r. From the continued results and the stiffness optimization result, Section 5.4.3 chooses $\boldsymbol{k}_{fix}$ and finds support results with $k_1 = k_{1,fix}$ and $k_2 = k_{2,fix}$. These support results are used by the raster continuation in Section 5.4.3 to spread clutch switching times optimization results over all raster nodes.

Radius Continuation

As described in Section 4.3.4, the fixed stiffness values are chosen from the results of the stiffness optimization. Since r and the stiffness values correlate, the radius continuation varies r for each of these results to find further results with different stiffness values as shown in Figure 5.17. The algorithm starts from each stiffness optimization result that fulfils (5.3) with k_1 and k_2 strictly inside their bounds in (5.1). In two separate runs, the radius continuation increases and decreases the common radius $r = r_{c_1} = r_{c_2}$ of both switching circles around $\boldsymbol{c}_m$ from Section 5.1.2 in steps of Δr_{step}. At each new sample $r_{curr} = r \pm \Delta r_{step}$ next to an already solved sample r, $\mathrm{LS4}_{stiff}$ of the stiffness optimization from Section 5.3.4 optimizes α_{c_m}, T_B, k_1 and k_2 with the successful result at r as initial guess. Unlike in the stiffness optimization, $\mathrm{LS4}_{stiff}$ does not adapt r_{curr} within its optimization steps to prevent vanishing gradients since we want the sample at the specified radius r_{curr}. The sampling starts with $\Delta r_{step} = 10^{-2}$ rad. Δr_{step} is changed by factors of 10 between the largest step width 10^{-2} rad and the smallest step width 10^{-6} rad. If the result at r_{curr} does not stop in B within (5.3a), Δr_{step} is reduced by a factor of 10 when it has not reached its smallest step width. r_{curr} is then adapted to a value

that is now Δr_{step} away from the previous r from which the result stopped in B. Each solved sample increases Δr_{step} again by a factor of 10 until its highest value 10^{-2} rad is reached. Each run stops when the result at $\Delta r_{step} = 10^{-6}$ rad does not stop in B or r_{curr} leaves the bounds of r in (5.5).

For each stiffness optimization result, SciPy's `PchipInterpolator` [33] then interpolates over r with piecewise cubic Hermite polynomials between all solved samples that stop in B and yields the function $\boldsymbol{z}(r) = (\alpha_{c_m}, T_B, k_1, k_2)^\top$ as shown in Figure 5.17. Figure 5.17 shows the

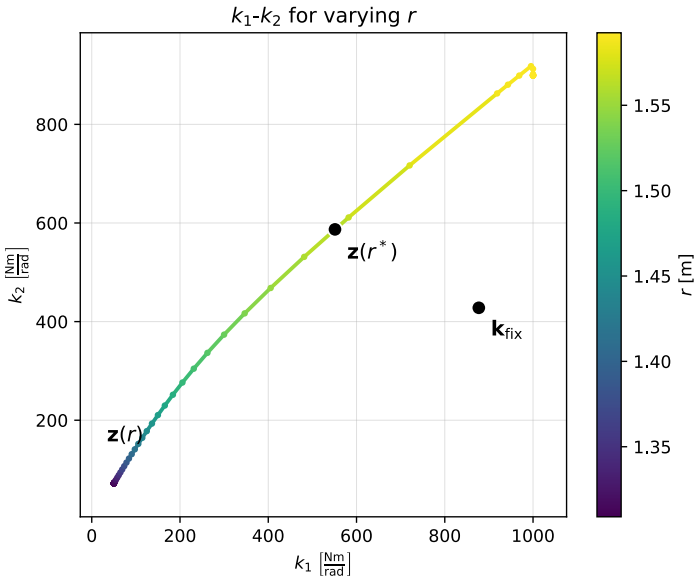


Figure 5.17.: Stiffness values k_1 and k_2 along the continuation over r of one stiffness optimization result for $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ and the chosen fixed stiffness $\boldsymbol{k} = \boldsymbol{k}_{fix}$.

stiffness values $\boldsymbol{k}(r) = (k_1(r), k_2(r))$ of $\boldsymbol{z}(r)$ for one stiffness optimization result of $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ where the values of r are displayed in different colours. It can be observed that both stiffness values increase with increasing radius r. A larger circle thus lets the clutches engage later on the way to $\boldsymbol{q}_B$, so that the springs are deflected less in $\boldsymbol{q}_B$ and need higher stiffness values to stop in B. Near the stiffness bounds, the curve is slightly distorted, as can be seen at $k_1 = 1000\,\mathrm{N\,m/rad}$. For a fixed stiffness $\boldsymbol{k}_{fix} = (k_{1,fix}, k_{2,fix})$, we take the point $\boldsymbol{z}(r^*)$ with the smallest relative distance $d_k(r)$ in (5.6) between the stiffnesses of $\boldsymbol{z}(r)$ and $\boldsymbol{k}_{fix}$. Starting from the sampled radius with the smallest $d_k(r)$, SciPy's `minimize_scalar` finds the radius r^* with $d_k(r)$ as cost function by a bounded minimization with Brent's method [13, Chapter 5]. Since the distance $d_k(r)$ divides the deviation of each stiffness value in $\boldsymbol{k}(r)$ by its value in $\boldsymbol{k}_{fix}$, deviations are measured relative to the fixed stiffness. Replacing k_1 and k_2 of $\boldsymbol{z}(r^*)$ by $k_{1,fix}$ and $k_{2,fix}$ converts it into an initial guess for the clutch switching times optimization with $\boldsymbol{z}_{switch} = (\alpha_{c_m}(r^*), T_B(r^*), r^*, r^*)$ at $\boldsymbol{k}_{fix}$.

$$d_k(r) = \max\left(\frac{|k_1(r) - k_{1,fix}|}{k_{1,fix}}, \frac{|k_2(r) - k_{2,fix}|}{k_{2,fix}}\right) \tag{5.6}$$

Stiffness Selection

Our goal is to choose k_1 and k_2 for the constant torsional springs of the CPEA but not to optimize for the best stiffness values. For this, we only consider results that stop in A and its x-axis mirrored point B within 1 mm and with a kinetic energy tolerance of $5 \cdot 10^{-4}$ J. Out of those

results, we form a subset in which A lies within the Cartesian circle of 0.6 m around the robot base with $p_{A,x} > 0.1\,\text{m}$. From this subset of results we collect all combinations of stiffness values $\boldsymbol{k}(r)$ at the sampled r in the continuation results. These collected stiffness values are tested as $\boldsymbol{k}_{fix}$. For each tested $\boldsymbol{k}_{fix}$, we determine the initial guesses $\boldsymbol{z}_{switch}$ from Section 5.4.3 of all curves $\boldsymbol{z}(r)$ where the resting position A lies within the defined subset. For each position A, we simulate $\boldsymbol{z}_{switch}$ at sample r^* for each curve $\boldsymbol{z}(r)$ whose successful results stop in A. The tested stiffness values for which the most positions A have at least one simulation that successfully stops in A and B within 1 mm and an energy tolerance of $5 \cdot 10^{-4}\,\text{J}$ become the chosen $\boldsymbol{k}_{fix}$ for the springs. For our dataset, we chose $\boldsymbol{k}_{fix} = (877.1211009026853, 427.99566297584414)^\top\,\text{N m/rad}$.

Support Results

For all stiffness optimization results, we create the curve $\boldsymbol{z}(r)$ in the radius continuation in Section 5.4.3 and determine the initial guesses $\boldsymbol{z}_{switch}$ at $\boldsymbol{k}_{fix}$. Each initial guess $\boldsymbol{z}_{switch}$ is used to optimize for the clutch switching times in the algorithm described in Section 5.4.4. All clutch switching times optimization results that fulfil the success criteria defined in Section 5.4.2 are then considered as support results for the raster continuation in Section 5.4.3.

Raster Continuation

The rasters from Section 4.3.4 place the positions for A on a Cartesian grid with a step size of 0.02 m. To obtain good initial guesses for the optimization algorithm in Section 5.4.4, we use a continuation method over neighbouring raster nodes for each raster. The continuation only covers the raster nodes in the quarter of the workspace from Section 4.3.2, where the neighbours of a raster node are the adjacent raster nodes in the x and y direction.

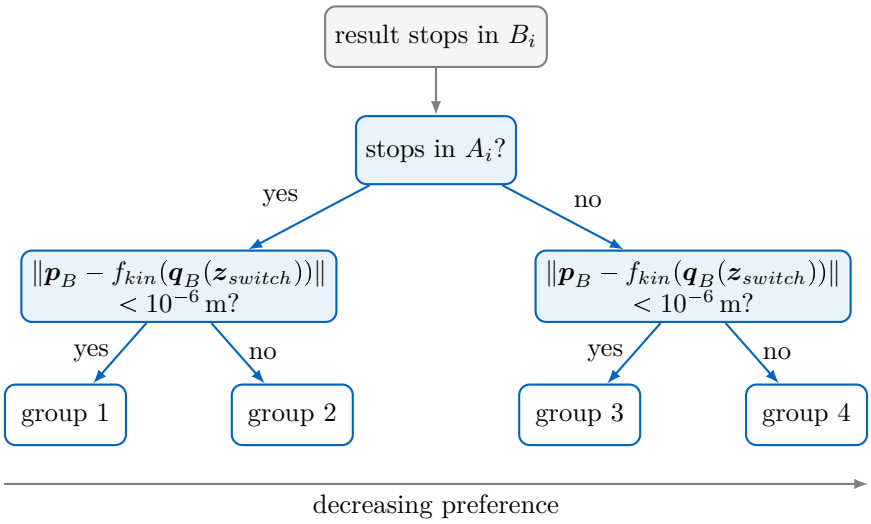


Figure 5.18.: The algorithm to group the members of a raster node. Stopping in A_i and B_i refers to the success criteria of Section 5.4.2. (AI-generated illustration)

All members of a raster node are deduplicated when they describe the same trajectory. This is the case if the difference of their initial velocity angles α_{c_m}, wrapped to $[-\pi, \pi)$, is at most 10^{-3} rad and if T_B, r_{c_1} and r_{c_2} agree within a relative tolerance of 10^{-3} plus an absolute tolerance of 10^{-3} s for T_B and 10^{-6} rad for the radii. The deduplicated members of a raster node that stop in B within 1 mm can then be sorted into the four groups displayed in Figure 5.18. Members that belong to group 1 are considered better than those of group 2. In general, the members of the group with the lower group number are considered better than those with the

higher group number in Figure 5.18. Within a group, the lowest sum of the costs $\frac{1}{2}\|\boldsymbol{s}_A\|^2$ and $\frac{1}{2}\|\boldsymbol{s}_B\|^2$ at A and B from (5.4) decides which member of this group is considered the best.

Each raster continuation only considers support results from Section 5.4.3 that start in the raster's specified $\boldsymbol{c}_m$ and stop in $\boldsymbol{q}_{B_i}$ at the raster's specified joint configuration index i. The selected support results are sorted into the fitting raster nodes. To determine the support raster nodes, displayed in orange in Figure 5.19, we compare the best members of each raster node with the grouping algorithm in Figure 5.18. The raster nodes whose best member belongs to the best group from all support results are considered support raster nodes. From those support raster nodes, the optimization results are spread wave-like as initial guesses over all possible raster nodes. This is done in different iterations where each iteration defines the value n_{initial}.

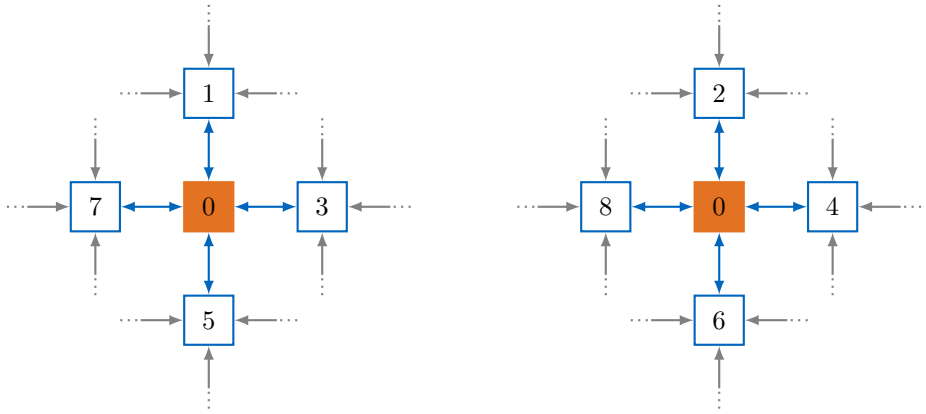


Figure 5.19.: Order in which the neighbours of two support nodes (orange) are processed for $n_{\text{initial}} = 1$ if no better result arrives in between. (AI-generated illustration)

As Figure 5.19 displays, each raster node passes its members as initial guesses to the adjacent raster node. From all members of the raster node, the best-ranked member that has not already served as initial guess is passed to the adjacent raster node. This is done until the best n_{initial} members of the raster node already served as initial guesses to the neighbour. To determine $\boldsymbol{q}_{B_i}$ for the optimization of the neighbour, we represent it relative to the configuration from the initial guess since this representation has the shortest joint-space distance to that configuration. This is a better initial guess since the cost function compares the joint angles directly and does not treat joint angles differing by multiples of 2π, such as π and $-\pi$, as equal. $\boldsymbol{q}_{A_i}$ is then again determined by $\boldsymbol{q}_{B_i}$ and $\boldsymbol{c}_m$. Since a raster node can also get new members over time through initial guesses from neighbours, such a member can be ranked as one of the best n_{initial} members and serve as initial guess to other raster nodes. This way, a raster node can collect more than n_{initial} members in one iteration.

In each iteration, the algorithm starts at the support raster nodes, represented as orange nodes with number 0 in Figure 5.19 and passes their members to the neighbouring raster nodes as initial guesses in the order of the increasing numbers in Figure 5.19. After the support raster nodes, the algorithm processes raster nodes that lie closest to the support raster node first in the order displayed in Figure 5.19 and determines which members of the processed raster node were not already passed as initial guess to their neighbours. As the arrows in Figure 5.19 show, the support raster nodes also get initial guesses from their neighbours. This way, a raster node for which an optimization failed is tried again with the next initial guesses of its neighbours and the best members of a raster node can be continuously improved.

The first iteration of the algorithm starts with $n_{\text{initial}} = 1$. Once every node has passed on its

best n_{initial} members, we increase n_{initial} by one and continue. As soon as every raster node in the sampled workspace has members ranking in group 1 of Figure 5.18, only members belonging to group 1 or 2 are passed to their neighbours.

5.4.4. Optimization Algorithm

Both least squares optimizations use the same least squares algorithm implemented in SciPy as the no-spring optimization in Section 5.2.3, but with different decision variables (Section 5.4.1) and a different cost function (Section 5.4.5). Figure 5.20 shows how this algorithm optimizes an initial guess $\boldsymbol{z}_{switch}$. Since B is reached at the times $T_B + n_{multiple}T$ with $n_{multiple} \in \mathbb{N}$ and we optimize for the first stop in B, we set T_B of $\boldsymbol{z}_{switch}$ to $t_{both} + 5\,dt$. This is done to prevent optimizing for a stop in B where the trajectory also stopped at B at a time t with $0 < t < T_B$. t_{both} is the time at which both clutches are engaged for the first time and thus the entry time $t_{\text{entry},i}$ of the clutch with the larger radius r_{c_i}. The buffer of $5\,dt$ should keep the first evaluations of LS4_{switch} from sampling the joint state at a time where one clutch is not engaged and its gradient vanishes. We abort an optimization if the simulations used for evaluation are

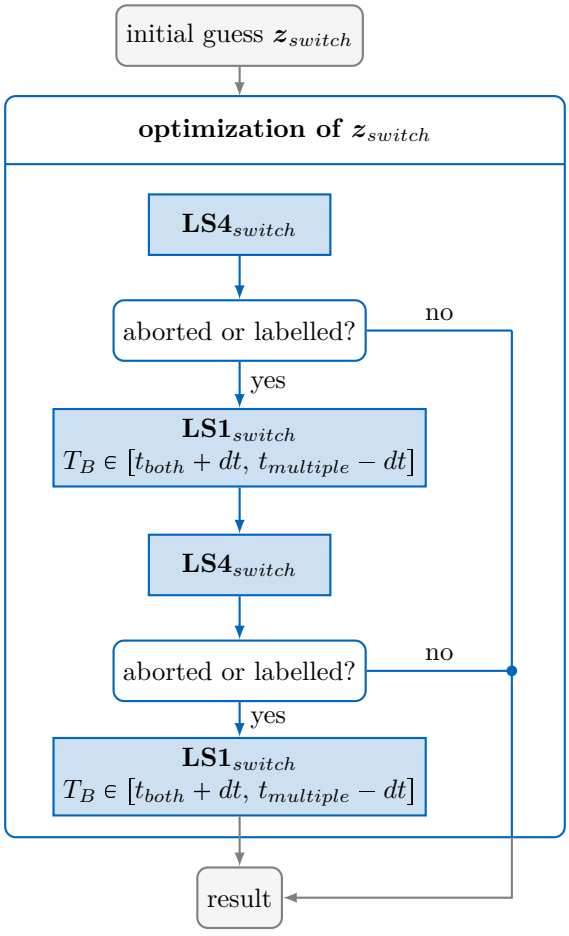


Figure 5.20.: Optimization of an initial guess $\boldsymbol{z}_{switch}$ in the clutch switching times optimization (AI-generated illustration)

labelled for the same reason in 10 consecutive evaluations. A simulation is labelled for two reasons where one of them is that at least one clutch has not engaged until T_B, so that the gradient with respect to its radius vanishes. Another reason for labelling is that the trajectory $\boldsymbol{q}(t)$ for $t \leqslant T_B$ enters the region where both clutches are engaged, swings to an area where

both clutches are disengaged and then engages one of the clutches again. Such a trajectory is not desired, since we do not stop in $\boldsymbol{q}_B$ on the direct way from $\boldsymbol{c}_m$.

For an aborted optimization or a labelled result, $\mathrm{LS1}_{switch}$ keeps α_{c_m}, r_{c_1} and r_{c_2} of the result of $\mathrm{LS4}_{switch}$ fixed and optimizes T_B. $\mathrm{LS1}_{switch}$ uses $t_{both} + 5\,dt$ as an initial guess and sets the bounds of T_B to $[t_{both} + dt, t_{multiple} - dt]$. $t_{multiple}$ is the first time at which a clutch engages again after both were disengaged at $t > t_{both}$. The interval $[t_{both} + dt, t_{multiple} - dt]$ ensures that no $\mathrm{LS1}_{switch}$ optimization result is labelled by construction. The $\mathrm{LS1}_{switch}$ optimization result is given as an initial guess to $\mathrm{LS4}_{switch}$ for optimization. If $\mathrm{LS4}_{switch}$ is aborted or its result labelled again, $\mathrm{LS1}_{switch}$ finds an unlabelled result for the last time and returns this as a clutch switching times optimization result.

After optimizing $\boldsymbol{z}_{switch}$ with $T_B = t_{both} + 5\,dt$, we check whether the returned result stops in A and B within the tolerances of Section 5.4.2. Otherwise, we run the whole algorithm in Figure 5.20 again with the original T_B of $\boldsymbol{z}_{switch}$ from the initial guess.

