CONTROL OF MULTI-CONTACT SYSTEMS VIA LOCAL HYBRID MODELS

Alp Aydinoglu

A DISSERTATION

in

Electrical and Systems Engineering

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2023

Supervisor of Dissertation

Michael Posa, Assistant Professor of Mechanical Engineering and Applied Mechanics

Graduate Group Chairperson

Troy Olsson, Associate Professor of Electrical and Systems Engineering

Dissertation Committee

Manfred Morari (Chair), Practice Professor of Electrical and Systems Engineering, Peter and Susanne Armstrong Faculty Fellow
George J. Pappas, UPS Foundation Professor of Electrical and Systems Engineering
Zachary Manchester, Assistant Professor of Robotics Institute, Carnegie Mellon University

To all that are curious,

ACKNOWLEDGEMENT

I want to thank Michael Posa for giving me the opportunity to be a part of Dynamic Autonomy and Intelligent Robotics Lab. He has been a great mentor and I have learned so much from him over the years.

Thank you to members of my committee: Manfred Morari, George Pappas and Zachary Manchester. I would especially like to thank Manfred for his guidance and support.

Thank you to all of my labmates and collaborators. It was fun to talk about research with you.

Thank you to my undergraduate advisors, particularly Neslihan Serap Sengor for her mentorship. Neuroscience Modeling and Research Group has always been an inspiration for me.

Most importantly, thank you to my friends and family. I love you all.

ABSTRACT

CONTROL OF MULTI-CONTACT SYSTEMS VIA LOCAL HYBRID MODELS

Alp Aydinoglu

Michael Posa

For many important tasks such as manipulation and locomotion, robots need to make and break contact with their environment. Although such multi-contact systems are common, they pose a significant challenge when it comes to analysis and control. This difficulty primarily stems from two key factors: 1) the rapid increase in the number of possible ways that a system can move or behave as the number of contacts increase (as a result of the hybrid structure), and 2) the inherent nonlinearities present in the system's dynamics. As a result, for tasks which require initiating contact with the environment, many state-of-the-art methods struggle as the number of contacts increase.

Considering the substantial difficulty of multi-contact problems, it's only natural to raise the question: How can we solve such problems? In addressing this query, this thesis directs its attention toward the simplification of multi-contact problems. It does so by concentrating on local hybrid approximations, wherein the non-smooth, hybrid structure is retained, while linearizing the smooth elements within the dynamics to mitigate the complexities arising from nonlinearities. As a result, we focus on local hybrid models called linear complementarity systems which are simple models that qualitatively capture the underlying non-smooth, hybrid structure.

Employing these local hybrid models, this thesis presents scalable and fast algorithmic solutions for challenging multi-contact problems. First, we present the first real-time MPC framework for multi-contact manipulation. The method is based on the alternating direction method of multipliers (ADMM) and is capable of high-speed reasoning over potential contact events. Then, we focus on utilizing tactile measurements for reactive control, which is very natural yet underexplored in the robotics community. We propose a control framework to design tactile feedback policies for

multi-contact systems by exploiting the local complementarity structure of contact dynamics. This framework can close the loop on tactile sensors and it is non-combinatorial, enabling optimization algorithms to automatically synthesize provably stable control policies. Then, inspired by the connection between rectified linear unit (ReLU) activation functions and linear complementarity problems, we present a method to analyze stability of multi-contact systems in feedback with ReLU network controllers.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

In recent years, robotic automation has excelled in dealing with repetitive tasks in static and structured environments. On the other hand, as robots slowly become an integral part of our society, some will operate alongside humans, such as caring for elderly people, and most will need to perform in complex, unstructured environments in an agile manner. To achieve the promise of the field, robots must effectively and safely reason about the physical interactions between themselves and the environment (e.g. people, robots, inanimate objects). We refer to such physical interactions as *contact* and many important tasks require reasoning about multiple contact interactions and are *multi-contact* in nature. For example, dexterous manipulation and legged locomotion, fundamentally require intentionally initiating contact with the environment to achieve positive results.

Even though multi-contact systems are common, they are notoriously hard to analyze or control (Bemporad et al., 2000; Lauer, 2015), primarily because of two main reasons: 1) the rapid increase in the number of possible ways that a system can move or behave as the number of contacts increase (*hybrid* structure), 2) inherent nonlinearities in dynamics. To clarify the first point, consider the task of pushing a box standing on a table. This simple system has three *modes*: sliding left, sliding right and sticking. Now assume that one wants to plan a trajectory looking ten steps ahead into the future. There are roughly sixty thousand potential outcomes one needs to consider because there is a need to decide if the box is sliding left, sliding right or sticking at every planning step. This is not surprising as a trajectory optimization problem with a multi-contact system is an *NP-hard* problem (Pia et al., 2017). Regardless of these difficulties, researchers have developed methods to perform stability analysis, design feedback control policies and discover interesting trajectories (Johansson, 2003; Ferrari-Trecate et al., 2002; Posa et al., 2015, 2014; Pang et al., 2023; Shirai et al., 2022; Önol et al., 2019; Winkler et al., 2018; Cheng et al., 2022; Raghunathan et al., 2022) but existing methods struggle as the number of contacts increase.

As multi-contact problems are extremely challenging, it is natural to ask: How can one simplify

the problem in a meaningful way? Universally, the most straightforward way of simplifying a dynamical system is via *linearization* (Kuznetsov et al., 1998). For smooth systems, linearization is really effective as it captures the first-order behavior of the system around the chosen operating point and unlike multi-contact systems there is no need to reason about neighboring modes. Also, there are plethora of tools to analyze and control linear systems (Chen, 1995; Garcia et al., 1989; Borrelli et al., 2017; Zhakatayev et al., 2017; Ding et al., 2019). However, for multi-contact systems, it is critical to reason about making and breaking contact. Unfortunately, this is not possible using linear models because inherently a linearization is local to a single mode (i.e. not hybrid). Throughout the thesis, we demonstrate that non-smooth multi-contact dynamics fundamentally cannot be captured by linearization and removing the hybrid structure is not meaningful.