5.4.5. Cost Function

$\mathrm{LS1}_{switch}$ and $\mathrm{LS4}_{switch}$ minimize cost_{stiff} from (5.4) with $\boldsymbol{z}_{switch}$ in place of $\boldsymbol{z}_{stiff}$ and keep its scaling with ϵ_q and $\epsilon_E = 5 \cdot 10^{-5}\,\mathrm{J}$. The cost function for these oscillations is thus cost_{switch} in (5.7). This stage still optimizes with the stricter tolerances of the stiffness optimization although the success criteria in Section 5.4.2 accept optimization results of up to $5 \cdot 10^{-3}\,\mathrm{J}$.

$$\mathrm{cost}_{switch} = \frac{1}{2}\|\boldsymbol{s}_B\|^2 = \frac{1}{2}\left(\frac{\|\boldsymbol{q}_B(\boldsymbol{z}_{switch}) - \boldsymbol{q}_B\|^2}{\epsilon_q^2} + \frac{2E_{kin}(\boldsymbol{z}_{switch})}{2\epsilon_E}\right) \tag{5.7}$$

5.5. Pick-and-Place Sequence Optimization

For each centre configuration $\boldsymbol{c}_m$ and placement mode, we solve the GTSP from Section 4.4.2 with the transition cost as inter-node link cost $\|\Delta\boldsymbol{\rho}\|_1$ by implementing the recursion (3.40) and the closing link (3.41) of Section 3.5 directly. The GTSP only uses raster node members that fulfil the success criteria of Section 5.4.2 as GTSP node members. With the grid centre $\boldsymbol{p}_{grid}$ and $\boldsymbol{c}_m$ defining the GTSP node n_1, we compute the lowest sequence cost for every member s_j of n_1 as start and select the member of n_1 with the lowest sequence cost. When several have the same sequence cost within a relative tolerance of 10^{-12}, we select the pick-and-place sequence with the shortest sequence duration as explained in Section 4.4.2.

Since the positions from the axis-aligned 3×3 task set with $d_x = d_y = 6\,\mathrm{cm}$ lie on the workspace sampled every 2 cm in the clutch switching times optimization, all nine GTSP nodes already have their members from the merged raster nodes and we do not have to find other oscillations. However, for the radially aligned 3×3 task set, there can be GTSP nodes for which no raster node exists, since the pick and place positions A and B do not lie on the sampled raster. On such an off-raster position lies an off-raster node. This off-raster node is similar to a raster node defined by the off-raster position, configuration index and $\boldsymbol{c}_m$.

The off-raster position lies between four raster positions as shown in Figure 5.21. At least one of the raster positions lies within the workspace and is valid. For all valid raster positions, there is a raster node identified by the same $\boldsymbol{c}_m$ and configuration index as the missing off-raster node. Members of these raster nodes are offered as initial guesses to the off-raster node when the member is a trajectory that stops in A and B. The targeted configuration representation of $\boldsymbol{q}_{B_i}$ in a member of the off-raster node is determined by representing it relative to the configuration

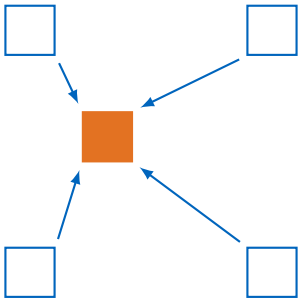


Figure 5.21.: Off-raster node (orange) that receives initial guesses from the raster nodes (blue) at the four neighbouring raster positions. (AI-generated illustration)

of the initial guess. $\boldsymbol{q}_{A_i}$ is then determined as the point-symmetric configuration of $\boldsymbol{q}_{B_i}$ about the centre configuration $\boldsymbol{c}_m$. First, the off-raster node collects all configuration representations for its members for which initial guesses exist. For each configuration representation on the off-raster position, we select the raster node which has the most initial guesses with the shortest joint-space distance to the configuration representation. These initial guesses are then used for the optimization in Section 5.4.4. The optimization result is stored as a GTSP node member if it is successful according to the success criteria of the clutch switching times optimization. With this off-raster continuation, all GTSP nodes have members if such pick-and-place oscillations were found in the optimization.

6 Evaluation

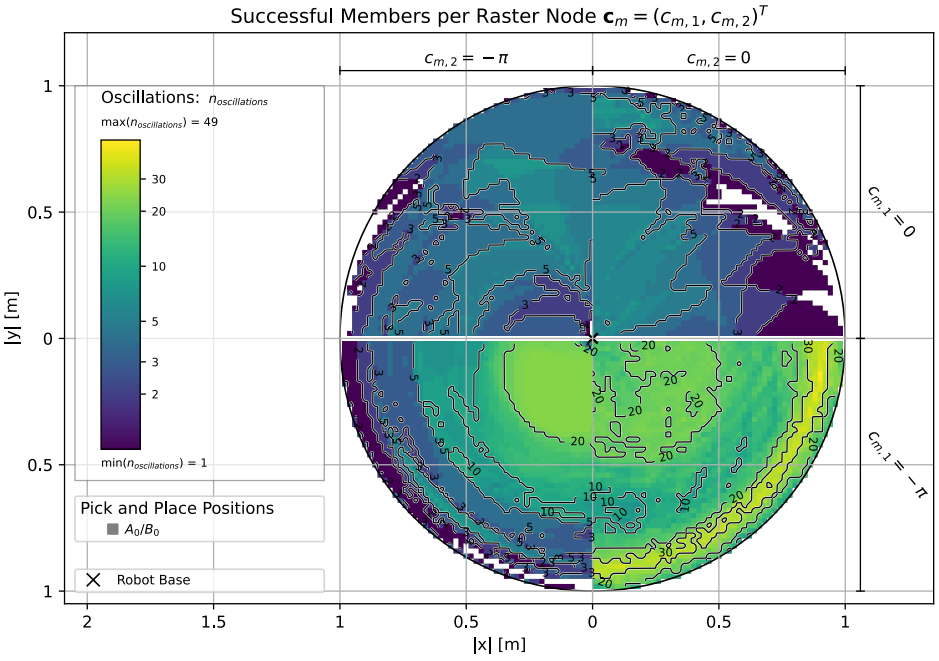
In Chapter 5, the optimization pipeline of Figure 5.1 applies the methods from Chapter 4. It yields a variety of oscillations as pick-and-place motions. Based on these oscillations, we use the GTSP to analyse the found conservative pick-and-place sequences fulfiling the 3×3 task set defined in Section 4.4.2. This chapter presents the results of the workspace analysis for single tasks in Section 6.1 and of the closed 3×3 pick-and-place sequences in Section 6.2. On this basis, Chapter 7 then answers the research question from Section 1.1.

6.1. Workspace Coverage for Single Tasks

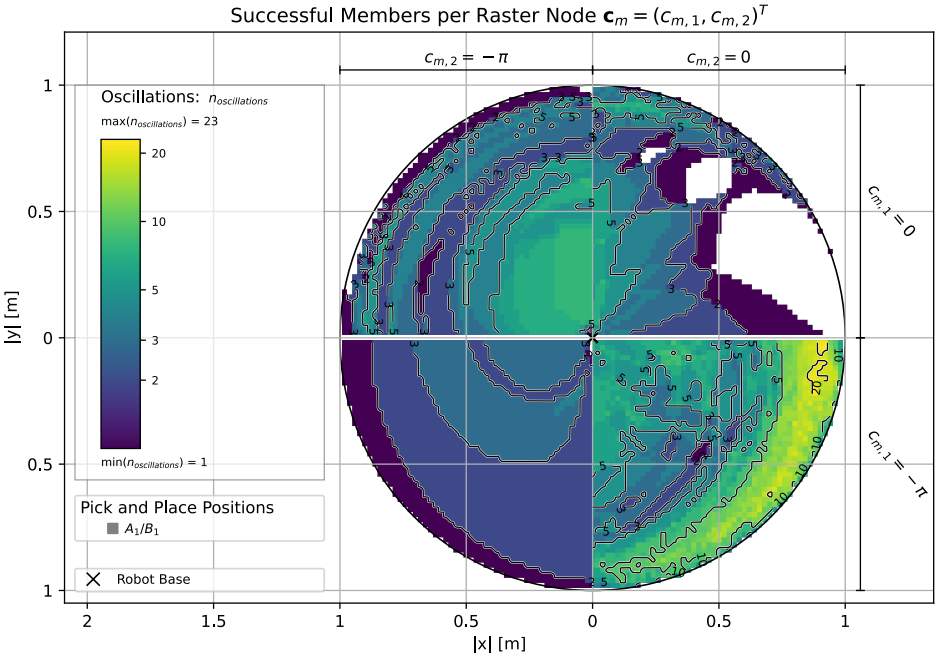
In this section, we evaluate the successful members of each raster node from Section 5.4.3 for each of the eight rasters. A successful member is the found point-symmetric oscillation in joint space. We first examine how many oscillations exist per raster node in Section 6.1.1. Then we look at the fastest oscillations per raster node in Section 6.1.2 and evaluate the accuracy of the found oscillations in Section 6.1.3.

6.1.1. Multiple Oscillations per Configuration Pair

In the clutch switching times optimization, a raster node can collect several successful members during the raster continuation. Figure 6.1 shows for each raster the raster nodes with at least one successful member, where Figure 6.1a contains the number of oscillations between the configuration pair $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ per centre configuration $\boldsymbol{c}_m$ and Figure 6.1b the number of oscillations between $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ per $\boldsymbol{c}_m$. Each quadrant of the figures shows which of the four centre configurations $\boldsymbol{c}_m$ from Theorem 1 was used as the starting point of the point-symmetric oscillation. Many raster nodes have more than two oscillations. For $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$, the quadrant of $\boldsymbol{c}_m = (-\pi, -\pi)$ in Figure 6.1b shows a fairly consistent picture with mostly two oscillations per raster node, marking the raster as one with fewer successful members per raster node in comparison to other rasters. At the border of the workspace near singularities, one can also observe that there are fewer oscillations per raster node. The white regions mark raster nodes where our raster continuation method found no successful member. As such, there are large areas with no oscillation per raster node, specifically for $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ with $\boldsymbol{c}_m = (0, 0)$ in Figure 6.1b. For some rasters, like $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ with $\boldsymbol{c}_m = (-\pi, -\pi)$, there are very small regions with no oscillation even though neighbouring raster nodes have oscillations. Those could come from numerical noise as we filtered for oscillations that are within 10^{-4} rad at $\boldsymbol{c}_m$ after a period. It is also interesting to note that the raster nodes near the white regions have fewer oscillations.



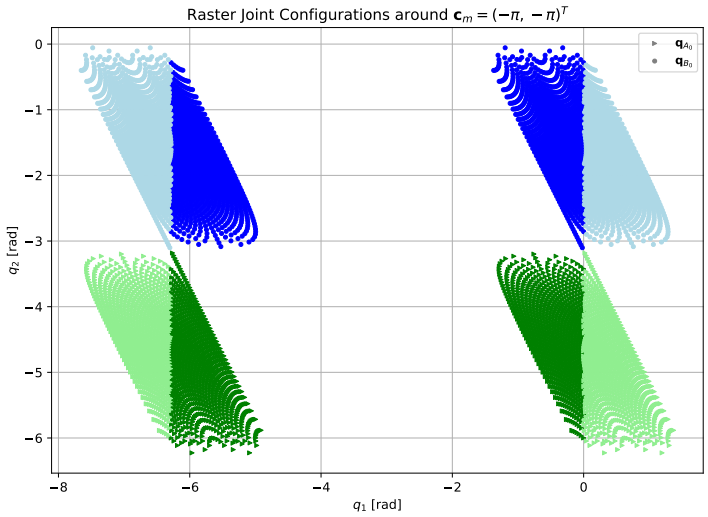
(a) A_0/B_0



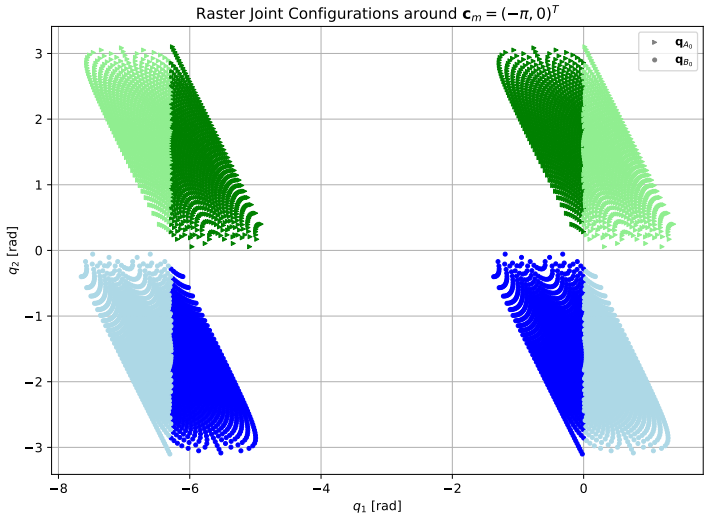
(b) A_1/B_1

Figure 6.1.: Number of oscillations per raster node for both configuration pairs. Each quadrant of the figures belongs to one centre configuration $\boldsymbol{c}_m$.

For the configuration pair $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$, there are regions with more than 20 oscillations per raster node for $\boldsymbol{c}_m = (-\pi, -\pi)$ and $\boldsymbol{c}_m = (-\pi, 0)$. For these two rasters, one can see in Figure 6.2 that



(a) $\boldsymbol{c}_m = (-\pi, -\pi)$



(b) $\boldsymbol{c}_m = (-\pi, 0)$

Figure 6.2.: Joint configuration representations of the rasters with configuration index 0 and $c_{m,1} = -\pi$ for which the raster continuation optimized.

through the raster continuation in Section 5.4.3, there are inverse kinematics solutions of the same configuration index with more than one representation. The light blue and light green are configurations which are not in the unique representation in reference to $\boldsymbol{c}_m$ from Definition 10. In contrast, the dark colours are the unique representations relative to $\boldsymbol{c}_m$. The stiffness optimization results are trajectories and oscillations stopping in the unique configurations of the pick and place positions represented relative to $\boldsymbol{c}_m$. Since the support results that served as initial guesses in the raster continuation stem from those stiffness optimization results, the configurations of those initial guesses are also in the representation relative to $\boldsymbol{c}_m$. For the two discussed rasters, which are identified by $c_{m,1} = -\pi$ and configuration index 0, the unique configuration representations in reference to $\boldsymbol{c}_m$ of the sampled Cartesian points are spread across two different sides of $\boldsymbol{c}_m$ as shown in dark blue and dark green in Figure 6.2. There are

thus jumps of $\sim 2\pi$ between the configurations represented relative to $\boldsymbol{c}_m$. To get better initial guesses through neighbouring configurations in joint space, we specifically implemented the raster continuation such that the configurations were uniquely represented with reference to the configuration of the initial guess. This led to the configurations in light blue and green in Figure 6.2. Due to this, the oscillations can be found in two directions. In Figure 6.2a, they can swing between top-left and bottom-right or between top-right and bottom-left. For other rasters than those two discussed above, all raster joint configurations are in the unique representation relative to $\boldsymbol{c}_m$ and there is only one direction in which the oscillations can swing. This could be the reason why the raster nodes of the two rasters in Figure 6.2 contain more oscillations than most of the other rasters. From the other rasters, only the raster with configuration index 1 and $\boldsymbol{c}_m = (-\pi, 0)$ contains raster nodes with over 20 oscillations in Figure 6.1b.

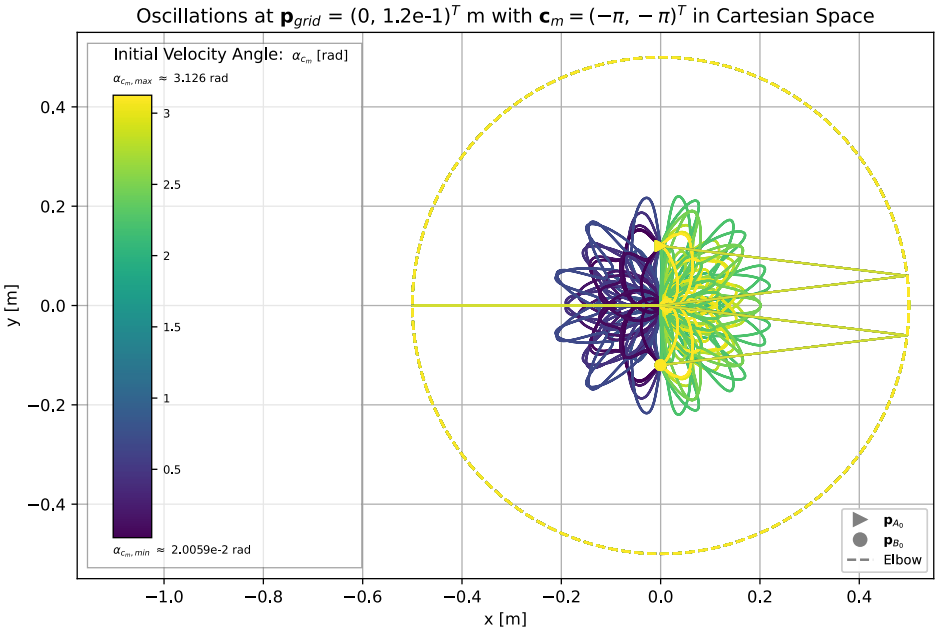
Configuration Pair	Centre Configuration $\boldsymbol{c}_m$	Oscillations per Raster Node			Raster Nodes with Oscillations [%]	
		Mean	Max	Min	> 1	$\geqslant 1$
A_0/B_0	$(0, 0)$	3.36	9	0	76.5	94.6
	$(0, -\pi)$	3.88	8	0	93.3	97.0
	$(-\pi, 0)$	18.89	49	7	100.0	100.0
	$(-\pi, -\pi)$	9.15	24	0	89.8	98.2
A_1/B_1	$(0, 0)$	2.09	8	0	54.7	73.1
	$(0, -\pi)$	3.98	8	0	89.3	97.9
	$(-\pi, 0)$	7.45	23	1	99.5	100.0
	$(-\pi, -\pi)$	2.13	4	0	83.3	99.8

Table 6.1.: Oscillations per raster node for each configuration pair and centre configuration $\boldsymbol{c}_m$.

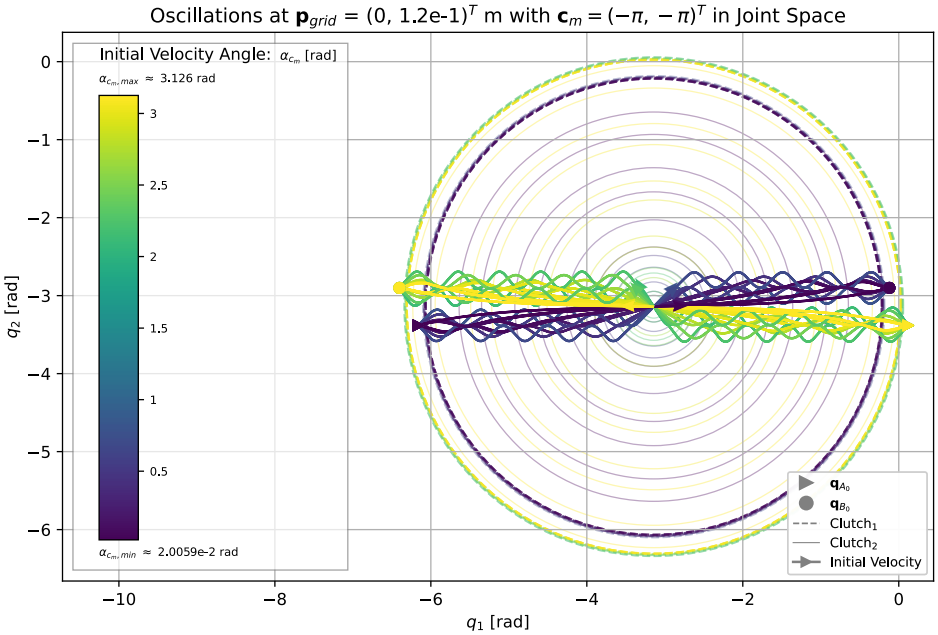
Table 6.1 confirms the observation that the rasters of $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ with $c_{m,1} = -\pi$ and $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ with $\boldsymbol{c}_m = (-\pi, 0)$ have on average more oscillations per raster node than all the other rasters. Their mean value is over 7 oscillations per raster node and their maximum number of oscillations per raster node varies between 23 and 49. In particular, $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ with $\boldsymbol{c}_m = (-\pi, 0)$ has by far the highest mean value per raster node of all rasters. Apart from these three, most rasters reach a maximum of about eight oscillations per raster node. The only exception is the raster with $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ and $\boldsymbol{c}_m = (-\pi, -\pi)$, which reaches a maximum of four. The table additionally shows that in most rasters more than 80 % of the raster nodes have more than one oscillation and more than 90 % of the raster nodes have at least one oscillation. For rasters with $\boldsymbol{c}_m = (-\pi, 0)$ all raster nodes have at least one oscillation. Across all rasters, the raster nodes have on average 6.37 oscillations per configuration pair and centre configuration.