In consideration of these challenges, this thesis focuses on simplifying the multi-contact problems by focusing on local hybrid approximations where we keep the non-smooth, hybrid structure but linearize the smooth components in the dynamics. As a result, we obtain local hybrid models called *linear complementarity systems* (Heemels et al., 2000). Focusing on these simple but expressive models, this thesis presents scalable and fast algorithmic solutions for challenging multi-contact problems. We develop methods to solve (adaptive) model predictive control problems at real-time rates, design tactile feedback controllers and perform stability analysis for multi-contact systems in closed-loop with neural network controllers.

1.1. Outline and Contributions

We start in Chapter 2 and summarize works related to stability analysis, control and planning for multi-contact systems.

In Chapter 3, we introduce the necessary background on multi-contact dynamics, linear complementarity problems, linear complementarity systems and sum-of-squares programming.

Our main contributions start with Chapter 4, where we present the first real-time MPC framework for multi-contact manipulation. We propose an algorithm, consensus complementarity control (C3), for solving the hybrid MPC problem approximately for multi-contact systems (Aydinoglu and Posa,

2022; Aydinoglu et al., 2023). We exploit the distributed nature of ADMM (combined with LCS approximations) and demonstrate that the hard part of the problem, reasoning about contact events, can be parallelized. This enables our algorithm to be fast, robust to disturbances and also minimizes the effect of control horizon on the run-time of the algorithm. In addition, we present a real-time adaptive learning framework that combines consensus complementary control with LCS learning (Jin et al., 2022). We also present state-of-the-art experimental results demonstrating real-time multi-contact control on two different hardware setups, including use of a Franka Emika Panda arm interacting with objects while frequently making and breaking contact.

In Chapter 5, we focus on utilizing tactile information for reactive control, which is very natural, yet underexplored in the robotics community. Towards this goal, we present an optimization-based numerical approach for designing control policies that use feedback on the contact forces (Aydinoglu et al., 2020, 2021b). Using LCS approximations, the proposed framework can synthesize control policies utilizing state and force feedback. The policy is provably stabilizing even during contact mode transitions for systems with possibly non-unique solutions. To achieve this, we choose a structure for controller and Lyapunov function designed specifically to leverage the complementarity structure of contact. The controller structure is non-combinatorial in nature and avoids enumerating the exponential number of potential hybrid modes that might arise from contact. More precisely, the contributions of each contact are additive, rather than combinatorial. Inspired by both prior work (Posa et al., 2015) and (Camlibel et al., 2001), we synthesize and verify a corresponding non-smooth, piecewise quadratic Lyapunov function. Additionally, we are able to explicitly define sparsity patterns allowing us to design controllers for systems where the full state information might be lacking, such as when the state of an object is unknown but tactile information is available. We also present a polynomial optimization program (Section 5.4, (5.21)) that can describe the non-unique solution sets of linear complementarity problems which is an important sub-step for tackling frictional contact problems. We demonstrate our approach on multiple numerical examples, including quasi-static friction problems and a high dimensional problem with ten contacts. Lastly, we validate our results on an experimental setup and show the effectiveness of the proposed method on an underactuated multi-contact system.

In Chapter 6, inspired by the connection between ReLU functions and linear complementarity problems, we develop a method to analyze linear complementarity systems in feedback with ReLU network controllers (Aydinoglu et al., 2021a). Our starting point is to show that we can represent ReLU neural networks as linear complementarity problems (Lemma 22). Next, we demonstrate that linear complementarity systems with neural network controllers have an equivalent LCS representation. Then, we leverage the theory of stability analysis for complementarity systems and derive the discrete time version of the results in (Camlibel et al., 2007). We describe the sufficient conditions for stability in the form of Linear Matrix Inequalities (LMI's). Denoting by N the number of neurons in the network plus the number of complementarity variables in the LCS, the size of the LMI's scales linearly with N. Furthermore, the maximum possible number of decision variables in our LMI scales quadratically with N.

We conclude in Chapter 7 with a summary and a discussion about the future challenges.

CHAPTER 2

RELATED WORK

2.1. Multi-Contact Planning

Multi-contact robotics has drawn growing interest over the last decade (Wensing et al., 2022), (Suomalainen et al., 2022). Early methods for both planning and control were focused on known contact sequences and hierarchical planners (Park et al., 2007; Abe et al., 2007). Over the course of the last decade, contact-implicit planners were developed (originally for offline motion synthesis (Posa et al., 2014)) and this body of work is growing rapidly (Önol et al., 2019), (Manchester and Kuindersma, 2020). These were first paired with controllers designed to track the planned mode (and limited to reasoning about that specific mode sequence) (Mastalli et al., 2020; Sleiman et al., 2021).

Other approaches achieved real-time hybrid planning via utilizing task-specific model simplifications. Some of these methods specialize in legged systems (Bledt, 2020; Kuindersma et al., 2016; Grandia et al., 2022) and focus on simplified dynamics models tailored for locomotion tasks, often times combined with gait-scheduling heuristics. However, these methods rely heavily on application-specific model simplifications and can not be applied across a wide range of robotics systems. Alternative methods rely on an offline phase to reduce the search space. This can lead to efficient real-time planning, but it requires a lengthy offline phase for each problem instance (Marcucci et al., 2017; Hogan and Rodriguez, 2020; Cauligi et al., 2021; Zhu and Righetti, 2022). Others approximate the dynamics via smooth contact models and modify the dynamics (e.g. diagonal approximation of Delassus operator) to enable high performance (Tassa et al., 2012; Koenemann et al., 2015; Kumar et al., 2014). While such methods can perform well in simulation, they have not been validated on hardware performing dynamic tasks that require constantly making and breaking contact.