Figure 6.3 shows the raster node with the most oscillations in the raster of $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ at $\boldsymbol{c}_m = (-\pi, -\pi)$. Due to the raster continuation, there are two configuration representations of the same configuration as shown in the joint space plot of the found oscillations (Figure 6.3b). As discussed before, the oscillations can then swing in two directions, from top-left to bottom-right or from top-right to bottom-left. Between the configurations that are not represented relative to $\boldsymbol{c}_m$, there are multi-turn oscillations as q_1 turns over 2π. In contrast, the configurations in their unique representation relative to $\boldsymbol{c}_m$ have single-turn oscillations between them. Figure 6.3a displays the oscillations in Cartesian space, where the dashed circle marks the trajectory of the elbow and the straight lines show the links in their configurations at $\boldsymbol{c}_m$, A and B. The oscillations between the configurations represented relative to $\boldsymbol{c}_m$ are the ones displayed in blue and violet that pass through the region of negative x. The other oscillations swing in

the region of positive x. Figure 6.3b further shows that for this raster node the multi-turn and single-turn oscillations behave alike. In both cases, one oscillation swings directly to its target. The other successful members of the raster node contain more and more oscillations along the trajectory, increasing the period time of the oscillation between A and B. The initial velocity angle α_{c_m} varies widely across the different oscillations, so that this raster node offers a large choice of different momenta at $\boldsymbol{c}_m$ which can be exploited to lower the transition cost.

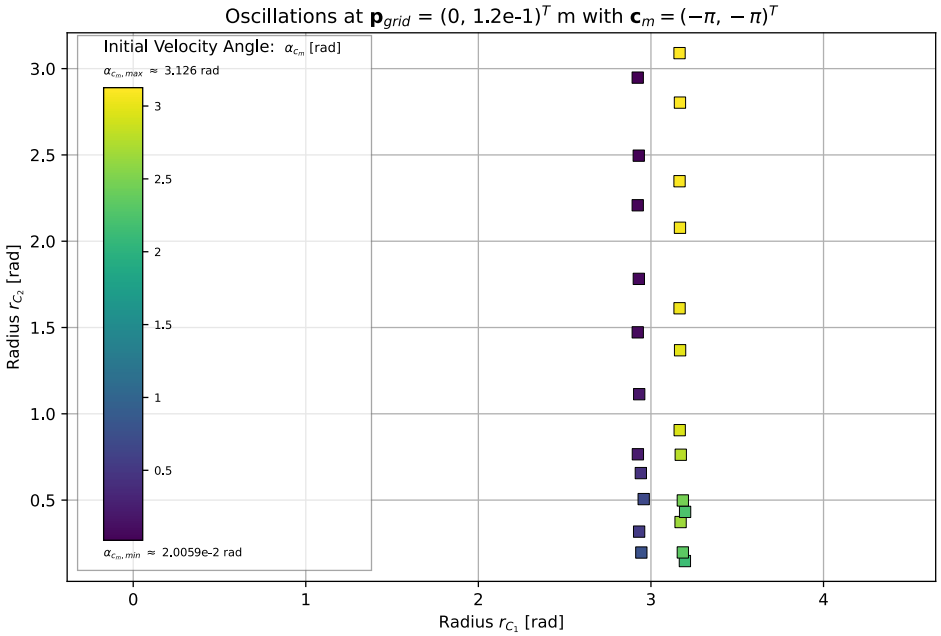


(a) Cartesian Space



(b) Joint Space

Figure 6.3.: All 24 oscillations for the configuration pair $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ and $\boldsymbol{c}_m = (-\pi, -\pi)$. The colour encodes the initial velocity angle α_{c_m} at $\boldsymbol{c}_m$.



(c) Parameters

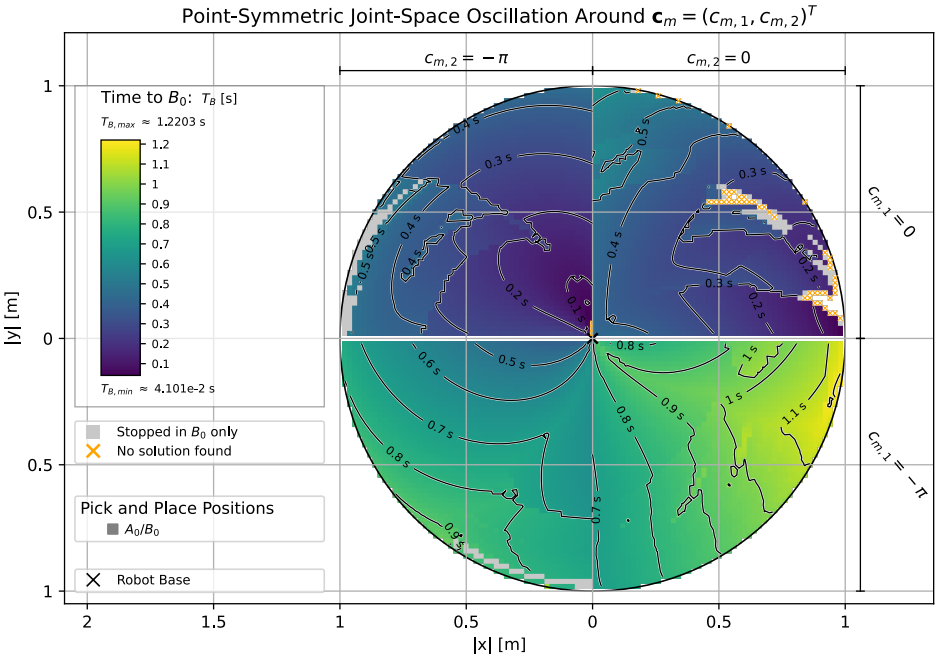
Figure 6.3.: All 24 oscillations for the configuration pair $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ and $\boldsymbol{c}_m = (-\pi, -\pi)$. The colour encodes the initial velocity angle α_{c_m} at $\boldsymbol{c}_m$.

Figure 6.3c shows the optimized parameters over the two clutch radii r_{c_1} and r_{c_2}, with the colour indicating the initial velocity angle α_{c_m}. The time T_B is not displayed, since it is only a decision variable for the least squares optimizations to calculate its cost from and keep the cost function differentiable. T_B thus does not change the shape of the trajectory or oscillation. For this raster node, r_{c_1} seems to only distinguish the multi-turn oscillations in q_1 from the single-turn ones, while r_{c_2} seems to determine how much the trajectory of the successful member oscillates along its way to B. This observation suggests that the multiple oscillations per raster node systematically depend on the optimization parameters in parameter space.

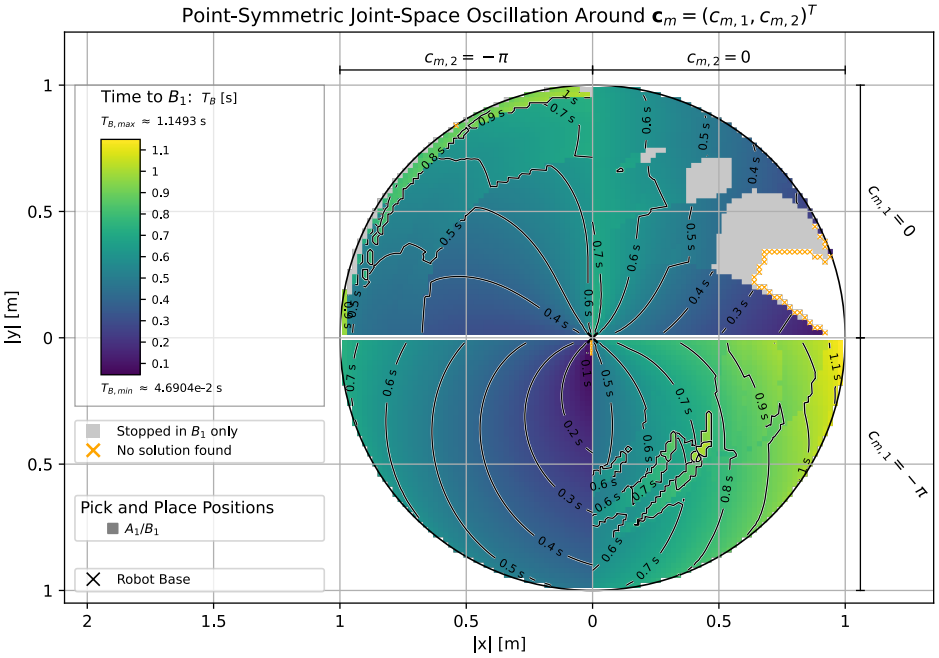
The multiple oscillations per raster node can explain the efficiency of our raster continuation method from Section 5.4.3, as each raster node provides its neighbours with several initial guesses for the optimization. Multiple oscillations also show that the used cost function can have multiple local minima for one raster node that produce successful results.

6.1.2. Period Times

Since we optimize to stop in B at $t = T_B$, Figure 6.4 shows the minimum time T_B of an oscillation per raster node. The period of the whole oscillation is then $T = 4T_B$. For $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$, the fastest period times are for rasters with $c_{m,1} = 0$ and longest for $c_{m,1} = -\pi$. For $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$, the faster oscillations lie at the raster nodes with $\boldsymbol{c}_m = (-\pi, -\pi)$. In this raster, the fastest times change smoothly between neighbouring raster nodes. The contour lines show that not all rasters have a smooth change in T_B across their raster nodes. Some rasters have regions which show sudden jumps to slightly longer or shorter times. We suspect that the oscillation shape changes between two neighbouring raster nodes at these jumps. An oscillation whose trajectory connects A and B directly and used as initial guess. The resulting oscillation from the initial guess at the neighboring position, oscillates along the trajectory and has thus a different shape then the initial guess. As shown through the multiple oscillations per raster



(a) $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$



(b) $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$

Figure 6.4.: Time T_B to stop in B of the fastest oscillation per raster node for all rasters. Each quadrant of the figures belongs to one centre configuration $\boldsymbol{c}_m$.

node in Section 6.1.1, there are multiple local minima with successful results. To prevent oscillations that stop in B only after several intermediate oscillations between A and B, we did not continue T_B from the neighbouring raster node in the optimization algorithm of the clutch switching times optimization. Instead, we initialized it with the first time t_{both} at which both clutches are engaged and added five MuJoCo time steps dt (Section 5.4.4). We continued T_B from the initial guess only when the optimizer did not find a successful result. Not continuing T_B in the first try in the optimization might have led the optimization to find another cost minimum where the oscillation's shape is different from the oscillation or trajectory shape of the initial guess. However, it is also possible that there is no oscillation at this raster position that has a similar shape to the initial guess.

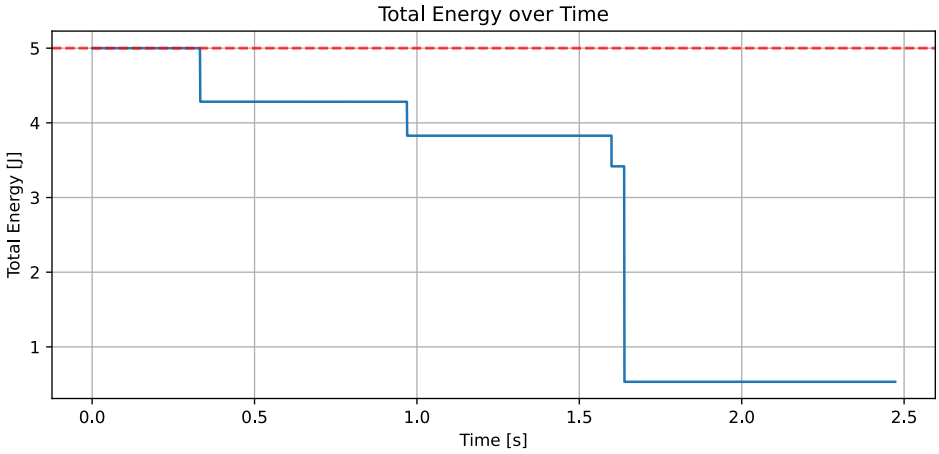
Near singularities, such as for $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ at $\boldsymbol{c}_m = (0, -\pi)$ in Figure 6.4b, these jumps between neighbouring raster positions at T_B occur particularly often. Even though we sampled the raster in Cartesian space with even step sizes, in joint space near singularities these sampled positions lie further apart and the joint-space distances between neighbouring sampled positions may be uneven. Near singularities, the configuration of the initial guess is further away and the optimization might have found another local minimum with an oscillation shaped differently from the initial guess. In addition, raster nodes near singularities have fewer oscillations from which the fastest T_B is chosen. Another reason for the jumps in T_B between neighboring raster nodes might be trajectories with similar shape to the one in the initial guess, which have been filtered out in the success criteria of the clutch switching times optimization (Section 5.4.2).

For single tasks, pick-and-place oscillations exist across the whole workspace if we do not distinguish between centre configurations and configuration pairs. Most of the fastest oscillations come from $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ at $\boldsymbol{c}_m = (0, 0)$ and $\boldsymbol{c}_m = (0, -\pi)$. The gaps where there are no oscillations at a Cartesian position are covered by other raster nodes that have oscillations at this position. For $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$, the raster at $\boldsymbol{c}_m = (-\pi, -\pi)$ is particularly fast in comparison to the other centre configurations and can fill the gaps of $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ when one searches for the fastest oscillation per Cartesian position A and its mirror image B about the x-axis.

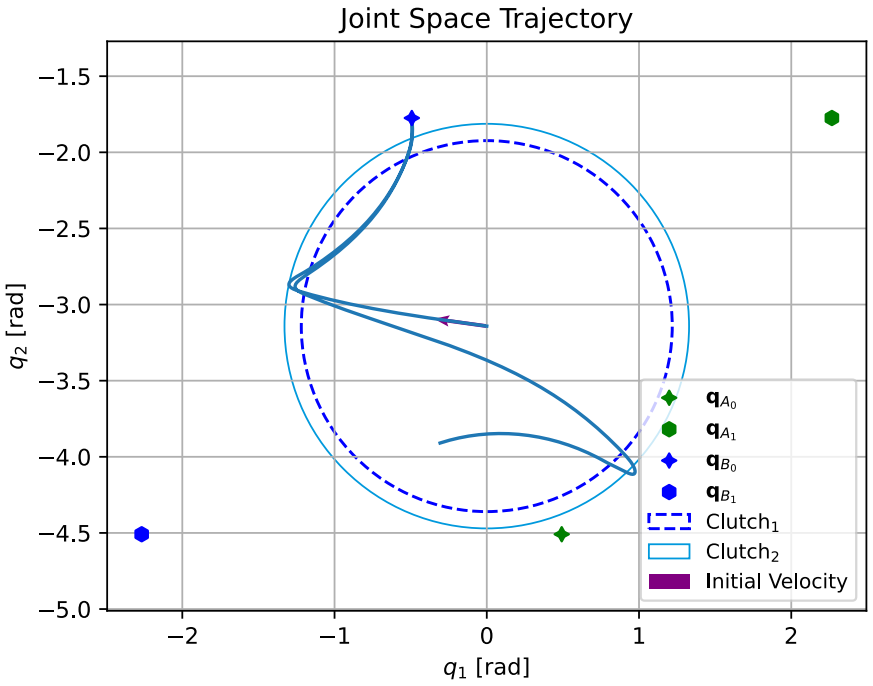
6.1.3. Accuracy

In the workspace coverage of Figure 6.4, the raster nodes marked with an orange cross are raster nodes for which the raster continuation found no oscillation even though the neighbouring raster node contains oscillations. As a consequence, the raster nodes besides those orange crosses have never received an initial guess and thus have no members. Those raster nodes are displayed as white pixels. The grey regions mark raster nodes with no successful member. At least one of their members stops in B but not in A. For most of those members, B is reached very accurately within $1\,\mu$m. This is one of the reasons why we extended the success criteria of the stiffness optimization and the clutch switching times optimization in Sections 5.3.2 and 5.4.2 so that only results that also stop in A count as successful. In Figure 6.4 and Figure 6.7, there are coloured pixels, each displaying a found oscillation, that are only surrounded by grey, white regions or orange crosses. These show that initial guesses from results that only stop in B can lead to oscillations that stop in B and A in another position. This is another reason why the raster continuation also considered raster node members that stop in B as initial guesses.

Figure 6.5 shows an example of an optimization result for $\boldsymbol{p}_A = (0.12, 0.62)^\top$ m that only stops in B and uses the switching circles to bounce in the direction toward B. When analysing Figure 6.5a, the energy jumps come from disengaging a clutch while its spring was still deflected. The first energy jump was caused as the trajectory entered the switching circle of clutch 1 and left it at a position different from $q_{\text{entry},1}$ (Section 5.1.2) as illustrated in Figure 6.5b. After



(a) Total Energy

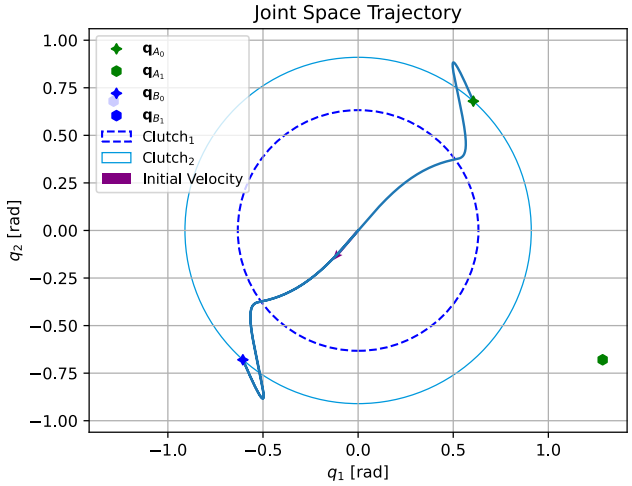


(b) Joint Space

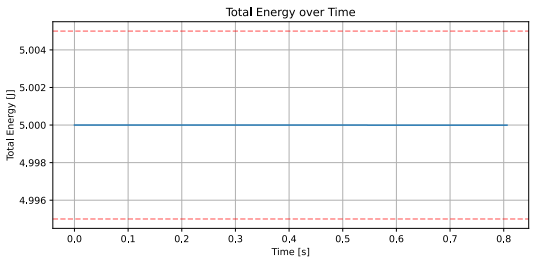
Figure 6.5.: Bouncing trajectory that stops in B but not in A.

this energy loss, the trajectory still stopped in B. Directly afterwards, the trajectory retraces its path but deviates more and more from the point-symmetric trajectory over time, such that it does not stop in A at $t = 3T_B$. This result is therefore filtered out.

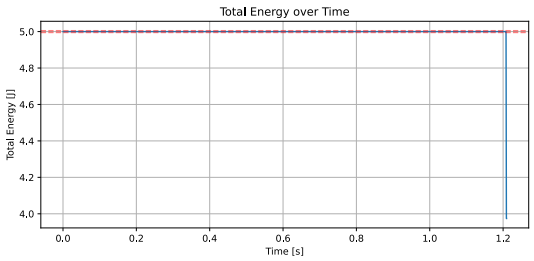
In general, there are also bouncing oscillations that stop in A within 1 mm. Their energy jumps at clutch disengaging events were so small that the trajectory still stayed approximately on a point-symmetric oscillation. However, those oscillations reached A less accurately than B and no longer return to $\boldsymbol{c}_m$ within 10^{-4} rad after one period T. We excluded such pick-and-place motions by filtering them in the clutch switching times optimization (Section 5.4.2), as the trajectory needed to return to $\boldsymbol{c}_m$ for the impulsive transition between pick-and-place motions. Another reason for the filter in $\boldsymbol{c}_m$ are sensitive point-symmetric oscillations.



(a) Joint-Space Trajectory up to T_A



(b) Total Energy up to $2T_B$



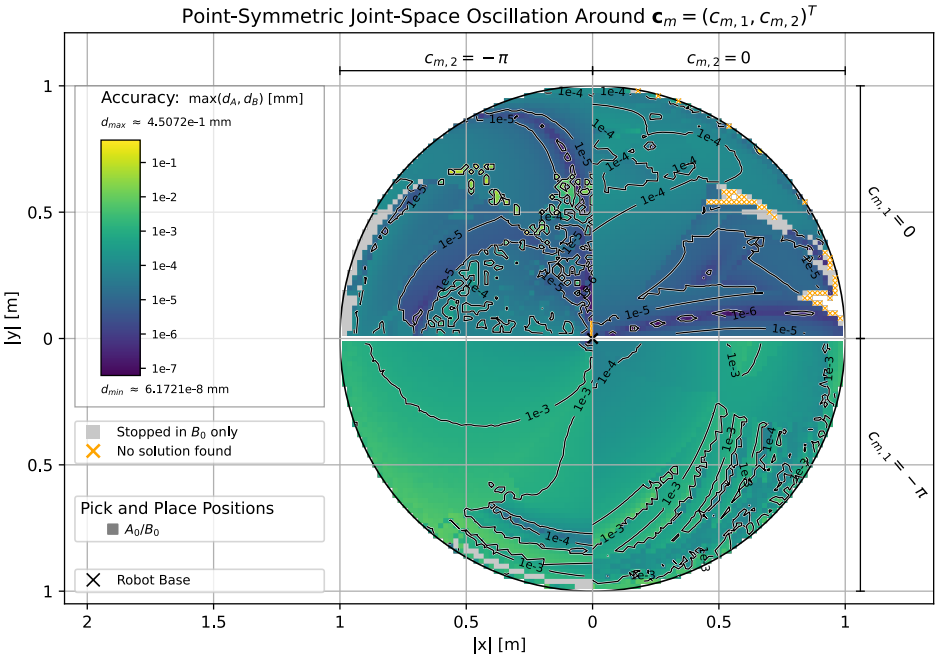
(c) Total Energy up to T_A

Figure 6.6.: Sensitive point-symmetric oscillation that stops in B and A but no longer returns to $\boldsymbol{c}_m$ due to a slight deviation from the resting position A.

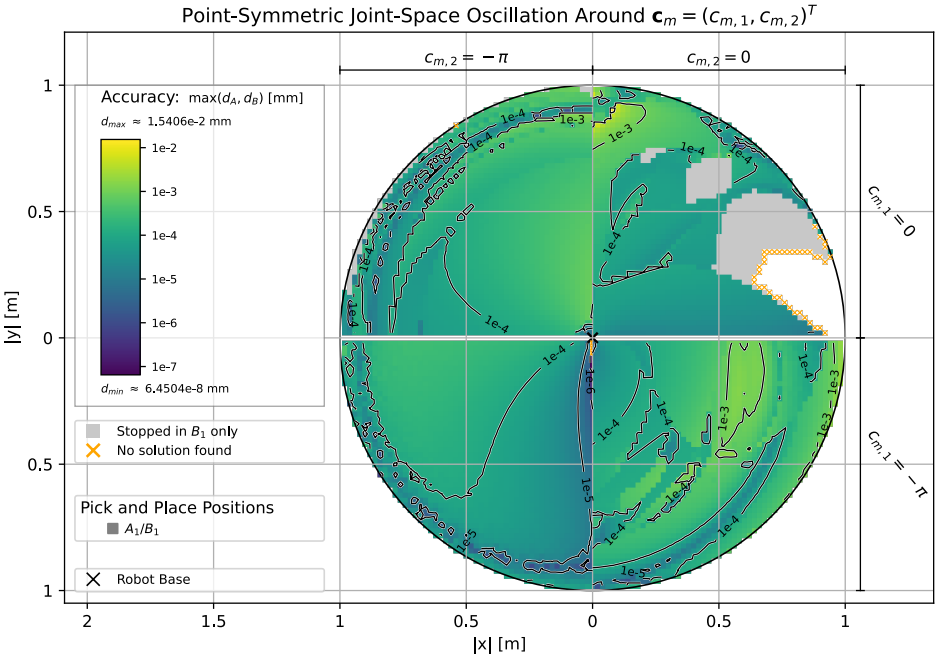
Figure 6.6a shows such a sensitive point-symmetric oscillation for $\boldsymbol{p}_A \approx (0.55183, 0.76446)^\top$ m. At T_B, this oscillation misses B by $d_B = \|\boldsymbol{p}_B - f_{kin}(\boldsymbol{q}_B(\boldsymbol{z}_{switch}))\| = 1.89 \cdot 10^{-4}$ m with a kinetic energy of $1.59 \cdot 10^{-7}$ J. Half a period later at $T_A = 3T_B$, the distance $d_A = \|\boldsymbol{p}_A - f_{kin}(\boldsymbol{q}_A(3T_B, \boldsymbol{z}_{switch} \backslash \{T_B\}))\| = 2.84 \cdot 10^{-4}$ m to A is only slightly larger, whereas the kinetic energy of $1.11 \cdot 10^{-3}$ J is almost four orders of magnitude higher than in B. Figure 6.6b shows no visible energy loss from $\boldsymbol{c}_m$ to B and back to $\boldsymbol{c}_m$. There the constant energy level shows that the clutch does not disengage at the rest position at time T_B. In contrast to that, Figure 6.6c shows an energy loss directly at A. The trajectory thus stops in B and A within 1 mm, but due to numerical noise or a minimal energy loss along the trajectory, the trajectory arrives at A at a slightly different position than in B. There, the clutch briefly disengages, so that the trajectory leaves the potential field at a completely different position than where it entered and loses a considerable amount of energy due to releasing the deflected spring at A. The trajectory then deviates strongly and does not retrace its trajectory. It thus no longer returns to $\boldsymbol{c}_m$ after one period T and is filtered out with our tolerance of 10^{-4} rad in $\boldsymbol{c}_m$. In such a case, one could consider not disengaging the clutch at $T_A = 3T_B$ or restricting the upper bound r_{max} of the radii in the optimization so that B and A no longer lie in the immediate vicinity of the switching circle.

Figure 6.7 shows in millimetres how accurately the fastest oscillations per raster node from Section 6.1.2 stop at A and B. The figures show that the found oscillations are very accurate in simulation. Only individual oscillations lie above 0.1 mm while most oscillations lie in the micrometer range of 10^{-3} mm or below. Similar to the period times in Section 6.1.2, the accuracy mostly changes smoothly between different raster nodes for $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ with $\boldsymbol{c}_m = (-\pi, -\pi)$.

This section and the previous one suggest that there is no trade-off between fastest oscillation per raster node and accurate pick-and-place motions in our found oscillations. When looking at all found oscillations across all rasters, around 89.7 % reach A and B within 1 μm.



(a) A_0/B_0



(b) A_1/B_1

Figure 6.7.: Accuracy of the fastest oscillation per raster node.

6.2. Closed Conservative 3×3 Pick-and-Place Sequence

We evaluate the sequence cost and duration of the found closed 3×3 pick-and-place sequences fulfiling the 3×3 task set from Section 4.4.2. The implemented GTSP optimization searches for sequences with minimum sequence cost for pick-and-place motions starting in $\boldsymbol{c}_m$ and determines the order in which the motions are performed. As introduced in Section 4.4.2, the sequence cost is the sum of $\|\Delta\boldsymbol{\rho}\|_1$ of the momentum differences at $\boldsymbol{c}_m$ between consecutive motions. Each valid placement of the 3×3 task set is identified by $\boldsymbol{p}_{grid}$, where all place positions A lie in the quadrant of $x \geqslant 0$ and $y > 0$. We first look at some of the found sequences in Section 6.2.1 and then evaluate placements across the whole workspace in Section 6.2.2.

6.2.1. Individual 3×3 Pick-and-Place Sequences

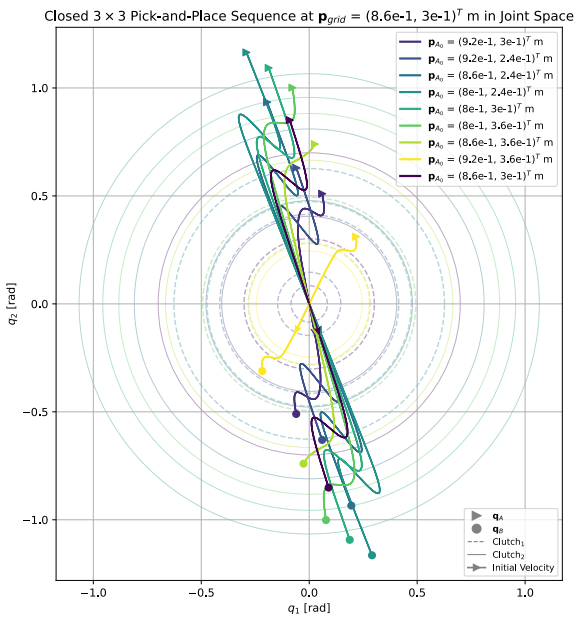
We first consider individual sequences and focus on characteristic examples with varying initial velocity angles for the impulsive transition from one oscillation to the next at $\boldsymbol{c}_m$, shown in Figure 6.8.

Near singularities, the joint configurations of the 3×3 task set's pick and place positions lie further apart. In Figure 6.8a, the GTSP only found oscillations of which most have a similar momentum, except for the oscillation of the GTSP node closest to the singularity for $\boldsymbol{p}_{A_0} = (0.92, 0.36)^\top$ m. Although this oscillation moves almost directly between its two configurations, its initial velocity angle differs from the others. Since the momentum difference of this oscillation to other oscillations is higher, this oscillation is responsible for the higher sequence cost of 11.91 N m s. The example shown in Figure 6.8a is also the most expensive axis-aligned placement of the task set where the pick-and-place motions of the sequence fulfiling the task set start in $\boldsymbol{c}_m = (0, 0)$.

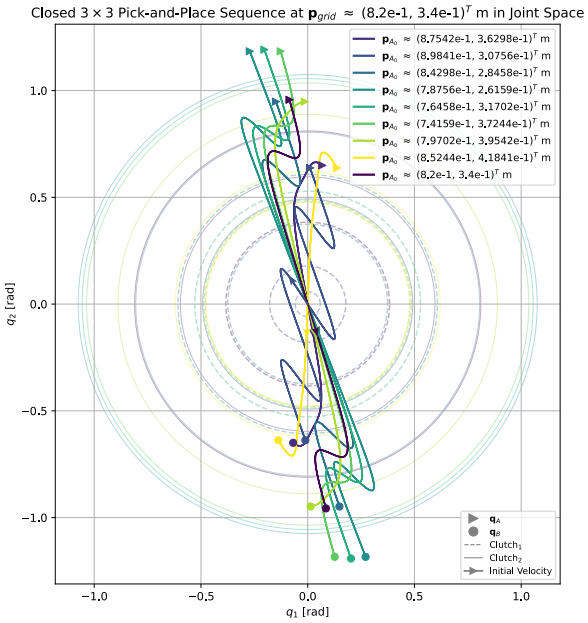
The radially aligned 3×3 task set in Figure 6.8b also lies near a singularity but not as close as Figure 6.8a. There, its configurations are closer together than in Figure 6.8a. In this example, there is a single outlier at $\boldsymbol{p}_{A_0} \approx (0.84298, 0.28458)^\top$ m where the initial velocity angle points almost in the opposite direction of the other angles. This outlier is responsible for the higher sequence cost of 10.703 N m s. The sequence in Figure 6.8b is thus the most expensive radially aligned placement in which the pick-and-place motions of the sequence fulfiling the task set start in $\boldsymbol{c}_m = (0, 0)$.

Apart from these two cases near a singularity, a sequence with all oscillations from a single raster can be unfavourable. Either a raster node of this raster has no oscillation or the initial velocity angles of its oscillations make the sequence more expensive than a combination of both rasters with the same $\boldsymbol{c}_m$ and different configuration index. The axis-aligned 3×3 task set in Figure 6.8c shows such a combination of different rasters for $\boldsymbol{c}_m = (0, -\pi)$. Here, there are three out of the nine oscillations from the raster with configuration index 1. Even though the initial velocity angles of one configuration index differ considerably from those of the other configuration index, the combination results in a sequence cheaper than one where all oscillations come from a single raster. This example is also the axis-aligned task set placement with the highest sequence cost of 7.918 N m s for motions starting in $\boldsymbol{c}_m = (0, -\pi)$.

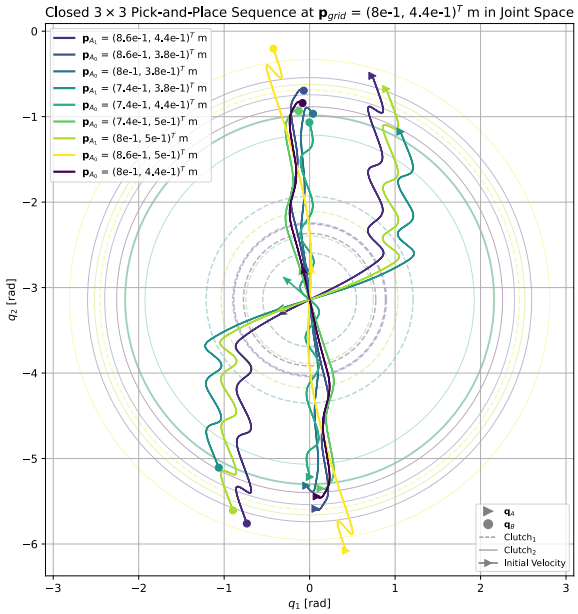
The raster continuation intends to find similar shaped oscillations between neighbouring raster nodes as the initial guesses for one raster node come from the members of the neighboring raster nodes. Figure 6.8d shows that similarly shaped oscillations do not necessarily have similar initial velocity angles and thus small transition costs. Some of them have almost the same initial velocity angle, while the angles of others vary. This example is the radially aligned



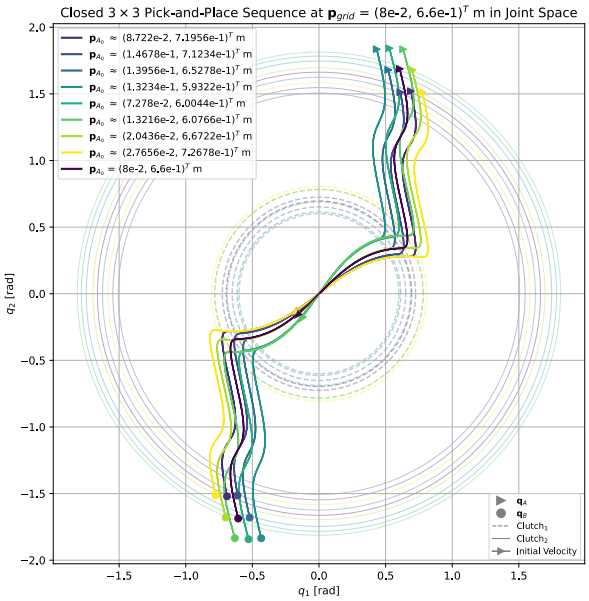
(a) Sequence fulfiling the axis-aligned 3 × 3 task set with $\boldsymbol{c}_m = (0, 0)$ with a sequence cost of around 11.91 N m s



(b) Sequence fulfiling the radially aligned 3 × 3 task set with $\boldsymbol{c}_m = (0, 0)$ with a sequence cost of around 10.703 N m s



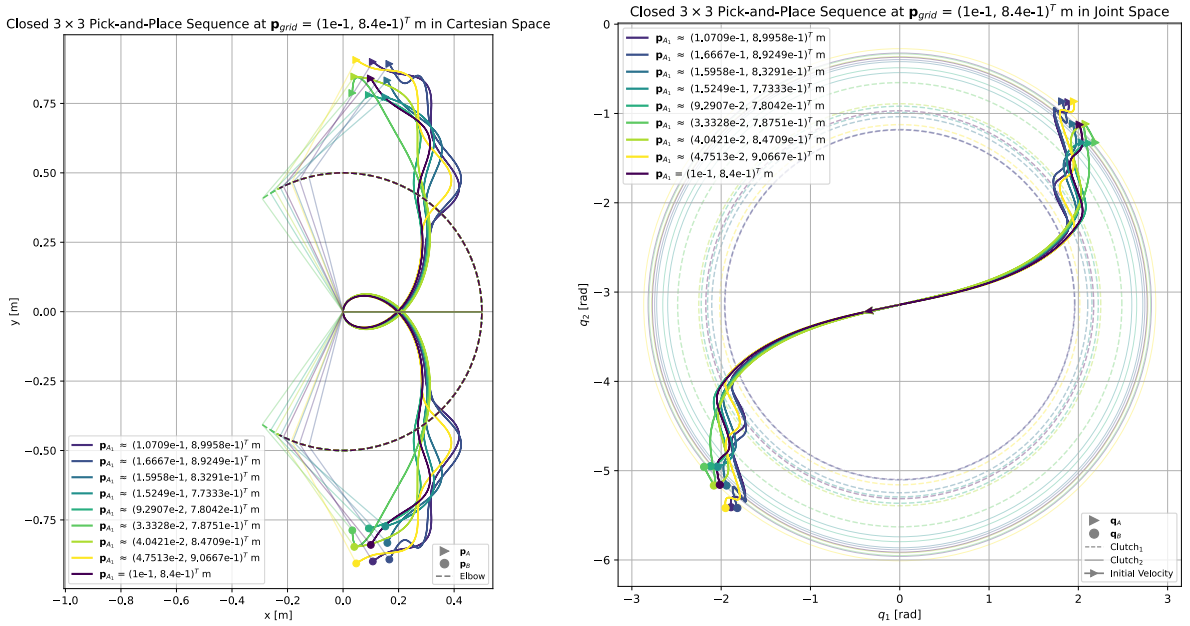
(c) Sequence fulfiling the axis-aligned 3 × 3 task set with $\boldsymbol{c}_m = (0, -\pi)$ with a sequence cost of around 7.918 N m s



(d) Sequence fulfiling the radially aligned 3 × 3 task set with $\boldsymbol{c}_m = (0, 0)$ with a sequence cost of around 0.163 N m s

Figure 6.8.: Joint-space plots of 3 × 3 pick-and-place sequences with varying initial velocity angles.

placement of the task set with the lowest sequence cost of 0.163 N m s for $\boldsymbol{c}_m = (0, 0)$.