Building upon this growing body of literature, this thesis presents an approach to real-time generic multi-contact MPC. In the preliminary version of our work (Aydinoglu and Posa, 2022), and in the

related work (Cleac'h et al., 2023), there has been very recent progress in developing fast contact-implicit MPC algorithms. While contact-implicit trajectory optimization approaches use global models and can take minutes to solve (Posa et al., 2014; Önol et al., 2019), these novel approaches focus on local hybrid models to significantly reduce the computational cost of optimal control. Le Cleac'h/Howell et al. (Cleac'h et al., 2023) use a softened contact model in combination with a custom primal-dual interior-point solver to achieve real-time rates. They focus on tracking a pre-computed trajectory, which provides a nominal mode sequence to track (although their algorithm can adapt this sequence online). In contrast, our approach (Section 4) is demonstrated to be fully capable of mode sequence synthesis without the need for a nominal trajectory or any other offline computations.

2.2. Adaptive MPC

Adaptive control focuses on real-time control of uncertain dynamical systems through adaptation and learning (Annaswamy and Fradkov, 2021) and optimal control in this setting has been an ongoing research direction for many years (Becker et al., 1985). Researchers have developed robust (adaptive) MPC methods (Tanaskovic et al., 2014; Fukushima et al., 2007; Weiss and Di Cairano, 2014; Desaraju et al., 2017). Also, there are multiple works that focus on applications such as electric vehicles (Wang et al., 2018), climate control (Ma et al., 2012), wind turbine control (Kou et al., 2016) or have combined adaptive control with MPC for applications on quadrotors (Pereida and Schoellig, 2018). Similarly, there are multiple recent methods that perform adaptive MPC under unknown noise distributions (Stamouli et al., 2022) or incorporate components from adaptive control into learning-based MPC (Chee et al., 2023). (Guzman et al., 2022) presents an adaptive MPC variant that automatically estimates control and model parameters by leveraging ideas from Bayesian optimization performing manipulation tasks. Unlike our approach that focuses on real-time adaptive MPC for systems that make and break contact, the authors focus on reaching tasks avoiding any contact interaction.

2.3. Tactile Feedback

The need for contact-aware control is driven, in part, by recent advances in tactile sensing (e.g (Taylor et al., 2022; Guggenheim et al., 2017; Yuan et al., 2015; Kumar et al., 2016; Donlon et al., 2018) and others). Given these advances, there has been ongoing research to design control policies using tactile feedback for tasks that require making and breaking contact. However, these approaches are largely based on static assumptions, for instance with guarded moves (Howe, 1993), or rely upon switching controllers (e.g. (Romano et al., 2011; Yamaguchi and Atkeson, 2016)). Other recent methods incorporate tactile sensors within learning frameworks, though they offer no guarantees on stability (Merzic et al., 2018; Tian et al., 2019; Nadeau et al., 2020; Sunil et al., 2023; Oller et al., 2023). By contrast to these methods, this thesis introduces an optimization-based numerical approach for designing control policies that use feedback on the contact forces with provable guarantees.

2.4. Analysis of Dynamical Systems with Neural Network Controllers

The connection between nonlinearities in neural networks and mathematical optimization has been exploited recently in various contexts. In (Raghunathan et al., 2018; Fazlyab et al., 2019b,a) the authors use quadratic constraints to describe ReLU activation functions followed by a semidefinite relaxation to perform robustness analysis of ReLU networks. In (Fazlyab et al., 2019c), the authors exploit the fact that all commonly used activation functions in deep learning are gradients of convex potentials, hence they satisfy incremental quadratic constraints that can be used to bound the global Lipschitz constant of feed-forward neural networks. The works (Dutta et al., 2019; Fan et al., 2020; Tran et al., 2020) perform reachability analysis for closed-loop systems with neural network controllers and (Sidrane and Kochenderfer, 2019) considers safety verification for such systems. The work in (Amos and Kolter, 2017) integrates quadratic programs as end-to-end trainable deep networks to encode constraints and more complex dependencies between the hidden states. Yin et al. (Yin et al., 2020) considers uncertain linear time-invariant systems with neural network controllers. By over approximating the input-output map of the neural network and uncertainties by quadratic and integral quadratic constraints, respectively, the authors develop an SDP whose solution yields

quadratic Lyapunov functions. In (Chen et al., 2020) the authors develop a learning-based iterative sample guided strategy based on the analytic center cutting plane method to search for Lyapunov functions for piece-wise affine systems in feedback with ReLU networks where the generation of samples relies on solving mixed-integer quadratic programs. In (Karg and Lucia, 2020), the authors use a mixed-integer linear programming formulation to perform output range analysis of ReLU neural networks and provide guarantees for constraint satisfaction and asymptotic stability of the closed-loop system. Inspired by the previous work, we propose a method to analyze multi-contact systems in feedback with ReLU network controllers.

CHAPTER 3

BACKGROUND

3.1. Mathematical Definitions

We first introduce the some notation and definitions used throughout this thesis. For a positive integer l, $\bar{l}$ denotes the set $\{1, 2, \ldots, l\}$. Given a matrix $M \in \mathbb{R}^{k \times l}$ and two subsets $I \subseteq \bar{k}$ and $J \subseteq \bar{l}$, we define $M_{IJ} = (m_{ij})_{i \in I, j \in J}$. For the case where $J = \bar{l}$, we use the shorthand notation $M_{I\bullet}$. For two vectors $a \in \mathbb{R}^m$ and $b \in \mathbb{R}^m$, the notation $0 \leq a \perp b \geq 0$ is used to denote that $a \geq 0,\ b \geq 0,\ a^T b = 0$. The collection of all absolutely continuous functions on a closed interval $[\alpha, \beta]$ is denoted as $AC([\alpha, \beta])$. The indeterminates are denoted with bold vectors, e.g., $\boldsymbol{x}$. The function $\mathcal{I}_{\mathcal{A}}$ is the $0 - \infty$ indicator function for an arbitrary set $\mathcal{A}$ such that $\mathcal{I}_{\mathcal{A}}(z) = 0$ if $z \in \mathcal{A}$ and $\mathcal{I}_{\mathcal{A}}(z) = \infty$ if $z \notin \mathcal{A}$. $\mathbb{N}_0$ denotes natural numbers with zero. For a positive semi-definite matrix Q, $Q^{1/2}$ denotes a matrix P such that $PP = P^T P = Q$. The notation $\mathbf{blkdiag}(A_0, \ldots, A_N)$ denotes a block diagonal matrix with entries in the given order (from top left (A_0) to bottom right (A_N)). The notation $\mathbf{diag}(a_0, \ldots, a_N)$ is used if entries a_i are scalars.