(a) Cartesian Space

(b) Joint Space

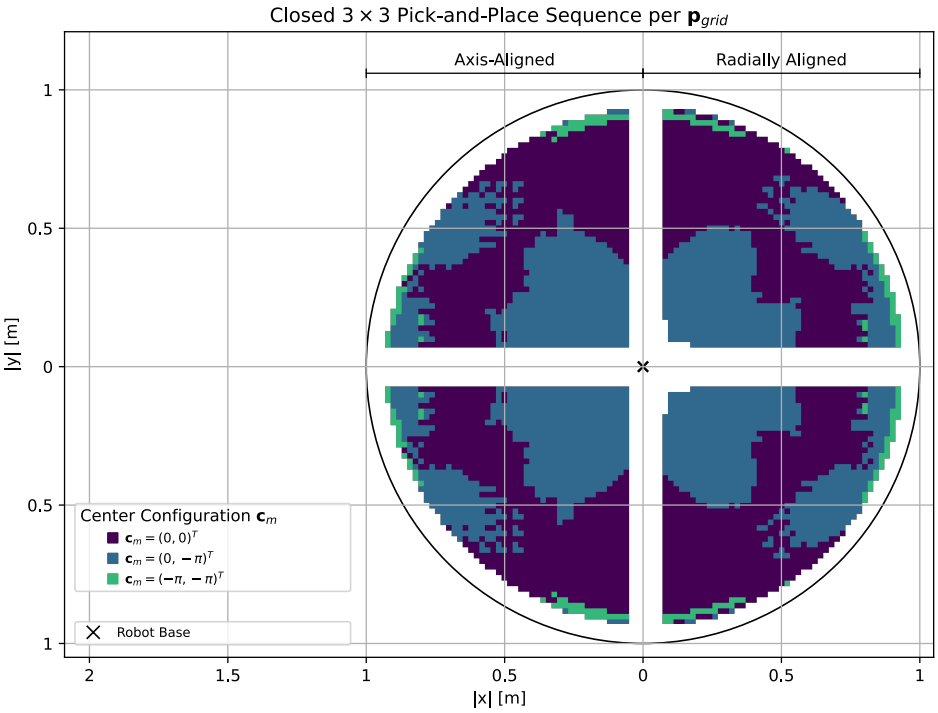
Figure 6.9.: Pick-and-place sequence with a sequence cost of around 0.086 N m s fulfiling the radially aligned 3 × 3 task set with $\boldsymbol{c}_m = (0, -\pi)$.

There are also closed conservative 3 × 3 pick-and-place sequences where the initial velocity angle almost stays the same for all oscillations selected for the sequence. Such an example is displayed in joint space by Figure 6.9b. One can see that the oscillations overlap for a time in joint space before they deviate from each other. For this sequence, different clutch switching times shape the oscillations such that they stop in the pick and place positions of the task set. Figure 6.9a shows this sequence in Cartesian space, where the different oscillations also overlap for a time. Toward the resting positions A and B, however, their shapes differ from one another. Since the initial velocity angles are almost the same, the transition costs are quite low. Sequences like this one suggest that for some placements the clutches could be completely responsible for transitioning between different oscillations and that the motor does not have to deliver the short torque pulse. For the example in Figure 6.9b, this is the radially aligned placement for the task set with the lowest sequence cost of 0.086 N m s for motions starting in $\boldsymbol{c}_m = (0, -\pi)$. The springs' maximum deflection in joint 1 is around 0.097 rad and for joint 2 around 0.151 rad.

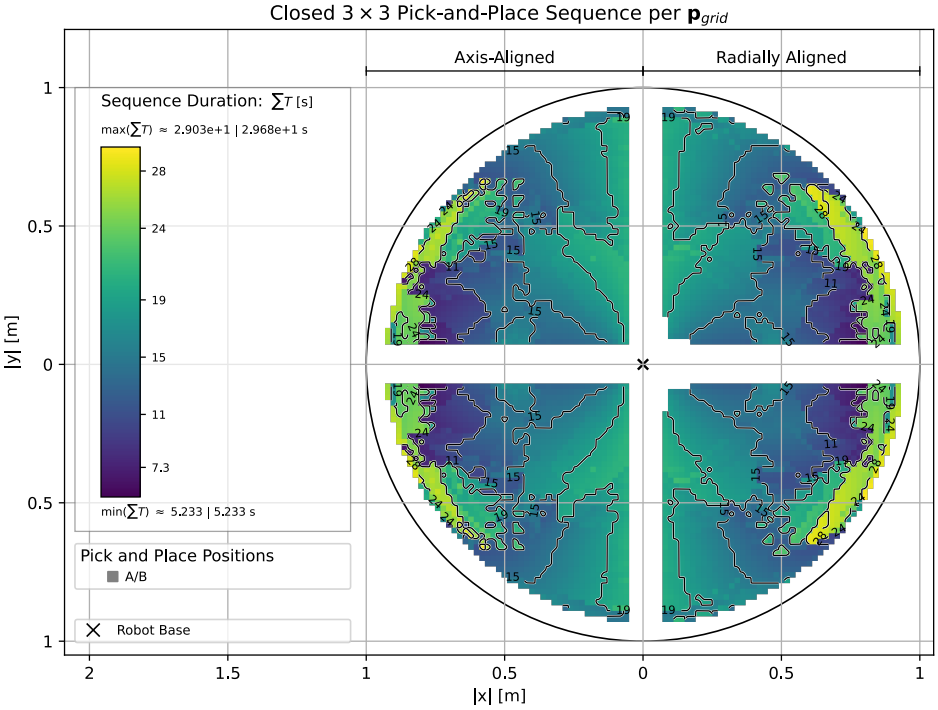
6.2.2. Coverage

Beyond these individual pick-and-place sequences, we examine the found sequences across the whole workspace to answer our research question and sub-question. The GTSP finds sequences with the lowest sequence cost for a placement at $\boldsymbol{p}_{grid}$, whose pick-and-place motions all start in the same $\boldsymbol{c}_m$. From the found sequences of the GTSP for a 3 × 3 task set placement at $\boldsymbol{p}_{grid}$, we select the sequence that has the shortest sequence duration.

Figure 6.10 displays the information on the selected sequence per placement at $\boldsymbol{p}_{grid}$. In each of the figures displayed, the axis-aligned and radially aligned placements look similar. Figure 6.10a displays the centre configuration $\boldsymbol{c}_m$ in which the motions of the selected sequence start for

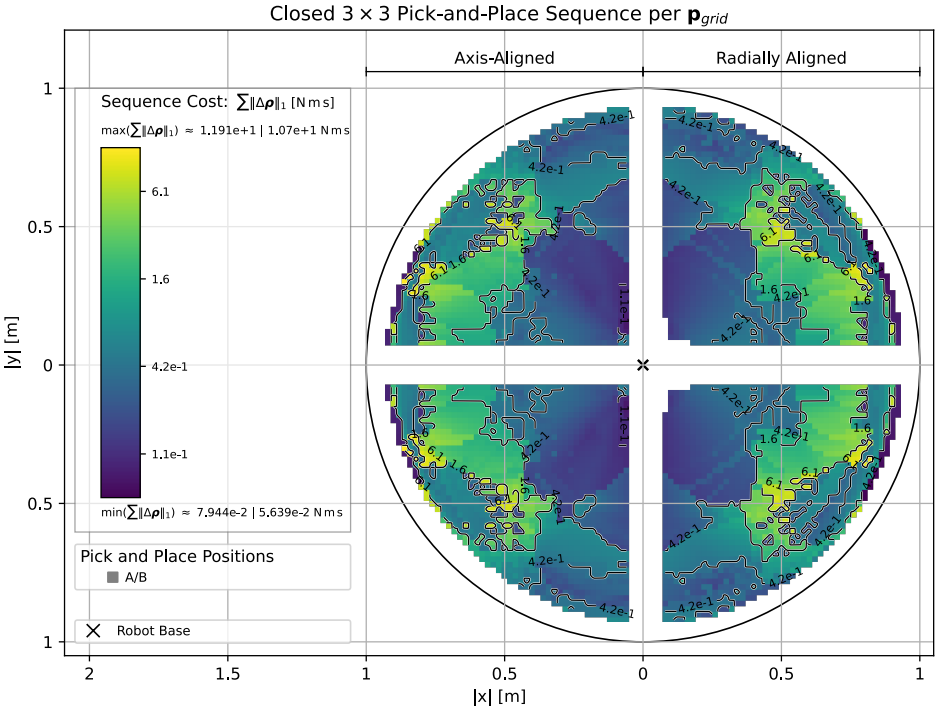


(a) Chosen Centre Configuration $\boldsymbol{c}_m$ per $\boldsymbol{p}_{grid}$



(b) Sequence Duration

Figure 6.10.: Closed 3×3 pick-and-place sequence per 3×3 task set position $\boldsymbol{p}_{grid}$ and placement. The axis-aligned placement is displayed on the left half and the radially aligned placement on the right half in each figure. Both halves show the sampled quadrant with $x \geqslant 0$ and $y > 0$ for position A and share one colour scale.



(c) Sequence Cost

Figure 6.10.: Closed 3×3 pick-and-place sequence per 3×3 task set position $\boldsymbol{p}_{grid}$ and placement. The axis-aligned placement is displayed on the left half and the radially aligned placement on the right half in each figure. Both halves show the sampled quadrant with $x \geqslant 0$ and $y > 0$ for position A and share one colour scale.

each placement at $\boldsymbol{p}_{grid}$. As the selected sequence is the fastest sequence of the placement, the shortest sequence durations across most of the workspace are reached with $c_{m,1} = 0$. Sequences with $\boldsymbol{c}_m = (-\pi, -\pi)$ are the fastest only in a small part of the workspace.

Figure 6.10b displays the sequence duration of the selected sequence. In this figure, one can see that the sequence durations of both placements range from around 5 s up to 30 s. Near the singularity on the outer workspace, there are sequences that have an abruptly higher sequence duration than their neighbouring positions. When looking at Figure 6.10a, the fastest sequence durations are for sequences with $\boldsymbol{c}_m = (0, 0)$. At the fastest positions $\boldsymbol{p}_{grid}$, Figure 6.4 shows that the fastest oscillations per raster node lie within $\boldsymbol{c}_m = (0, 0)$. In the regions of the highest sequence durations, the quadrants of Figure 6.1 at $\boldsymbol{c}_m = (0, 0)$ show that there are raster nodes with few oscillations or none at all. At those positions, the selected sequence has another centre configuration than $\boldsymbol{c}_m = (0, 0)$ as there might have not existed a pick-and-place sequence for $\boldsymbol{c}_m = (0, 0)$. Figure 6.10b also shows that around $x = 0$ the sequence duration is between 19 and 24 s even for placement positions. Near the robot's base, the robot performs the task set with a lower mean velocity than those near the outer workspace boundary around $x = 0$.

When looking at the sequence cost in Figure 6.10c, one can see that on placement positions where the sequence duration was lowest, the corresponding sequence cost is among the highest. Here, the values for the maximum and minimum sequence cost in the workspace vary between the two placements with the radially aligned placement having a minimum sequence cost of 0.05639 N m s and the axis-aligned a minimum of 0.07944 N m s. The lowest sequence costs were found for $\boldsymbol{c}_m = (-\pi, -\pi)$

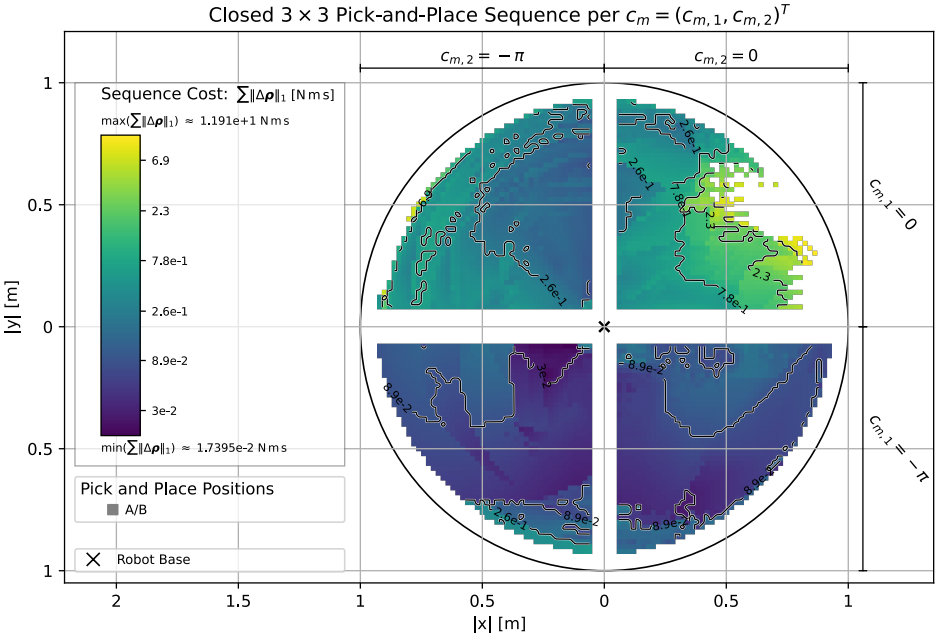
Near the origin, the radially aligned placement covers less of the workspace than the axis-aligned placement. This is not caused by missing sequences but by the valid positions $\boldsymbol{p}_{grid}$, whose number and location differ between the two placements. The radially aligned placement has fewer valid $\boldsymbol{p}_{grid}$ near the origin but more at the outer workspace boundary. In total, there are 1405 valid $\boldsymbol{p}_{grid}$ for the radially aligned placement and 1398 for the axis-aligned placement. The reason for the different amounts of valid $\boldsymbol{p}_{grid}$ per placement mode is that the 3×3 grid of the place positions A might have a different orientation for the same $\boldsymbol{p}_{grid}$ in the radially and axis-aligned placement. For the same $\boldsymbol{p}_{grid}$, all positions A of the axis-aligned placement might lie within $x \geqslant 0$ and $y > 0$ but not for the radially aligned placement, making the 3×3 task set position $\boldsymbol{p}_{grid}$ invalid for the radially aligned placement.

Centre Configuration $\boldsymbol{c}_m$	Placement	Workspace Coverage	Sequence Cost [N m s]				Sequence Duration $\sum T$ [s]			
		[%]	Mean	Std	Min	Max	Mean	Std	Min	Max
$(0, 0)$	Axis-Aligned	83.83	1.39	1.67	0.16	11.91	16.53	4.18	5.23	28.84
	Radially Aligned	81.78	1.47	1.68	0.16	10.70	16.40	4.21	5.23	31.05
$(0, -\pi)$	Axis-Aligned	98.86	0.37	0.55	0.08	7.92	20.09	4.05	12.64	31.33
	Radially Aligned	98.43	0.41	0.54	0.09	7.47	20.30	4.30	12.51	32.65
$(-\pi, 0)$	Axis-Aligned	100.00	0.09	0.04	0.03	0.28	40.73	5.86	22.14	65.85
	Radially Aligned	100.00	0.10	0.06	0.02	0.43	41.29	6.24	22.42	65.13
$(-\pi, -\pi)$	Axis-Aligned	100.00	0.09	0.08	0.02	0.57	27.12	3.55	14.92	36.76
	Radially Aligned	100.00	0.09	0.08	0.02	0.58	27.24	3.56	15.07	36.71

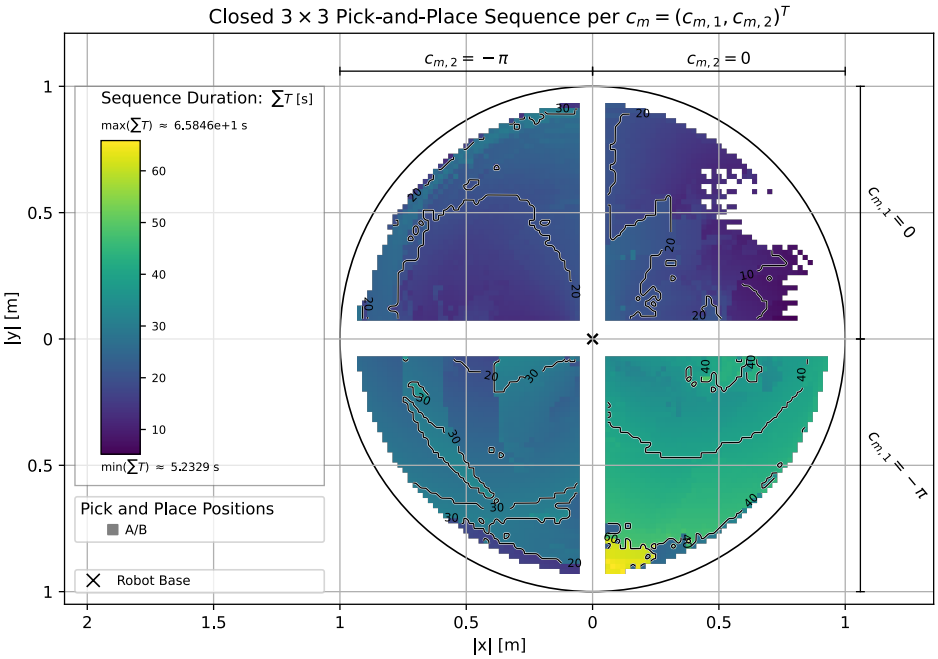
Table 6.2.: Coverage, sequence cost and sequence duration of the lowest sequence cost per 3×3 task set placement at $\boldsymbol{p}_{grid}$ for each centre configuration $\boldsymbol{c}_m$.

To analyse the sequence cost and sequence duration per $\boldsymbol{c}_m$, we select the sequence with the lowest sequence duration per placement and $\boldsymbol{c}_m$ among the GTSP results with the lowest sequence cost. Table 6.2 displays the workspace coverage of the chosen sequence per placement position and centre configuration $\boldsymbol{c}_m$. The 3×3 task set can be fulfiled with conservative sequences across the whole workspace for both placements at $\boldsymbol{c}_m = (-\pi, 0)$ and $\boldsymbol{c}_m = (-\pi, -\pi)$. At $\boldsymbol{c}_m = (0, 0)$ and $\boldsymbol{c}_m = (0, -\pi)$, the axis-aligned placement covers a slightly larger share of the valid $\boldsymbol{p}_{grid}$ positions. The coverage of $c_{m,1} = 0$ stays below 100 %, since fewer oscillations per raster node exist for $c_{m,1} = 0$, as the numbers in Figure 6.1 show. Table 6.2 additionally shows the mean, standard deviation, minimum and maximum values of the sequence cost and durations of the chosen sequence per placement position and centre configuration $\boldsymbol{c}_m$. As in the workspace coverage, the displayed values in Table 6.2 for the sequence cost and duration are similar between the two placements. For most of these values, the axis-aligned placement has slightly lower sequence cost and durations. The off-raster nodes in the off-raster continuation did not receive all successful members of the neighbouring raster nodes as initial guesses. An off-raster node could thus have slightly fewer oscillations than one of the neighbouring raster nodes such that the GTSP has fewer oscillations to choose from for finding a suitable sequence. This could make the values displayed in Table 6.2 slightly worse for the radially aligned placement compared to the axis-aligned one. Another reason for the slightly worse values might be that the radially aligned placement has more valid placements on the outer boundary, where fewer oscillation per raster nodes are (Section 6.1.1).

Nevertheless, the values of both placements are similar, suggesting that the plot of the chosen sequence per $\boldsymbol{p}_{grid}$ and centre configuration $\boldsymbol{c}_m$ is also similar. As the axis-aligned placement has slightly better values, we evaluate Figure 6.11 for this placement example. The similar



(a) Sequence Cost



(b) Sequence Duration

Figure 6.11.: Lowest sequence cost per centre configuration for the 3 × 3 axis-aligned task set. Each quadrant in the figures belongs to one centre configuration $\boldsymbol{c}_m$.

figure for the radially aligned placement is given in Appendix A.2.

In Figure 6.11a, most sequence costs in the two quadrants with $c_{m,1} = -\pi$ lie below 0.089 N m s. In contrast, neither quadrant with $c_{m,1} = 0$ has values below 0.089 N m s. Of these two quadrants, $\boldsymbol{c}_m = (0, -\pi)$ has more placements with low sequence costs, whereas $\boldsymbol{c}_m = (0, 0)$ has a larger area with sequence costs above 0.78 N m s. According to Table 6.1, the merged rasters of both configuration pairs provide many oscillations per GTSP node for $c_{m,1} = -\pi$, even though the raster nodes with $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ have only up to 4 oscillations at $\boldsymbol{c}_m = (-\pi, -\pi)$. With more oscillations per raster node, the GTSP can have more choices with varying initial velocity angles to reduce the sequence cost. Since the rasters at $c_{m,1} = -\pi$ have more oscillations and the sequence cost at the $c_{m,1} = -\pi$ quadrants is lowest, the sequence costs can presumably be improved for $c_{m,1} = 0$ by finding more oscillations for the GTSP to choose from.

Figure 6.11b shows the sequence durations per placement position. In the quadrant of $\boldsymbol{c}_m = (0, 0)$, the sequence durations are the lowest as the quadrant has regions of under 10 s. It is also noticeable, that in the region of the highest sequence durations in Figure 6.10b, the quadrant of $\boldsymbol{c}_m = (0, 0)$ does not have sequences to choose from. Following $\boldsymbol{c}_m = (0, 0)$, the quadrant at $\boldsymbol{c}_m = (0, -\pi)$ has a larger region of sequences of under 20 s than $c_{m,1} = -\pi$. When comparing the quadrants of $c_{m,1} = -\pi$, $\boldsymbol{c}_m = (-\pi, -\pi)$ is the quadrant which has more sequences under 30 s. Within $\boldsymbol{q}_{A_1}/\boldsymbol{q}_{B_1}$ in Figure 6.4, the largest region with oscillations faster than 0.3 s is in the quadrant of $\boldsymbol{c}_m = (-\pi, -\pi)$. This might explain why $\boldsymbol{c}_m = (-\pi, -\pi)$ has better sequence duration values across the workspace than $\boldsymbol{c}_m = (-\pi, 0)$ and was chosen for the small workspace part in Figure 6.10.

As the sequence cost is lowest in the quadrants of $c_{m,1} = -\pi$ and the sequence duration lowest in the quadrants of $c_{m,1} = 0$, this could suggest a trade-off. However, one has to consider that for $\boldsymbol{q}_{A_0}/\boldsymbol{q}_{B_0}$ and $c_{m,1} = -\pi$ in Figure 6.2 there are two directions in which oscillations can be found with the optimization pipeline. All other rasters have only one direction for the oscillations and thus a presumably lower oscillation count per raster node. Together with the assumption that more oscillations per raster node can lead to lower sequence cost, no statement can be made regarding a possible trade-off.

Overall, the radially aligned and the axis-aligned placement hardly differ in coverage, sequence cost and sequence duration. The 3 × 3 task set can be placed across the whole workspace with both placement modes.

7 Discussion

Since the closed 3×3 pick-and-place sequences in Chapter 6 are conservative in our conservative system, they are quasi-conservative in a real system (Definitions 2 and 4). Our answer to the research question from Section 1.1 is thus that we can execute the considered closed 3×3 quasi-conservative pick-and-place sequences across the whole workspace for the axis-aligned and radially aligned placement (Section 4.4.2). For this 3×3 task set, the positions of the pick and place positions were restricted to ones where the pick position is the place position mirrored about the x-axis. Within the optimized pick-and-place sequences, the sequence durations range from 5 to 30 s across the whole workspace. The shortest sequence duration of 5.23 s is reached by both placements at $\boldsymbol{p}_{grid} = (0.8, 0.08)\,\mathrm{m}$ with $\boldsymbol{c}_m = (0, 0)$ (Section 6.2.2). This value is only an upper bound for what our approach can achieve. The GTSP minimizes the sequence cost for each centre configuration $\boldsymbol{c}_m$ and the sequence duration only decides among the sequences with the lowest sequence cost and then among the centre configurations at $\boldsymbol{p}_{grid}$. As we also did not search for all possible oscillations, sequences with an even shorter duration or lower sequence cost might exist. Even without finding all possible oscillations, this work's results already show great potential of using clutched parallel elastic actuators for quasi-conservative pick-and-place sequences across the workspace.

To the best of our knowledge, Sachtler and Albu-Schäffer [81] are the only authors who also exploit the resonance of a multi-DoF robot and consider transitions between different strict normal modes, which are a special kind of free oscillation. Their mode selection controller changes between modes during operation with small torques and is inactive close to the equilibrium. It thus shifts the trajectory onto the other mode during the oscillation. We instead aim for an impulsive transition directly at the equilibrium.

7.1. Search Space

Already with a subset of all possible oscillations, task set placements and quasi-conservative pick-and-place sequences, we can find closed quasi-conservative 3×3 pick-and-place sequences across the whole workspace. Since this is only a subset of the possible search space, this raises some questions.

The orientation of the 3×3 task set is coupled to the centre position of the 3×3 grid of place positions for both placement modes. Since the sequence costs and durations at the same position of both placements are similar, the question arises whether this also holds for an independently chosen orientation.

Beyond the investigated 3×3 sequences, the multiple oscillations per raster node in Section 6.1.1 suggest a degree of freedom to find a suitable combination of oscillations for other pick-and-

place sequences. This raises the question of how energy-efficient the transitions remain in the extreme case of a task set that spreads across the whole workspace. For such larger pick-and-place sequences, a heuristic GTSP solver would presumably be a better choice than the exact dynamic programming. For the 3×3 sequence, Section 6.2.2 suggests that more oscillations per GTSP node lower the transition costs. More oscillations at GTSP nodes might thus find quasi-conservative pick-and-place sequences with even lower sequence cost. If several sequences again share the lowest sequence cost, more members per GTSP node might also improve the sequence durations. The tolerances of 1 mm at A and 10^{-4} rad at $\boldsymbol{c}_m$ in Section 5.4.2 reduced the number of accepted trajectories. A brief look at one task set placement, not shown in the evaluation, indicates that the transition cost of the impulsive transition between some of the trajectories excluded due to the tolerance at $\boldsymbol{c}_m$ is almost zero. The deviation of the excluded trajectories stopping in B and A grows along the trajectory until they return to $\boldsymbol{c}_m$ within 10^{-2} rad. Since part of these deviations might stem from numerical inaccuracies and small energy losses during clutch disengagement, a slight relaxation of the tolerances at $\boldsymbol{c}_m$ could admit more pick-and-place motions with very small transition cost between them. One could correct the slight deviations with a motor. When considering motor losses, this raises the question whether it is more energy-efficient to have free oscillations with higher transition cost or to correct slight deviations to produce oscillations and have almost no transition cost.

Besides relaxing the tolerances, the search itself leaves room for finding more oscillations. Continuing the results of the stiffness optimization in Section 5.3 over neighbouring pick and place positions for each centre configuration and configuration pair could provide further support raster nodes and members for the raster continuation in the clutch switching times optimization. A smaller joint-space distance between neighbouring raster nodes near singularities, where neighbouring raster nodes lie further apart in joint space (Section 6.1.2), more oscillations near singularities could be found with the clutch switching times optimization.

The cost functions in the optimization stages all aim for the specified configuration values and do not treat joint angles differing by multiples of 2π, such as π and $-\pi$, as equal. This way, oscillations that may stop in another representation than the targeted one were excluded from the search space to keep the joint-space distance between $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$ small and favour single-turn oscillations where possible. The raster continuation in the clutch switching times optimization sometimes also aims for $|\boldsymbol{q}_A - \boldsymbol{q}_B| \geqslant 2\pi$ when the configuration values of these $\boldsymbol{q}_A$ and $\boldsymbol{q}_B$ have a smaller joint-space distance to the ones from the initial guess. The optimization for these values gave way to multi-turn oscillations. As Section 6.1.1 showed, there are multi-turn oscillations that swing directly between the configurations while there are single-turn oscillations that oscillate strongly along their path. This raises the question of how far multi-turn oscillations are faster than single-turn oscillations and how far it would improve the sequence duration and transition cost when the cost functions treat such angles as equal and the optimization pipeline finds more oscillations. As Section 6.1.1 additionally suggests a relation between the multiple oscillations of one raster node, analysing this relation could help in finding desired oscillations in a targeted way.

Besides the search, the system itself offers further degrees of freedom. Together with realistic link lengths and masses, the fixed stiffness values could be optimized to find as many oscillations per raster node as possible. As long as all pick-and-place motions considered for a sequence share the same energy for impulsive transitions, another total energy value might yield oscillations at raster nodes where none were found with $E_{total} = 5\,\mathrm{J}$. Varying the energy slightly between tasks would be a further option, but raises the question of how energy-efficient the impulsive transitions remain since they are then no longer conservative.

Pick-and-place applications demand short period times and precise positioning[67, 6]. The

Epson GX20B with the same reach of 1 m as our robot needs 0.42 s to move a 2 kg payload back and forth over 300 mm[28]. This corresponds to a mean speed of 1.43 m/s. Our sequences with pick and place positions, which are on average 0.3 m apart, reach a mean speed of 0.39 m/s on average and at most 0.81 m/s. They are thus considerably slower, although our robot carries no payload. So far, the period entered the analysis only as a result criterion on the sequences and not as an objective of the optimizations. The degrees of freedom of the system from the previous paragraph also offer ways to shorter sequence durations. Since the period time depends on the system's energy, another total energy value might improve the period to a desired value[82]. At the same energy, we observed that larger switching circles come with higher stiffness values in the radius continuation (Section 5.4.3). Higher stiffness values could thus let the clutches engage closer to the pick and place positions, so that the kinetic energy equals E_{total} for a longer part of the trajectory and the robot could move faster.

The found oscillations could also be passed through faster. Since the total energy equals the potential energy of the deflected springs in the resting positions A and B, multiplying the stiffness values and the total energy by the same factor does not modify the path in joint space and decreases the period times with the square root of this factor[56, §10, Problem 2]. If all masses are reduced by the same factor at unchanged stiffness values and potential energy, the total energy stays the same since $E_{pot}(\boldsymbol{q}_B) = E_{total}$. The period times then decrease with the square root of this factor, while the robot oscillates along the same path[56, §10, Problem 1]. Any other change of the total energy, the stiffness values or the masses, however, might make the found oscillations vanish, so that they have to be searched again with the optimization pipeline.

Raising the kinetic energy at $\boldsymbol{c}_m$, however, comes at a price. On the same path and thus with the same initial velocity angles, the momentum difference at $\boldsymbol{c}_m$ grows with $\sqrt{E_{total}}$ according to (4.24). This raises the sequence cost. In contrast, lower link and gripper masses could reduce the momentum difference at the same total energy and thus also reduce the sequence cost. Since the found sequence durations are only upper bounds, it remains to be analysed how far they can be improved by including the period time as an objective of the least squares optimizations and by choosing E_{total}, the stiffness values and the link and gripper masses differently. Including the period time in the GTSP as an objective raises the question of which sequence costs one is willing to accept for shorter sequence durations. A renewed comparison of the improved sequence durations with conventional pick-and-place trajectories on the same robot with the same payload across the whole workspace would then show how close quasi-conservative sequence durations can come to conventional ones.

7.2. Pick-and-Place Task Scope

We are searching for pick-and-place motions that are point-symmetric oscillations through a shared centre configuration between Cartesian points mirrored about the x-axis. This restriction lets the configuration at B determine the one at A, so that the optimization only needs to target B (Section 4.3.2). Oscillations between arbitrary Cartesian points A and B require optimizing towards A as well. For impulsive transitions, the oscillations of different task positions then only need to pass through the same equilibrium placed anywhere within both clutch circles, so that the transition approach discussed in this work itself carries over. This generalization would allow for arbitrary quasi-conservative pick-and-place sequences.

So far, the tasks also move no payload. Since picking up an object changes the mass of the system[48], it remains to be analysed how the clutch times can be adapted to this change.

Gripping and releasing an object also takes time, which the pick-and-place motions do not yet include as dwell times at the pick and place positions. At these positions, the robot comes to rest and undergoes its peak acceleration[67] of the oscillation, so that holding it there with engaged springs demands a higher motor torque[6]. Kordik, Gattringer, and A. Müller [53] argue that current grippers grip and release objects fast enough when the tool centre point moves within a tolerated region around each target. Applied to our oscillations, it remains to be analysed whether the time span in which the trajectory is within the tolerated region suffices for the gripper. Another approach to handling dwell times would be the output-locked mode that Plooij, Wisse, and Vallery [78] use in their CPEA, in which a deflected spring keeps its deflection. In their review, Plooij, Wolfslag, and Wisse [79] explain that in this mode the motor no longer has to work against spring, gravitational or external forces. Although these clutch systems differ from our model, they note that one extra clutch suffices in principle to add this mode to any clutched elastic actuator. A CPEA that locks the links with deflected springs at the pick and place positions would thus presumably suit the requirement for dwell times best. The relaxation of the spring when disengaging the clutch is necessary, since a spring that is still deflected when it engages again could shift the equilibrium away from its intended position.

7.3. Transfer to a Realistic System

To answer the research question, an idealized model of the clutched parallel elastic actuator sufficed. Through the optimization, however, the simulated values become unrealistically small for a physical system, with some oscillations reaching B within 10^{-15} m and stopping there with a kinetic energy of 10^{-30} J (Section 6.1.3). It would be interesting to examine the accuracy of such oscillations in hardware and whether these oscillations remain within our tolerances in A, B and $\boldsymbol{c}_m$.

In a realistic system, a controller would have to keep the robot on the oscillations. A simple tracking controller, such as a PD controller, would be one option. Controllers based on nonlinear normal mode theory are another option, since our oscillations can be considered as normal modes. Such controllers excite oscillations on eigenmanifolds, stabilize them and sustain them against dissipation by regulating the energy[9, 25, 81]. For exciting the desired oscillation, the review of Plooij, Wolfslag, and Wisse [79] provides another approach. They describe that the motor can deflect a spring while the joint is locked. Applied to our system, the springs could be deflected at a rest position such that their equilibria lie at the previously calculated crossing point $q_{\text{entry},i}$ between the desired trajectory and the circle. Removing the lock could then excite the system, so that the robot starts at A or B and follows the desired oscillation with the help of the controllers.

Since we modelled the clutch reaction times as instantaneous, it remains to be analysed how fast such clutches can disengage and engage in hardware and how far these reaction times cause the system to deviate from the found oscillation. The clutches themselves have to be controlled as well. Both the algorithms of this work and the control of the oscillations would therefore have to be adapted to the respective clutch design, which adds control complexity[72, 16]. This raises the question of how such a control has to be designed to stabilize the oscillations with a realistic clutch model, in which a recoiling spring can additionally destabilize the system when the clutches switch[20].

Besides their concern of a recoiling spring, Chen et al. [20] criticize that clutches can lose energy when they disengage. Our model includes this loss, which vanishes for the optimized oscillations since their springs are fully relaxed at disengagement (Section 5.1.2). However, our model does

not include the energy needed to switch and control the clutches. In the CPEA of Penzlin et al. [76], fixing the springs only takes a short current pulse, after which a bistable magnet holds them without further current. This suggests a low energy consumption for switching the clutches. The conservative model of our system furthermore omits non-conservative forces and thus the energy a real system would need to sustain the oscillations.

Within our conservative model, the impulsive transitions are conservative as well, since a lossless regenerative drive only redistributes the kinetic energy between the joints (Section 4.4.1). With such lossless drives, however, a rigid SCARA robot without springs would also be conservative, as it recovers its braking energy and needs no net energy over a closed pick-and-place sequence. The conservative model alone can therefore not show an efficiency gain of the CPEAs over a rigid SCARA robot. In a real system, the springs enable the quasi-conservative free oscillations between the pick and place position (Section 2.2). They provide the torque for decelerating and accelerating the links, which the motors of a rigid SCARA robot have to provide themselves. In real drives, the motor torque causes heat losses in the armature resistance that grow with the square of the torque[6, 7, 53, 64]. These losses were not considered in this work.

Quantifying the energy efficiency and comparing it with installed SCARA robots in factories would thus require a model of these losses as well as a realistic model of the torque pulse, the clutches and the non-conservative forces. Since the heat losses grow with the square of the torque, the integral (7.1) of the squared torque $\tau_{m,i}$ of each motor i over the sequence duration T_{seq} (Definition 8) could then be compared between both robots with identical motors.

$$\int_0^{T_{seq}} \tau_{m,i}^2(t)\,\mathrm{d}t \tag{7.1}$$

For the CPEAs, this integral only contains the torque pulses of the transitions and the compensation of the non-conservative forces during the quasi-conservative oscillations. As a baseline, a rigid SCARA robot would execute the same pick-and-place task set on a trajectory as typically used in industry, with its motor torques following from the left-hand side of (4.3). In hardware, the comparison could instead rely on the electrical energy that the drives of both robots draw from their power supply over the sequence. This measured energy would already contain all losses of the real system, such as the heat losses of the motors, friction and the energy for switching the clutches.

Since there is almost no sequence cost in some pick-and-place sequences such as the one in Figure 6.9, it is also worth analysing whether the transitions between oscillations can be performed solely by adapting the switching times of the clutches and in which workspace regions this is possible. The motors would then no longer have to apply the torque pulses, which would reduce the energy consumption.

8 Conclusion

The main goal of this thesis was to investigate the workspace regions in which a SCARA robot with clutched parallel elastic actuators can execute quasi-conservative pick-and-place sequences. When there are several suitable sequences for the same pick-and-place task set, the shortest sequence duration was considered.

In our model, the two clutches shift the equilibria of the springs by switching at different times and even form a region of equilibria in which no spring force acts on the robot. We analysed this model by implementing an optimization pipeline that searches for point-symmetric joint-space oscillations with fixed stiffness values between predefined configuration pairs of Cartesian pick and place positions mirrored about the x-axis. These oscillations swing around a shared equilibrium configuration $\boldsymbol{c}_m$ from Theorem 1. To analyse quasi-conservative pick-and-place sequences, we considered a 3×3 task set of Cartesian pick and place position pairs mirrored about the x-axis. We assumed that the motors of the CPEAs can perform impulsive transitions between the oscillations of consecutive tasks at $\boldsymbol{c}_m$. For different placements of the 3×3 task set across the workspace, we optimized for the lowest sequence cost with a GTSP.

Without distinguishing between the used centre configuration $\boldsymbol{c}_m$ and configuration pairs, the oscillations between pick and place positions mirrored about the x-axis cover the whole workspace. Depending on the chosen centre configuration $\boldsymbol{c}_m$, a single $\boldsymbol{c}_m$ with only one configuration pair already covers almost the whole workspace with oscillations. Since the impulsive transitions can be considered conservative when neglecting motor losses, quasi-conservative 3×3 pick-and-place sequences can be found across the whole workspace with sequence durations between 5 and 30 s.

Many pick-and-place position pairs have more than one oscillation with up to 49 and on average 6.37 for the same configuration pair and shared equilibrium. With these multiple oscillations, pick-and-place sequences other than the considered 3×3 sequence can choose the oscillations that suit them best. Clutched parallel elastic actuators could thus have great potential to make SCARA robots more energy-efficient and thereby contribute to more sustainable robot-based factories.