3.2. Linear Complementarity Problem

We utilize complementarity problems to represent the contact forces (detailed explanation in Section 3.4). The theory of linear complementarity problems (LCP) is well established (Cottle et al., 2009).

Definition 1. *Given a vector $q \in \mathbb{R}^m$, and a matrix $F \in \mathbb{R}^{m \times m}$, the $LCP(q, F)$ describes the following mathematical program:*

$$\begin{aligned} &\textit{find} && \lambda \in \mathbb{R}^m \\ &\textit{subject to} && y = F\lambda + q, \\ & && 0 \leq \lambda \perp y \geq 0. \end{aligned}$$

Here, the vector λ typically represents the contact forces and slack variables (Stewart and Trinkle,

2000), y represents the gap function and orthogonality constraint embeds the hybrid structure.

The solution set of the $LCP(q, F)$ is denoted by

$$\text{SOL}(q, F) = \{\lambda : y = F\lambda + q, 0 \leq \lambda \perp y \geq 0\}.$$

The LCP(q, F) may have multiple solutions or none at all. The cardinality of the solution set SOL(q, F) depends on the matrix F and the vector q. In particular, if F is a P-matrix, SOL(q, F) is always a singleton.

Definition 2. *A matrix $F \in \mathbb{R}^{m\times m}$ is a P-matrix, if the determinant of all of its principal submatrices are positive; that is, $det(F_{\alpha\alpha}) > 0$ for all $\alpha \subseteq \{1, \ldots, m\}$.*

Theorem 3. *(Cottle et al., 2009) The solution set SOL(q, F) is a singleton for all q if F is a P-matrix.*

If we denote the unique element of SOL(q, F) as $\lambda(q)$, then $\lambda(q)$ is a piece-wise linear function of q. We can describe this function explicitly as in (Camlibel et al., 2007). Consider $y = F\lambda(q) + q$, and define the index sets

$$\alpha(q) = \{i : \lambda_i(q) > 0 = y_i\},$$

$$\beta(q) = \{i : \lambda_i(q) = 0 \leq y_i\},$$

Then, $\lambda(q)$ is equivalent to

$$\lambda_\alpha(q) = -(F_{\alpha\alpha})^{-1} I_{\alpha\bullet} q, \;\; \lambda_\beta(q) = 0, \tag{3.1}$$

where $\alpha = \alpha(q)$ and $\beta = \beta(q)$. Furthermore, $\lambda(q)$ as in (3.1) is Lipschitz continuous since it is a continuous piece-wise linear function of q (Scholtes, 2012).

3.3. Linear Complementarity System

We use linear complementarity systems (LCS) as local representations of multi-contact systems (see Chapters 4, 5, 6). An LCS (Heemels et al., 2000) is a differential/difference equation coupled with a variable that is the solution of an LCP. Next, we describe the discrete-time and continuous-time

LCS formulations. In this thesis, we overload the term LCS and refer to both discrete-time and continuous-time formulations as LCS as it is clear from the context which one we are referring.

3.3.1. Discrete-Time LCS

Definition 4. *A discrete-time linear complementarity system describes the trajectories* $(x_k)_{k\in\mathbb{N}_0}$ *and* $(\lambda_k)_{k\in\mathbb{N}_0}$ *for an input sequence* $(u_k)_{k\in\mathbb{N}_0}$ *starting from* x_0 *such that*

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + D\lambda_k + d, \\ 0 \leq \lambda_k &\perp Ex_k + F\lambda_k + Hu_k + c \geq 0, \end{aligned} \tag{3.2}$$

where $x_k \in \mathbb{R}^{n_x}$, $\lambda_k \in \mathbb{R}^{n_\lambda}$, $u_k \in \mathbb{R}^{n_u}$, $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $D \in \mathbb{R}^{n_x \times n_\lambda}$, $d \in \mathbb{R}^{n_x}$, $E \in \mathbb{R}^{n_\lambda \times n_x}$, $F \in \mathbb{R}^{n_\lambda \times n_\lambda}$, $H \in \mathbb{R}^{n_\lambda \times n_u}$ *and* $c \in \mathbb{R}^{n_\lambda}$. *Equation* (3.2) *is often called as generalized LCS (Pang and Stewart, 2008) or inhomogeneous LCS (Camlibel et al., 2007) due to existence of the vectors* c *and* d*, but we use LCS for brevity.*

Vector x_k represents the state and often consists of the generalized positions q_k and velocities v_k. For a given k, x_k and u_k, the corresponding complementarity variable λ_k can be found by solving $\mathrm{LCP}(Ex_k + Hu_k + c, F)$ (see Definition 1). Similarly, x_{k+1} can be computed using the first equation in (3.2) when x_k, u_k and λ_k are known.