As discussed in Chapter 7, this thesis has also raised further questions that future work could examine. The search space still leaves room for more oscillations (Section 7.1). Changing the cost functions to treat joint angles differing by multiples of 2π as equal and using additional degrees of freedom such as the total energy, other fixed stiffness values, link lengths and masses could increase the number of found oscillations and reduce the transition and sequence costs. The relation between multiple oscillations of the same configuration pair and centre configuration $\boldsymbol{c}_m$ and an optimization for shorter period times in the found oscillations remain to be analysed. These improved period times should then be compared to conventional systems. Furthermore, other placements for the 3×3 task set as well as other pick-and-place task sets could be considered. In general, the considered pick-and-place tasks could be generalized to

tasks with arbitrary pick and place positions and include a payload (Section 7.2).

Since the concept was examined only in simulation and not in hardware, a more realistic model would be the next step. Such a model would also allow analysing the stability of the oscillations and designing a controller that keeps the robot on these oscillations while also controlling the clutch state (Section 7.3). A comparison of energy consumption of our approach to conventional systems in simulation still needs to be done. With a hardware setup, the consumption of electric energy could be compared and the setup could consider dwell times (Section 7.2).

Bibliography

[1] Alin Albu-Schäffer and Arne Sachtler. “What Can Algebraic Topology and Differential Geometry Teach Us About Intrinsic Dynamics and Global Behavior of Robots?” In: vol. 27. 2023, pp. 468–484.

[2] Jorge Angeles. *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*. 4th ed. 2014. Mechanical Engineering Series 124. Cham: Springer, 2014. 589 pp. ISBN: 978-3-319-01850-8 978-3-319-01851-5.

[3] V. I. Arnol’d. *Mathematical Methods of Classical Mechanics*. 2nd ed. Graduate Texts in Mathematics 60. New York: Springer-Verlag, 1989. 508 pp. ISBN: 978-0-387-96890-2.

[4] Rafael Balderas Hill, Sébastien Briot, Abdelhamid Chriette, and Philippe Martinet. “Minimizing the Energy Consumption of a Delta Robot by Exploiting the Natural Dynamics”. In: *ROMANSY 23 - Robot Design, Dynamics and Control*. Ed. by Gentiane Venture, Jorge Solis, Yukio Takeda, and Atsushi Konno. Cham: Springer International Publishing, 2021, pp. 213–221. ISBN: 978-3-030-58380-4.

[5] Rafael Balderas Hill, Sébastien Briot, Abdelhamid Chriette, and Philippe Martinet. “Performing Energy-Efficient Pick-and-Place Motions for High-Speed Robots by Using Variable Stiffness Springs”. In: *Journal of Mechanisms and Robotics* 14.5 (2022).

[6] J. P. Barreto, F. J.-F. Schöler, and B. Corves. “The Concept of Natural Motion for Pick and Place Operations”. In: *New Advances in Mechanisms, Mechanical Transmissions and Robotics*. Ed. by Burkhard Corves, Erwin-Christian Lovasz, Mathias Hüsing, Inocentiu Maniu, and Corina Gruescu. Cham: Springer International Publishing, 2017, pp. 89–98. ISBN: 978-3-319-45450-4.

[7] Juan Pablo Barreto and Burkhard Corves. “Resonant Delta Robot for Pick-and-Place Operations”. In: *Advances in Mechanism and Machine Science*. Ed. by Tadeusz Uhl. Cham: Springer International Publishing, 2019, pp. 2309–2318. ISBN: 978-3-030-20131-9.

[8] Richard Bellman. “Dynamic Programming Treatment of the Travelling Salesman Problem”. In: *Journal of the ACM* 9.1 (Jan. 1962), pp. 61–63.

[9] Filip Bjelonic, Arne Sachtler, Alin Albu-Schaffer, and Cosimo Della Santina. “Experimental Closed-Loop Excitation of Nonlinear Normal Modes on an Elastic Industrial Robot”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 1689–1696.

[10] Pierre Blanchard, Desmond J. Higham, and Nicholas J. Higham. *Accurate Computation of the Log-Sum-Exp and Softmax Functions*. Sept. 8, 2019. URL: `http://arxiv.org/abs/1909.03469` (visited on 10/20/2026). Pre-published.

[11] Paolo Boscariol and Dario Richiedei. “Energy-Efficient Design of Multipoint Trajectories for Cartesian Robots”. In: *The International Journal of Advanced Manufacturing Technology* 102.5 (June 1, 2019), pp. 1853–1870.

[12] Mary Ann Branch, Thomas F. Coleman, and Yuying Li. “A Subspace, Interior, and Conjugate Gradient Method for Large-Scale Bound-Constrained Minimization Problems”. In: *SIAM Journal on Scientific Computing* 21.1 (1999), pp. 1–23.

[13] R. P. Brent. *Algorithms for Minimization without Derivatives*. Prentice-Hall Series in Automatic Computation. Englewood Cliffs, N.J: Prentice-Hall, 1972. 195 pp. ISBN: 978-0-13-022335-7.

[14] Jan Brinker, Marco Lübbecke, Yukio Takeda, and Burkhard Corves. "Optimization of the Reconfiguration Planning of Handling Systems Based on Parallel Manipulators With Delta-Like Architecture". In: *IEEE Robotics and Automation Letters* 2.3 (July 2017), pp. 1802–1808.

[15] Luca Bruzzone, Matteo Verotti, and Pietro Fanghella. "A Comparative Study of Natural and Exact Elastic Balancing Methods for the RR-4R-R Manipulator". In: *Machines* 13.11 (Nov. 2025), p. 1023.

[16] Luca Bruzzone, Matteo Verotti, and Pietro Fanghella. "Dynamic Model and Performance Assessment of the Natural Motion of a SCARA-like Manipulator in Pick-and-Place Tasks". In: *Robotics and Autonomous Systems* 195 (Jan. 1, 2026), p. 105215.

[17] Richard L. Burden and J. Douglas Faires. *Numerical Analysis*. 9th ed. Boston, MA: Brooks/Cole, Cengage Learning, 2011. 872 pp. ISBN: 978-0-538-73351-9.

[18] Giovanni Carabin and Renato Vidoni. "Energy-Saving Optimization Method for Point-to-Point Trajectories Planned via Standard Primitives in 1-DoF Mechatronic Systems". In: *The International Journal of Advanced Manufacturing Technology* 116.1 (Sept. 1, 2021), pp. 331–344.

[19] Giovanni Carabin, Erich Wehrle, and Renato Vidoni. "A Review on Energy-Saving Optimization Methods for Robotic and Automatic Systems". In: *Robotics* 6.4 (Dec. 2017), p. 39.

[20] Yixi Chen, Evangelos Chatziandreou, Chase W. Mathews, Beau P. Johnson, and David J. Braun. "Design and Control of a Parallel Elastic Actuator With Adjustable Equilibrium Position". In: *IEEE Robotics and Automation Letters* 11.6 (June 2026), pp. 6512–6519.

[21] Christopher Clapham and James Nicholson. *The Concise Oxford Dictionary of Mathematics*. 4th ed. Oxford University Press, Jan. 1, 2009. ISBN: 978-0-19-923594-0.

[22] *CMAEvolutionStrategy*. URL: `https://cma-es.github.io/apidocs-pycma/cma.evolution_strategy.CMAEvolutionStrategy.html` (visited on 09/29/2026).

[23] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Third edition. Pearson New International Edition. Harlow: Pearson, 2014. 1 p. ISBN: 978-1-292-04004-2 978-1-292-05252-6.

[24] Qi Cui, ShiWen Yao, Chenyu Meng, Mahuaqing Zuo, and Yu Liu. "Energy and Economic Consequences of Large-Scale Industrial Robot Applications in China's Manufacturing Industry". In: *Structural Change and Economic Dynamics* 77 (Apr. 1, 2026), pp. 230–247.

[25] Cosimo Della Santina, Dominic Lakatos, Antonio Bicchi, and Alin Albu-Schaeffer. "Using Nonlinear Normal Modes for Execution of Efficient Cyclic Motions in Articulated Soft Robots". In: *Springer Proceedings in Advanced Robotics*. Vol. 19. Springer Science and Business Media B.V., 2021, pp. 566–575.

[26] *Directive (EU) 2023/1791 of the European Parliament and of the Council of 13 September 2023 on Energy Efficiency and Amending Regulation (EU) 2023/955 (Recast)*. Sept. 20, 2023.

[27] Mingyuan Dou, Ning He, Lile He, and Jiaxuan Chen. "Parallel Elastic Ankle Actuator With a Unidirectional Clutch for Explosive Motion in Legged Robots". In: *IEEE Robotics and Automation Letters* 11.5 (May 2026), pp. 6202–6209.

[28] Epson. *Epson GX20B SCARA Robot - 1000mm*. Epson US. URL: https://epson.com/For-Work/Robots/SCARA/Epson-GX20B-SCARA-Robot---1000mm/p/RGX20-BA01SSTE (visited on 11/13/2026).

[29] eurostat. *Final Energy Consumption in Industry - Detailed Statistics*. Statistics Explained. May 2026. URL: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Final_energy_consumption_in_industry_-_detailed_statistics (visited on 09/20/2026).

[30] *Evolution_strategy*. URL: https://cma-es.github.io/apidocs-pycma/cma.evolution_strategy.html#fmin2 (visited on 09/29/2026).

[31] Giuliano Fabris, Lorenzo Scalera, and Alessandro Gasparetto. "Dynamic Modelling and Energy-Efficiency Optimization in a 3-DOF Parallel Robot". In: *The International Journal of Advanced Manufacturing Technology* 132.5 (May 1, 2024), pp. 2677–2699.

[32] Roy Featherstone. *Rigid Body Dynamics Algorithms*. New York: Springer, 2008. 272 pp. ISBN: 978-0-387-74314-1 978-0-387-74315-8.

[33] F. N. Fritsch and J. Butland. "A Method for Constructing Local Monotone Piecewise Cubic Interpolants". In: *SIAM Journal on Scientific and Statistical Computing* 5.2 (June 1984), pp. 300–304.

[34] Raphael Rustici Garcia, André Carvalho Bittencourt, and Emilia Villani. "Relevant Factors for the Energy Consumption of Industrial Robots". In: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 40.9 (Sept. 3, 2018), p. 464.

[35] Yann Gilpin, Vince Kurtz, and Hai Lin. "A Smooth Robustness Measure of Signal Temporal Logic for Symbolic Control". In: *IEEE Control Systems Letters* 5.1 (Jan. 2021), pp. 241–246.

[36] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Fourth edition. Johns Hopkins Studies in the Mathematical Sciences. Baltimore: The Johns Hopkins University Press, 2013. 756 pp. ISBN: 978-1-4214-0794-4.

[37] Hidemasa Goya, Kento Matsusaka, Mitsunori Uemura, Yasutaka Nishioka, and Sadao Kawamura. "Realization of High-Energy Efficient Pick-and-Place Tasks of SCARA Robots by Resonance". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. Oct. 2012, pp. 2730–2735.

[38] D. F. B. Haeufle, M. D. Taylor, S. Schmitt, and H. Geyer. "A Clutched Parallel Elastic Actuator Concept: Towards Energy Efficient Powered Legs in Prosthetics and Robotics". In: *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. 2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob). June 2012, pp. 1614–1619.

[39] E. Hairer, S. P. Nørsett, and Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. 2nd rev. ed. Springer Series in Computational Mathematics 8. Heidelberg ; London: Springer, 2009. 528 pp. ISBN: 978-3-540-56670-0 978-3-642-05163-0 978-3-540-78862-1.

[40] Nikolaus Hansen. *The CMA Evolution Strategy: A Tutorial*. Mar. 10, 2023. URL: http://arxiv.org/abs/1604.00772 (visited on 09/29/2026). Pre-published.

[41] Charles R. Harris et al. "Array Programming with NumPy". In: *Nature* 585.7825 (Sept. 17, 2020), pp. 357–362.

[42] Michael Held and Richard M. Karp. “A Dynamic Programming Approach to Sequencing Problems”. In: *Proceedings of the 1961 16th ACM National Meeting On* -. The 1961 16th ACM National Meeting. Not Known: ACM Press, 1961, pp. 71.201–71.204.

[43] Rafael Balderas Hill, Sébastien Briot, Abdelhamid Chriette, and Philippe Martinet. “Increasing Energy Efficiency of High-Speed Parallel Robots by Using Variable Stiffness Springs and Optimal Motion Generation”. In: *Proc. ASME Des. Eng. Tech. Conf.* Proceedings of the ASME Design Engineering Technical Conference. Vol. 5B-2018. American Society of Mechanical Engineers (ASME), 2018. ISBN: 978-0-7918-5181-4.

[44] Rafael Balderas Hill, Sébastien Briot, Abdelhamid Chriette, and Philippe Martinet. “Minimizing Input Torques of a High-Speed Five-Bar Mechanism by Using Variable Stiffness Springs”. In: *ROMANSY 22 – Robot Design, Dynamics and Control*. Ed. by Vigen Arakelian and Philippe Wenger. Cham: Springer International Publishing, 2019, pp. 61–68. ISBN: 978-3-319-78963-7.

[45] John D. Hunter. “Matplotlib: A 2D Graphics Environment”. In: *Computing in Science & Engineering* 9.3 (May 2007), pp. 90–95.

[46] International Federation of Robotics. *Record of 4 Million Robots in Factories Worldwide*. International Federation of Robotics. Sept. 24, 2024. URL: `https://ifr.org/ifr-press-releases/news/record-of-4-million-robots-working-in-factories-worldwide` (visited on 09/20/2026).

[47] International Federation of Robotics. *TOP 5 Global Robotics Trends 2025*. International Federation of Robotics. Jan. 22, 2025. URL: `https://ifr.org/ifr-press-releases/news/top-5-global-robotics-trends-2025` (visited on 09/20/2026).

[48] Makoto Iwamura, Shunsuke Ide, Yeongju Baek, Yoshiki Maeda, and Kodai Ryomoto. “Minimum Energy Control of Robot Manipulators Using a Simple Variable Stiffness Mechanism”. In: *Optimal Design and Control of Multibody Systems*. Ed. by Karin Nachbagauer and Alexander Held. Cham: Springer Nature Switzerland, 2024, pp. 110–119. ISBN: 978-3-031-50000-8.

[49] Amir Jafari, Nikos G. Tsagarakis, and Darwin G. Caldwell. “Exploiting Natural Dynamics for Energy Minimization Using an Actuator with Adjustable Stiffness (AwAS)”. In: *2011 IEEE International Conference on Robotics and Automation*. 2011 IEEE International Conference on Robotics and Automation. May 2011, pp. 4632–4637.

[50] Kawasaki Robotics. *Pick & Place Robots*. Kawasaki Robotics. URL: `https://kawasakirobotics.com/applications/pick-and-place/` (visited on 09/20/2026).

[51] G. Kerschen, M. Peeters, J. C. Golinval, and A. F. Vakakis. “Nonlinear Normal Modes, Part I: A Useful Framework for the Structural Dynamicist”. In: *Mechanical Systems and Signal Processing*. Special Issue: Non-linear Structural Dynamics 23.1 (Jan. 1, 2009), pp. 170–194.

[52] Hassan K. Khalil. *Nonlinear Systems*. 3rd ed. Upper Saddle River, N.J: Prentice Hall, 2002. 750 pp. ISBN: 978-0-13-067389-3.

[53] Thomas Kordik, Hubert Gattringer, and Andreas Müller. “Simultaneous Stiffness and Trajectory Optimization for Energy Minimization of Pick-and-Place Tasks of Series Elastic Actuators-Actuated Parallel Kinematic Manipulators”. In: *Journal of Computational and Nonlinear Dynamics* 20.8 (2025).

[54] Erez Krimsky and Steven H. Collins. “Elastic Energy-Recycling Actuators for Efficient Robots”. In: *Science Robotics* 9.88 (Mar. 20, 2024), eadj7246.

[55] Jeroen S. W. Lamb and John A. G. Roberts. “Time-Reversal Symmetry in Dynamical Systems: A Survey”. In: *Physica D: Nonlinear Phenomena*. Proceedings of the Workshop on Time-Reversal Symmetry in Dynamical Systems 112.1 (Jan. 15, 1998), pp. 1–39.

[56] Lev Davidovič Landau and Evgenij M. Lifšic. *Mechanics*. 3. ed., repr. Course of theoretical physics / L. D. Landau and E. M. Lifshitz 1. Amsterdam Heidelberg: Elsevier, Butterworth-Heinemann, 2011. 170 pp. ISBN: 978-0-7506-2896-9.

[57] Xin Liu, Anthony Rossi, and Ioannis Poulakakis. “A Switchable Parallel Elastic Actuator and Its Application to Leg Design for Running Robots”. In: *IEEE/ASME Transactions on Mechatronics* 23.6 (Dec. 2018), pp. 2681–2692.

[58] Xin Liu, Anthony Rossi, and Ioannis Poulakakis. “SPEAR: A Monopedal Robot with Switchable Parallel Elastic Actuation”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Sept. 2015, pp. 5142–5147.

[59] Yanjie Liu, Le Liang, Haijun Han, and Shijie Zhang. “A Method of Energy-Optimal Trajectory Planning for Palletizing Robot”. In: *Mathematical Problems in Engineering* 2017.1 (2017), p. 5862457.

[60] Kevin Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*. Cambridge, UK: Cambridge University Press, 2017. 528 pp. ISBN: 978-1-107-15630-2 978-1-316-60984-2.

[61] Wes McKinney. “Data Structures for Statistical Computing in Python”. In: Python in Science Conference. Austin, Texas, 2010, pp. 56–61.

[62] Davis Meike and Leonids Ribickis. “Energy Efficient Use of Robotics in the Automobile Industry”. In: *2011 15th International Conference on Advanced Robotics (ICAR)*. 2011 15th International Conference on Advanced Robotics (ICAR). June 2011, pp. 507–511.

[63] Davis Meike and Leonids Ribickis. “Recuperated Energy Savings Potential and Approaches in Industrial Robotics”. In: *2011 IEEE International Conference on Automation Science and Engineering*. 2011 IEEE International Conference on Automation Science and Engineering. Aug. 2011, pp. 299–303.

[64] Christian Mirz, Frédéric Schöler, Juan Pablo Barreto, and Burkhard Corves. “Optimal Control Based Path Planning for Parallel Kinematic Manipulators Utilising Natural Motion”. In: *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*. 2018 IEEE 14th International Conference on Automation Science and Engineering (CASE). Aug. 2018, pp. 223–228.

[65] Juan Pablo Mora, Juan Pablo Barreto, and Carlos F. Rodriguez. “Energy Optimization of a Parallel Robot in Pick and Place Tasks”. In: *Mech. Mach. Sci.* Mechanisms and Machine Science. Vol. 110 MMS. Springer Science and Business Media B.V., 2022, pp. 191–200. ISBN: 978-3-030-88750-6.

[66] Juan Pablo Mora, Carlos F. Rodriguez, and Burkhard Corves. “Exploiting Natural Dynamics and Trajectory Planning in Parallel Manipulators for Energy-Efficient Pick-and-Place Tasks”. In: *Multibody Mechatronic Systems*. Ed. by Mario Acevedo, Leonardo Valdivia, and Hiram Ponce. Cham: Springer Nature Switzerland, 2026, pp. 132–141. ISBN: 978-3-032-10862-3.

[67] Juan Pablo Mora, Carlos F. Rodriguez, and Burkhard Corves. “Natural Motion Design for Energy-Efficient Pick-and-Place Scenarios”. In: *Machines* 14.3 (Mar. 2026), p. 330.

[68] Jorge J. Moré. "The Levenberg-Marquardt Algorithm: Implementation and Theory". In: *Numerical Analysis*. Ed. by G. A. Watson. Berlin, Heidelberg: Springer, 1978, pp. 105–116. ISBN: 978-3-540-35972-2.

[69] MuJoCo. *Computation*. MuJoCo Documentation. URL: `https://mujoco.readthedocs.io/en/stable/computation/index.html` (visited on 10/19/2026).

[70] MuJoCo. *Globals*. MuJoCo Documentation. URL: `https://mujoco.readthedocs.io/en/stable/APIreference/APIglobals.html#mjcb-control` (visited on 10/19/2026).

[71] Christopher Müller. *World Robotics 2025 – Industrial Robots (Executive Summary)*. Frankfurt am Main, Germany: IFR Statistical Department, VDMA Services GmbH, 2025.

[72] Johannes Muru and Anton Rassõlkin. "A Scoping Review of Energy Consumption in Industrial Robotics". In: *Machines* 13.7 (July 2025), p. 542.

[73] Duc-Vinh Nguyen, Mohamed Jebahi, Victor Champaney, and Francisco Chinesta. "Identification of Material Parameters in Low-Data Limit: Application to Gradient-Enhanced Continua". In: *International Journal of Material Forming* 17.1 (Jan. 2, 2024), p. 10.

[74] *Options_parameters*. URL: `https://cma-es.github.io/apidocs-pycma/cma.options_parameters.html#cma_default_options_` (visited on 09/29/2026).

[75] Ilaria Palomba, Erich Wehrle, Giovanni Carabin, and Renato Vidoni. "Minimization of the Energy Consumption in Industrial Robots through Regenerative Drives and Optimally Designed Compliant Elements". In: *Applied Sciences* 10.21 (Jan. 2020), p. 7475.

[76] Bernhard Penzlin, Mustafa Enes Fincan, Yinbo Li, Linhong Ji, Steffen Leonhardt, and Chuong Ngo. "Design and Analysis of a Clutched Parallel Elastic Actuator". In: *Actuators* 8.3 (Sept. 2019), p. 67.

[77] Michiel Plooij, Marvin van Nunspeet, Martijn Wisse, and Heike Vallery. "Design and Evaluation of the Bi-directional Clutched Parallel Elastic Actuator (BIC-PEA)". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 2015 IEEE International Conference on Robotics and Automation (ICRA). May 2015, pp. 1002–1009.

[78] Michiel Plooij, Martijn Wisse, and Heike Vallery. "Reducing the Energy Consumption of Robots Using the Bidirectional Clutched Parallel Elastic Actuator". In: *IEEE Transactions on Robotics* 32.6 (Dec. 2016), pp. 1512–1523.

[79] Michiel Plooij, Wouter Wolfslag, and Martijn Wisse. "Clutched Elastic Actuators". In: *IEEE/ASME Transactions on Mechatronics* 22.2 (Apr. 2017), pp. 739–750.

[80] Petrică C. Pop, Ovidiu Cosma, Cosmin Sabo, and Corina Pop Sitar. "A Comprehensive Survey on the Generalized Traveling Salesman Problem". In: *European Journal of Operational Research* 314.3 (May 1, 2024), pp. 819–835.

[81] Arne Sachtler and Alin Albu-Schäffer. "Strict Modes Everywhere - Bringing Order Into Dynamics of Mechanical Systems by a Potential Compatible With the Geodesic Flow". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 2337–2344.

[82] Arne Sachtler, Davide Calzolari, Maximilian Raff, Annika Schmidt, Yannik P. Wotte, Cosimo Della Santina, C. David Remy, and Alin Albu-Schäffer. "Swing-Up of a Weakly Actuated Double Pendulum via Nonlinear Normal Modes". In: *European Control Conf., ECC*. 2024 European Control Conference, ECC 2024. Institute of Electrical and Electronics Engineers Inc., 2024, pp. 2392–2398. ISBN: 978-3-907144-10-7.

[83] Cosimo Della Santina, Yannik Wotte, Arne Sachtler, and Alin Albu-Scäffer. *Sufficient Conditions for an Eigenmanifold to Be of the Extended Rosenberg Type*. Mar. 9, 2022. URL: `https://www.researchsquare.com/article/rs-1416735/v1` (visited on 06/17/2026). Pre-published.

[84] Lorenzo Scalera, Ilaria Palomba, Erich Wehrle, Alessandro Gasparetto, and Renato Vidoni. "Natural Motion for Energy Saving in Robotic and Mechatronic Systems". In: *Applied Sciences* 9.17 (Jan. 2019), p. 3516.

[85] Bruno Siciliano, ed. *Robotics: Modelling, Planning and Control*. Advanced Textbooks in Control and Signal Processing. London: Springer, 2009. 632 pp. ISBN: 978-1-84628-641-4 978-1-84628-642-1.

[86] Emanuel Todorov, Tom Erez, and Yuval Tassa. "MuJoCo: A Physics Engine for Model-Based Control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2012). Vilamoura-Algarve, Portugal: IEEE, Oct. 2012, pp. 5026–5033. ISBN: 978-1-4673-1736-8 978-1-4673-1737-5 978-1-4673-1735-1.

[87] Agajan Torayev, Giovanna Martínez-Arellano, Jack C. Chaplin, David Sanderson, and Svetan Ratchev. "Online and Modular Energy Consumption Optimization of Industrial Robots". In: *IEEE Transactions on Industrial Informatics* 20.2 (Feb. 2024), pp. 1198–1207.

[88] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17.3 (Mar. 2, 2020), pp. 261–272.

[89] Alexandra Velasco Vivas, Antonello Cherubini, Manolo Garabini, Paolo Salaris, and Antonio Bicchi. "Minimizing Energy Consumption of Elastic Robots in Repetitive Tasks". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 53.8 (Aug. 2023), pp. 5006–5018.

[90] Wenpan Wang, Zirui Chen, Linghao Yu, Lei Li, Shengfeng Zhu, Shiwu Zhang, Weihua Li, and Shuaishuai Sun. "Design and Validation of a Quasi-Passive Knee Exoskeleton with Clutched Elastic Actuators for Human Walking Assistance". In: *Biomimetic Intelligence and Robotics* 6.2 (June 1, 2026), p. 100294.

[91] Giri Wahyu Wiriasto, Siti Sendari, Dyah Lestari, Muhamad Syamsu Iqbal, and Norrima Mochtar. "Energy-Conscious Trajectory Methods for Robotic Manipulators: A Systematic Review". In: *International Journal of Robotics and Control Systems* 6.1 (Dec. 23, 2025), pp. 54–78.

[92] Václav Záda and Květoslav Belda. "Mathematical Modeling of Industrial Robots Based on Hamiltonian Mechanics". In: *2016 17th International Carpathian Control Conference (ICCC)*. 2016 17th International Carpathian Control Conference (ICCC). May 2016, pp. 813–818.

List of Figures

3.1. Planar 2-DoF Robot . . . 10
3.2. Loop of a Randomized Black-Box Search . . . 17
3.3. Search Distribution of CMA-ES over Several Generations . . . 18
3.4. Generalized Hamiltonian Tour . . . 19

4.1. Pick-and-Place Motion between Cartesian Points A and B . . . 20
4.2. Top View of the SCARA Robot Model . . . 22
4.3. SCARA Manipulator with CPEAs . . . 23
4.4. Schematic of a CPEA Joint . . . 23
4.5. Configurations of the Points A and B . . . 25
4.6. Trajectory from A through B . . . 26
4.7. Joint-Space Point-Symmetric Oscillation in Cartesian and Joint Space . . . 27
4.8. Transition between Two Oscillations at $\boldsymbol{c}_m$ . . . 34
4.9. Placement Modes of the 3×3 Task Set . . . 36

5.1. Optimization Pipeline . . . 37
5.2. MuJoCo Model of the SCARA Robot . . . 38
5.3. Potential Field of the Clutches . . . 40
5.4. Total Energy for Different Integrators . . . 41
5.5. Total Energy with RK4 and the Control Function . . . 42
5.6. Total Energy with RK4 and Event Location . . . 42
5.7. Joint-Space Trajectory of the No-Spring Optimization . . . 43
5.8. Multi-Turn Trajectory of the No-Spring Optimization . . . 44
5.9. Robustness Analysis of the No-Spring Optimization . . . 45
5.10. Stiffness Optimization Pipeline . . . 47
5.11. Joint-Space Trajectory of the Stiffness Optimization . . . 48
5.12. Joint-Space Trajectory of the No-Spring Optimization from $\boldsymbol{c}_m$ . . . 50
5.13. Evaluation of a CMA-ES Sample in the Stiffness Optimization . . . 51
5.14. Steps after Each CMA-ES Generation in the Stiffness Optimization . . . 52
5.15. Workspace Coverage of the Stiffness Optimization . . . 55
5.16. Joint-Space Trajectory of the Clutch Switching Times Optimization with Decision Variables . . . 56
5.17. Stiffness Values along the Continuation over the Radius . . . 58
5.18. Algorithm to Group the Members of a Raster Node . . . 59
5.19. Processing Order of the Raster Continuation . . . 60
5.20. Optimization of an Initial Guess in the Clutch Switching Times Optimization . . . 61
5.21. Off-Raster Node between Four Raster Positions . . . 63

6.1. Successful Members per Raster Node . . . 65
6.2. Optimized Joint Configuration Representations for the Rasters with $c_{m,1} = -\pi$ and Configuration Index 0 . . . 66
6.3. Multiple Oscillations for One Configuration Pair and Centre Configuration . . . 68
6.4. Fastest Oscillation per Raster Position . . . 70
6.5. Bouncing Trajectory . . . 72

6.6. Sensitive Point-Symmetric Oscillation 73
6.7. Accuracy of the Fastest Oscillation per Raster Node 74
6.8. Pick-and-Place Sequences with Varying Initial Velocity Angles 76
6.9. Pick-and-Place Sequence with Low Sequence Cost 77
6.10. Pick-and-Place Sequence per Task Set Placement 78
6.11. Lowest Sequence Cost per Centre Configuration for the 3×3 Axis-Aligned Task Set 81

A.1. Lowest Sequence Cost per Centre Configuration for the 3×3 Radially Aligned Task Set III

A Appentix

A.1. MJCF Model

Listing A.1 contains the MJCF model from Section 5.1.1. The bodies `arm1` and `arm2` are the links 1 and 2 with the joints `tele_joint1` and `tele_joint2`. The body `gripper_base` forms the gripper with its two cylinders. Both links use the mesh `1st_arm.stl`, from which MuJoCo computes their CoM and inertia tensor in Table 5.3. The bodies `Start` and `Target` mark the points A and B, which serve as the pick and place positions. Like all other bodies, they are fixed to the world without collisions and thus do not affect the dynamics.

```
<?xml version='1.0' encoding='utf-8'?>
<mujoco>
  <compiler angle="degree" />
  <option gravity="0 0 0" timestep="0.001" integrator="RK4" />

  <visual>
    <headlight diffuse="0.6 0.6 0.6" ambient="0.3 0.3 0.3" specular="0 0 0" />
    <rgba haze="0.15 0.25 0.35 1" />
    <global azimuth="60" elevation="-20" />
  </visual>

  <asset>
    <texture type="skybox" builtin="gradient" rgb1="0.3 0.5 0.7" rgb2="0 0 0" width="512" height="3072" />

    <texture type="2d" name="groundplane" builtin="checker" mark="edge" rgb1="0.2 0.3 0.4" rgb2="0.1 0.2
        0.3" markrgb="0.2 0.3 0.4" width="300" height="300" />
    <material name="groundplane" texture="groundplane" texuniform="true" texrepeat="10 10" reflectance="
        0.3" />
  </asset>

  <worldbody>
    <light pos="0 0 1.5" dir="0 0 -1" directional="true" />
    <geom name="floor" size="0 0 0.0001" type="plane" material="groundplane" pos="0 0 0.0" contype="0"
        conaffinity="0" />

    <body name="origin_axes" pos="0 0 0">
      <geom type="capsule" size="0.0005 0.05" rgba="1 0 0 1" fromto="0 0 0 0.1 0 0" contype="0"
          conaffinity="0" />
      <geom type="capsule" size="0.0005 0.05" rgba="0 1 0 1" fromto="0 0 0 0 0.1 0" contype="0"
          conaffinity="0" />
      <geom type="capsule" size="0.0005 0.05" rgba="0 0 1 1" fromto="0 0 0 0 0 0.1" contype="0"
          conaffinity="0" />
      <site name="origin" pos="0 0 0" />
    </body>

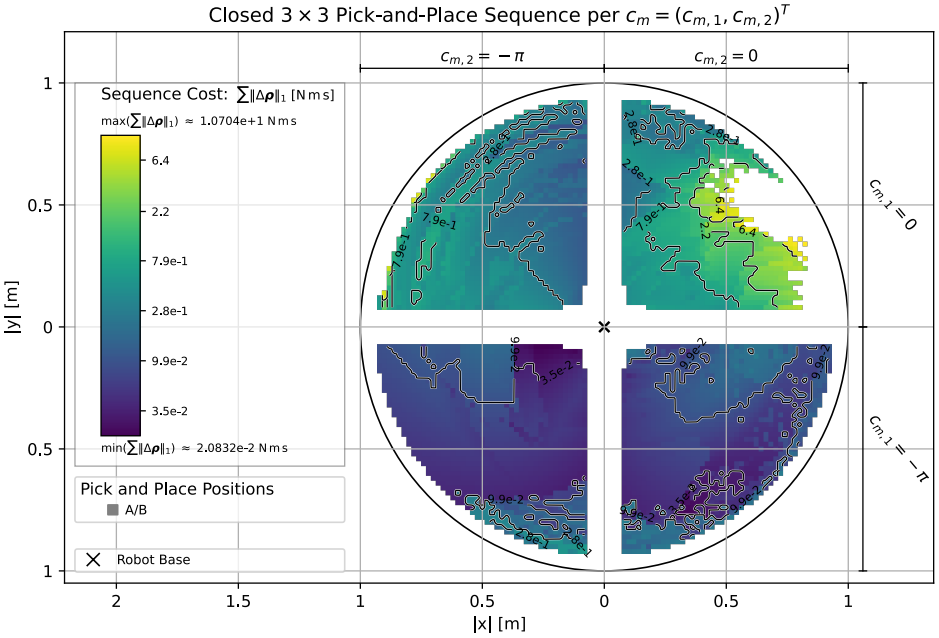
    <body name="Start" pos="0.6427 0.5 0.03" euler="0 0 0">
       <geom name="sphere_green" type="sphere" size="0.01" rgba="0 1 0 0.9" density="650" friction="0.0
           0.0 0.0" solref="0.0 0" solimp="0.0 0.0 0.0 0.0" contype="0" conaffinity="0" />
    </body>

```

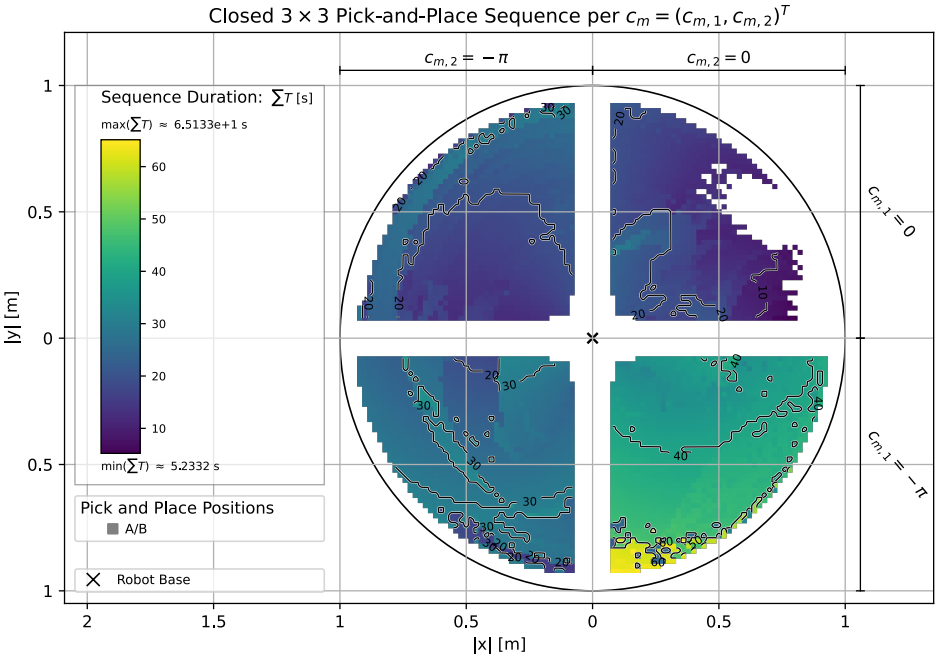
```
33      <body name="Target" pos="0.6427 -0.5 0.03" euler="0 0 0">
34          <geom name="sphere_blue" type="sphere" size="0.01" rgba="0 0 1 0.9" density="650" friction="0.0
                0.0 0.0" solref="0.0 0" solimp="0.0 0.0 0.0 0.0" contype="0" conaffinity="0" />
35      </body>
36
37      <body name="robot" pos="0.2115 0 0.0">
38
39              <body name="robotbase" pos="0 0 0.28" euler="0 180 0">
40                  <geom name="steppers" type="mesh" mesh="steppers" rgba="0.4 0.4 0.4 1" contype="0"
                        conaffinity="0" />
41                  <geom type="mesh" mesh="base_stepper" rgba="0.65 0.45 0.15 1" contype="0" conaffinity="0" /
                        >
42
43
44                  <body name="arm1" pos="0.2115 0 0.1" euler="0 180 0">
45                      <geom name="arm1" type="mesh" mesh="arm1" rgba="0.65 0.45 0.15 1" mass="2" contype="0"
                            conaffinity="0" />
46                      <joint name="tele_joint1" type="hinge" axis="0 0 1" pos="0 0 0" damping="0"
                            frictionloss="0" />
47                      <site name="vis1" />
48
49
50                      <body name="arm2" pos="0.5 0 -0.12" euler="0 0 0">
51                          <geom name="arm2" type="mesh" mesh="arm1" rgba="0.65 0.45 0.15 1" mass="2" contype=
                                "0" conaffinity="0" />
52                          <joint name="tele_joint2" type="hinge" axis="0 0 1" pos="0 0 0" damping="0"
                                frictionloss="0" />
53                          <site name="vis2" />
54
55
56                          <body name="gripper_base" pos="0.5 0 0">
57
58                              <geom name="gripper_base" type="cylinder" size="0.06 0.02" rgba="0.2 0.2 0.2 1"
                                    mass=".3" friction="0 0 0" contype="0" conaffinity="0" />
59                              <geom name="magnet" type="cylinder" pos="0 0 0.03" size="0.06 0.01" rgba="0.2
                                    0.2 0.2 1" mass=".3" friction="0 0 0" contype="0" conaffinity="0" />
60                              <site name="tcp" pos="0 0 0.03 " />
61                          </body>
62                      </body>
63                  </body>
64              </body>
65          </body>
66    </worldbody>
67
68    <asset>
69        <mesh name="steppers" file="steppers.stl" scale="0.001 0.001 0.001" />
70        <mesh name="arm1" file="1st_arm.stl" scale="0.005 0.005 0.005" />
71        <mesh name="arm2" file="2d_arm.stl" scale="0.005 0.005 0.005" />
72        <mesh name="base_stepper" file="base_stepper.stl" scale="0.005 0.005 0.005" />
73    </asset>
74
75    <sensor>
76
77        <jointpos name="angle_arm1" joint="tele_joint1" />
78        <jointpos name="angle_arm2" joint="tele_joint2" />
79    </sensor>
80
81
82
83  </mujoco>
```

Listing A.1: MJCF model of the SCARA robot

A.2. Radially Aligned Placement



(a) Sequence Cost



(b) Sequence Duration

Figure A.1.: Lowest sequence cost per centre configuration for the 3 × 3 radially aligned task set. Each quadrant in the figures belongs to one centre configuration $\boldsymbol{c}_m$.