3.3.2. Continuous-Time LCS

Definition 5. *A continuous-time linear complementarity system describes the evolution of two time-dependent trajectories* $\bar{x}(t) \in \mathbb{R}^{n_x}$ *and* $\lambda(t) \in \mathbb{R}^m$ *for a given* $u(t) \in \mathbb{R}^{n_k}$ *and* $\bar{x}(0)$ *such that*

$$\begin{aligned} \dot{\bar{x}} &= \bar{A}\bar{x} + Bu + \bar{D}\lambda + a, \\ 0 \leq \lambda &\perp \bar{E}\bar{x} + \bar{F}\lambda + Hu + c \geq 0, \end{aligned} \tag{3.3}$$

where $\bar{A}$ *determines the autonomous dynamics of the state vector* $\bar{x}$, B *models the effect of the input on the state,* $\bar{D}$ *describes the effect of the contact forces on the state and* a *models the constant forces acting on the state.*

Similar to the discrete-time case, the matrices $\bar{E}, \bar{F}, H$ and the vector c capture the relationship

between the contact force λ, the state vector $\bar{x}$ and the input u. Note that the contact forces λ are always non-negative which holds for basic model of normal force and slack variables are typically used to represent sign-indefinite frictional forces. (3.3) implies that either $\lambda = 0$ or $\bar{E}\bar{x}+\bar{F}\lambda+Hu+c = 0$, encoding the multi-modal dynamics of contact. Due to this complementarity structure, an LCS is a compact representation, as the variables and constraints scale linearly with m, rather than with the potential 2^m hybrid modes (Camlibel et al., 2001), (Lin and Antsaklis, 2009).

3.4. Multi-Contact Dynamics

In Subsection 3.4.1, we introduce the continuous-time formulation of multi-contact dynamics with frictional contact (without impacts) as such models are explored in Chapter 5 for designing tactile feedback policies (for a through introduction, see (Halm, 2023; Brogliato, 2016; Stewart, 2000)).

In Subsection 3.4.2, we introduce the discrete-time formulations (Stewart and Trinkle, 2000; Anitescu, 2006) that are used in real-time MPC problems in Chapter 4.

3.4.1. Continuous-time Formulation Without Impacts

Multi-contact dynamics with frictional contact (without impacts) can be modeled by the manipulator equation:

$$M(q)\dot{v} + C(q, v) = Bu + J_n(q)^T\lambda_n + J_t(q)^T\lambda_t, \tag{3.4}$$

where q is the generalized positions vector, v is the generalized velocities, $M(q)$ is the inertia matrix, $C(q, v)$ represents the combined Coriolis, centrifugal and gravitational terms, B maps control inputs to joint coordinates, $J_n(q) = [J_{n,1}^T(q), \ldots, J_{n,n_c}^T(q)]^T$ and $J_t(q) = [J_{t,1}^T(q), \ldots, J_{t,n_c}^T(q)]^T$ are contact Jacobians for normal and tangential directions (where n_c is the number of pairs that can interact). λ_n represents the contact-frame normal forces and λ_t represents the contact-frame tangential (friction) forces which can be described by the following physical laws (Halm and Posa, 2023):

- Normal complementarity:

$$0 \leq \lambda_{n,i} \perp \phi_i(q) \geq 0, \tag{3.5}$$

  where ϕ_i represents the signed-distance distance between the ith contact pair. Equation (3.5)

represents that neither penetration nor force-at-a-distance are possible. If one is interested in spring-type soft contact models (as in Section 5), it is possible to swap (3.5) with

$$0 \leq \lambda_{n,i} \perp \phi_i(q) + k_i \lambda_{n,i} \geq 0, \tag{3.6}$$

where $\frac{1}{k_i} > 0$ represents the stiffness of the spring for the pair i. We note that the constraints (3.6) result in $\lambda_{n,i} = \max\{0, -\frac{1}{k_i}\phi_i(q)\}$. If there is no penetration ($\phi_i(q) \geq 0$), then normal contact force is zero, otherwise it is equal to the penetration depth scaled by $\frac{1}{k_i}$.

- Maximal Dissipation: For each contact i, friction dissipates as much power ($J_{t,i}v \cdot \lambda_{t,i}$) as possible and Coulomb friction with coefficient μ_i lies in the admissible set $\{\lambda_{t,i} : ||\lambda_{t,i}|| \leq \mu_i \lambda_{n,i}\}$. The corresponding set of generalized forces are called the friction cone:

$$FC(q) = \sum_{i \in \{j:\phi_j(q) \leq 0\}} \{J_{n,i}^T(q)\lambda_{n,i} + J_{t,i}^T(q)\lambda_{t,i} : ||\lambda_{t,i}||_2 \leq \mu_i \lambda_{n,i}\}.$$

A common variant that we focus on this thesis is the linearized Coulomb model (Stewart and Trinkle, 2000), where the admissible set is replaced with $\{\lambda_{t,i} \in \mu_i \lambda_{n,i} \text{conv}(\{d_1, \ldots, d_{n_e}\})\}$ for some unit-length vectors $D = [d_1, \ldots, d_{n_e}]$. This leads to a linearized friction cone $LFC(q)$ such that $LFC(q) \subseteq FC(q)$ (Halm, 2023). Using the linearized friction cone, one can describe the frictional forces, λ_D, with the following complementarity relations (Halm and Posa, 2018):

$$\begin{aligned} 0 \leq \lambda_{D,i} \perp J_{D,i}(q)v + \mathbf{1}\gamma_i \geq 0, \\ 0 \leq \gamma_i \perp \mu_i \lambda_{n,i} - \mathbf{1}^T \lambda_{D,i} \geq 0, \end{aligned} \tag{3.7}$$

where $\mathbf{1}$ is a vector of ones, γ_i is a slack variable that corresponds to contact pair i and $J_{D,i}(q) = D^T J_{t,i}(q)$. Since $\lambda_{t,i} = D\lambda_{D,i}$, it follows that (3.4) is equivalent to:

$$M(q)\dot{v} + C(q,v) = Bu + \sum_i J_{n,i}^T(q)\lambda_{n,i} + J_{D,i}^T(q)\lambda_{D,i}$$

where $\lambda_{n,i}$ and $\lambda_{D,i}$ are defined as in complementarity equations (3.5), (3.6), (3.7).

As discussed, we represent multi-contact dynamics with frictional contact using (3.4) with the constraints (3.5), (3.6), (3.7). Such constrained dynamical system models are equivalent to differential inclusions (Smirnov, 2002). We discuss more about stability and solutions of differential inclusions in Chapter 5.

3.4.2. Discrete-time Formulation

Next, we define two discrete-time semi-implicit schemes that are used in Sections 4 and 6. For the discrete-time formulations, we use Δt to describe the discretization time-step, n_c is the number of rigid body pairs that can interact, n_e represents the number of edges of the polyhedral approximation of the friction cone (as discussed in Section 3.4.1), and J_n, J_t are contact Jacobians for normal and tangential directions. In addition, $E_t = \mathbf{blkdiag}(e, \ldots, e)$ with $e = [1, \ldots, 1] \in \mathbb{R}^{1 \times n_e}$ and ϕ represents the distance between rigid body pairs.

Stewart and Trinkle Formulation (Stewart and Trinkle, 2000)

Initially, we will explore Stewart and Trinkle formulation due to its intuitive, simple form. In almost all the examples in Chapter 4 (and partially in Chapter 6), we use this semi-implicit time-stepping scheme:

$$\begin{aligned} q_{k+1} &= q_k + \Delta t v_{k+1}, \\ v_{k+1} &= v_k + \Delta t M^{-1}(q_k)\Big(Bu_k - C(q_k, v_k) + J_n(q_k)^T \lambda_k^n + J_t(q_k)^T \lambda_k^t \Big). \end{aligned} \tag{3.8}$$

The forces λ_k^n and λ_k^t can be described via the following complementarity problem:

$$\begin{aligned} &0 \leq \gamma_k \perp \mu \lambda_k^n - E_t \lambda_k^t \geq 0, \\ &0 \leq \lambda_k^n \perp \phi(q_k) + J_n(q_k)(q_{k+1} - q_k) \geq 0, \\ &0 \leq \lambda_k^t \perp E_t^T \gamma_k + J_t(q_k) v_{k+1} \geq 0, \end{aligned} \tag{3.9}$$

where γ_k is a slack variable, $\mu = \mathbf{diag}(\mu_1, \ldots, \mu_{n_c})$ represents the coefficient of friction between rigid body pairs.

Anitescu Formulation (Anitescu, 2006)

In Section 4.6, we use a learning algorithm (Jin et al., 2022) that relies on convexity of the contact model (i.e. $F = F(x_k)$ as in (3.2) is positive semi-definite for any x_k) but Stewart and Trinkle formulation does not satisfy this assumption. Hence, we focus on the Anitescu formulation since it is a convex contact model:

$$\begin{aligned} q_{k+1} &= q_k + \Delta t v_{k+1}, \\ v_{k+1} &= v_k + M^{-1}(q_k)\Big(\Delta t B u_k - \Delta t C(q_k, v_k) + J_c(q_k)^T \lambda_k\Big), \end{aligned} \tag{3.10}$$

where the contact Jacobian is defined as $J_c = E_t^T J_n(q_k) + \mu J_t(q_k)$. Via the complementarity equations, λ_k is described as:

$$0 \leq \lambda_k \perp \frac{E_t^T \phi(q_k)}{\Delta t} + \frac{1}{\Delta t} E_t^T J_n(q_k)(q_{k+1} - q_k) + \mu J_t(q_k) v_{k+1} \geq 0. \tag{3.11}$$

3.5. Local Hybrid Approximations of Multi-Contact Dynamics

In this section, we discuss about the local hybrid approximations of multi-contact dynamics. First, we underline that all of the nonlinear hybrid models (both continuous-time and discrete-time) introduced in Section 3.4 can be described with elements $(M, C, B, J_n, J_t, \phi, \mu)$. For compactness (with a slight abuse of notation), we denote a nonlinear multi-contact model as $\mathcal{M}$ where $\mathcal{M} = \{M, C, B, J, \phi, \mu\}$.

Models $\mathcal{M}$ as in Section 3.4 has been employed for trajectory optimization of multi-contact robotics (Posa et al., 2014), though such methods have been limited to offline motion planning due to the inherent complexity in the nonlinear model. To reduce the complexity of such models, it is possible to locally approximate $\mathcal{M}$ with a linear complementarity system (defined in Section 3.3). Even though an LCS approximation is slightly less complex than the nonlinear model, it still captures the multi-modal nature of the problem and enables one to make decisions such as making or breaking contact (as demonstrated in Sections 4.4 and 4.5). Hence we focus on LCS approximations as they lead to more tractable optimal control problems than their nonlinear counterparts but note that

these are still hybrid optimal control problems that are difficult to solve (Section 4.3).

Next, given a model $\mathcal{M}$, we will describe how to obtain an LCS approximation. Towards this direction, we first focus on the Stewart and Trinkle model from Section 3.4.2 and present an approach to obtain an LCS approximation given the complementarity system model (3.8)-(3.9). For a given state $(x^*)^T = [(q^*)^T, (v^*)^T]$ and input u^*, we approximate (3.8) as:

$$\begin{aligned} q_{k+1} &= q_k + \Delta t v_{k+1}, \\ v_{k+1} &= v_k + \Delta t(J_f \begin{bmatrix} q_k \\ v_k \\ u_k \end{bmatrix} + D_1\lambda_k^n + D_2\lambda_k^t + d_v). \end{aligned} \tag{3.12}$$

Here $J_f = J_f(q^*, v^*, u^*)$ is the Jacobian of $f(q, v, u) = M^{-1}(q)Bu - M^{-1}(q)C(q, v)$ evaluated at (q^*, v^*, u^*), $d_v = f(q^*, v^*, u^*) - J_f \begin{bmatrix} (q^*)^T & (v^*)^T & (u^*)^T \end{bmatrix}^T$ is a constant vector, $D_1 = M^{-1}(q^*)J_n(q^*)^T$ and $D_2 = M(q^*)^{-1}J_t(q^*)^T$ represent the effect of contact forces. Similarly, we approximate (3.9) with the following LCP:

$$\begin{aligned} &0 \leq \gamma_k \perp \mu\lambda_k^n - E_t\lambda_k^t \geq 0, \\ &0 \leq \lambda_k^n \perp \phi(q^*) + J_n(q^*)q_k + \Delta t J_n(q^*)v_{k+1} - J_n(q^*)q^* \geq 0, \\ &0 \leq \lambda_k^t \perp E_t^T\gamma_k + J_t(q^*)v_{k+1} \geq 0. \end{aligned} \tag{3.13}$$

We underline that the second equation in (3.13) is equivalent to $\phi(q^*) + J_n(q^*)(q_{k+1} - q^*)$. Furthermore this holds for any k as it follows from the Taylor expansion of $\phi(q)$ around q^* where $\phi(q) \approx \phi(q^*) + J_n(q^*)(q - q^*)$. Observe that (3.12) and (3.13) can be written in the LCS form (3.2) where $x_k^T = [q_k^T, v_k^T]$ and $\lambda_k^T = [\gamma_k^T, (\lambda_k^n)^T, (\lambda_k^t)^T]$.

Similarly (for the Anitescu formulation), given state $(x^*)^T = [(q^*)^T, (v^*)^T]$ and input u^*, we can

approximate (3.10) as:

$$\begin{aligned} q_{k+1} &= q_k + \Delta t v_{k+1}, \\ v_{k+1} &= v_k + \Delta t J_f \begin{bmatrix} q_k \\ v_k \\ u_k \end{bmatrix} + D\lambda_k + d_v. \end{aligned} \tag{3.14}$$

where J_f, d_v are same as in (3.12) and $D = M^{-1}(q^*)J_c(q^*)^T$. The complementarity part (3.11) is:

$$0 \leq \lambda_k \perp \frac{1}{\Delta t} E_t^T \Big(\phi(q^*) + J_n(q^*)q_k - J_n(q^*)q^* \Big) + J_c(q^*)v_{k+1} \geq 0. \tag{3.15}$$

Similar to the LCS approximation of the Stewart and Trinkle formulation, (3.14) and (3.15) can be written in the LCS form (3.2) where $x_k^T = [q_k^T, v_k^T]$.

It is important to note that to obtain these LCS models, we approximate the differentiable terms (of the model $\mathcal{M}$), which arise form standard Lagrangian dynamics, by linearizing with respect to (q, v, u) about (q^*, v^*, u^*) but leave the multi-modal structure intact.

3.6. Sum-of-squares

In Chapter 5, describing the non-unique solution sets of LCP's is posed as a question of non-negativity of polynomials on basic semialgebraic sets. Towards this direction, sum-of-squares (SOS) optimization is used.

A multivariate polynomial $p(x)$ is a sum-of-squares (SOS) if there exist polynomials $q_i(x)$ such that

$$p(x) = \sum_i q_i^2(x).$$

The existence of a sum-of-squares decomposition of a polynomial can be decided by solving a semidefinite programming feasibility problem (Parrilo, 2003), which is a convex optimization problem. We represent the semialgebraic conditions using the S-procedure technique (Stengle, 1974),

(Boyd et al., 1994). For example, to show that (Vandenberghe and Boyd, 1996)

$$f(x) \geq 0, \quad \forall x \in \{z : g(z) \geq 0, h(z) = 0\},$$

it is sufficient to find polynomials $\sigma_1(x), \sigma_2(x), q(x)$ s.t.

$$\begin{aligned} &\sigma_1(x)f(x) - \sigma_2(x)g(x) - q(x)h(x) \geq 0, \\ &\sigma_1(x) - 1 \geq 0, \\ &\sigma_2(x) \geq 0. \end{aligned} \tag{3.16}$$

If constraints are in the form of (3.16) and the objective function is linear in the coefficients of any unknown/free polynomials, then the optimization problem can be represented as a semidefinite program (SDP) using the SOS relaxation.

CHAPTER 4

REAL-TIME MULTI-CONTACT MODEL PREDICTIVE CONTROL VIA ADMM

Parts of this chapter were previously published as parts of Alp Aydinoglu and Michael Posa. Real-Time Multi-Contact Model Predictive Control via ADMM. In 2022 International Conference on Robotics and Automation (ICRA), pages 3414–3421, ©2022 IEEE.

4.1. Introduction

Focusing on strategically approximating the multi-contact dynamics via LCS approximations, we propose a hybrid model predictive control algorithm, consensus complementarity control (C3), for systems that make and break contact with their environment. Many state-of-the-art controllers for tasks which require initiating contact with the environment, such as locomotion and manipulation, require *a priori* mode schedules or are too computationally complex to run at real-time rates. We present a method based on the alternating direction method of multipliers (ADMM) that is capable of high-speed reasoning over potential contact events. Via a consensus formulation, our approach enables parallelization of the contact scheduling problem. Furthermore, we also combine C3 with a residual learner to accomplish tasks where good estimates of model parameters are not available. We validate our results on six numerical examples, including five high-dimensional frictional contact problems, and a physical experimentation on an underactuated multi-contact system. We further demonstrate the effectiveness of our method on a physical experiment accomplishing a high-dimensional, multi-contact manipulation task with a robot arm.

4.2. Problem Formulation

We are interested in solving optimal control problems with model $\mathcal{M}$ (as described in Section 3.4). We consider discrete-time formulations as in (Posa et al., 2014):

$$\min_{q_k, v_k, u_k, \lambda_k} \quad \sum_k g_c(q_k, v_k, u_k) \tag{4.1}$$

$$\text{s.t.} \quad (q_{k+1}, v_{k+1}, q_k, v_k, u_k, \lambda_k) \text{ respects } \mathcal{M} \tag{4.2}$$

where g_c is the cost function (often quadratic). In most of the examples in this paper, we transcribe (4.2) using the model described in Section 3.4.2. Furthermore, instead of solving the problem with the nonlinear model, we follow the procedure in Section 3.5 and obtain an LCS approximation.

Moving forward, we focus on LCS models of the form (3.2) as it is a general form. We do not include time-varying LCS (where matrices such as A in (3.2) depend on k) for ease of notation but note that the work in this chapter can easily be extended to the time-varying setting (and code we provide can deal with time-varying LCS). With a slight abuse of notation, we denote the models as $\mathcal{L}_{\Delta t}(x, u)$ where dependence on a given state x^* and input u^* (e.g. as in (3.12), (3.13)) is suppressed. Furthermore we use the notation $(x_{k+1}, \lambda_k) = \mathcal{L}_{\Delta t}(x_k, u_k)$ to denote the state of the LCS system after one time step (Δt time later). Here, λ_k is the solution of LCP$(Ex_k + Hu_k + c, F)$ and $x_{k+1} = Ax_k + Bu_k + D\lambda_k + d$.

4.3. Model Predictive Control of Multi-Contact Systems

As we consider LCS models, we focus on the following mathematical optimization problem:

$$\begin{aligned}
\min_{x_k, \lambda_k, u_k} \quad & \sum_{k=0}^{N-1} (x_k^T Q_k x_k + u_k^T R_k u_k) + x_N^T Q_N x_N \\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k + D\lambda_k + d, \\
& Ex_k + F\lambda_k + Hu_k + c \geq 0, \\
& \lambda_k \geq 0, \\
& \lambda_k^T (Ex_k + F\lambda_k + Hu_k + c) = 0, \\
& (\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}) \in \mathcal{C}, \\
& \text{for } k = 0, \ldots, N-1, \text{given } x_0,
\end{aligned} \tag{4.3}$$

where N is the planning horizon, Q_k, Q_N are positive semidefinite matrices, R_k are positive definite matrices and $\mathcal{C}$ is a convex set (e.g. input bounds, safety constraints, or goal conditions) and $\boldsymbol{x}^T = [x_1^T, \ldots, x_N^T]$, $\boldsymbol{\lambda}^T = [\lambda_0^T, \lambda_1^T, \ldots, \lambda_{N-1}^T]$, $\boldsymbol{u}^T = [u_0^T, u_1^T, \ldots, u_{N-1}^T]$.

Given x_0, one solves the optimization (4.3) and applies u_0 to the plant and repeats the process in

every time step in a receding horizon manner.

4.3.1. Mixed Integer Formulation

One straightforward, but computationally expensive, approach to solving (4.3) is via a mixed integer formulation which exchanges the non-convex orthogonality constraints for binary variables:

$$\begin{aligned}
\min_{x_k,\lambda_k,u_k,s_k} \quad & \sum_{k=0}^{N-1}(x_k^T Q_k x_k + u_k^T R_k u_k) + x_N^T Q_N x_N \\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k + D\lambda_k + d, \\
& Ms_k \geq Ex_k + F\lambda_k + Hu_k + c \geq 0, \\
& M(\mathbf{1} - s_k) \geq \lambda_k \geq 0, \\
& (\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}) \in \mathcal{C}, s_k \in \{0,1\}^{n_\lambda}, \\
& \text{for } k = 0, \ldots, N-1, \text{given } x_0,
\end{aligned} \tag{4.4}$$

where $\mathbf{1}$ is a vector of ones, M is a scalar used for the big M method and s_k are the binary variables. This approach requires solving 2^{Nn_λ} quadratic programs as a worst case. This method is popular because, in practice, it is much faster than this worst case analysis. Even still, for the multi-contact problems explored in this chapter, directly solving (4.4) via state-of-the-art commercial solvers remains too slow for real-time control. Methods that learn the MIQP problem offline are promising, but this line of work requires large-scale training on every problem instance (Cauligi et al., 2020), (Aydinoglu et al., 2021a).

4.3.2. Consensus Complementarity Control (C3)

Utilizing a method based on ADMM, we will solve (4.3) more quickly than with mixed integer formulations. Since the problem we are addressing is non-convex, our method is not guaranteed to find the global solution or converge unlike MIQP-based approaches, but is significantly faster. First, we rewrite (4.3), equivalently, in the consensus form (Park and Boyd, 2017) where we create copies (named δ_k) of variables $z_k^T = [x_k^T, \lambda_k^T, u_k^T]$ and move the constraints into the objective function using

$0-\infty$ indicator functions:

$$\begin{aligned} \min_{z} \quad & c(z) + \mathcal{I}_{\mathcal{D}}(z) + \mathcal{I}_{\mathcal{C}}(z) + \sum_{k=0}^{N-1} \mathcal{I}_{\mathcal{H}_k}(\delta_k) \\ \text{s.t.} \quad & z_k = \delta_k, \ \forall k, \end{aligned} \tag{4.5}$$

where $z^T = [z_0^T, z_1^T, \ldots, z_{N-1}^T]$. Note that $\delta^T = [\delta_0^T, \delta_1^T, \ldots, \delta_{N-1}^T]$ is a copy of z^T and equality constraints can also be written as $z = \delta$. $c(z)$ is the cost function[1] in (4.3). The set $\mathcal{D}$ includes the dynamics constraints:

$$\cap_{k=0}^{N-2}\{z : x_{k+1} = Ax_k + Bu_k + D\lambda_k + d\},$$

and the sets $\mathcal{H}_k$ represent the LCP (contact) constraints:

$$\begin{aligned} \mathcal{H}_k = \{(x_k, \lambda_k, u_k) : & Ex_k + F\lambda_k + Hu_k + c \geq 0, \\ & \lambda_k \geq 0, \lambda_k^T(Ex_k + F\lambda_k + Hu_k + c) = 0\}. \end{aligned}$$

Note that we leverage the time-dependent structure in the complementarity constraints to separate δ_k's so that each set $\mathcal{H}_k$ depends only on δ_k, and not variables corresponding to any other timestep.

The general augmented Lagrangian ((Boyd et al., 2011), Section 3.4.2.) for the problem in consensus form (4.5) is (Appendix for details):

$$\begin{aligned} \mathcal{L}_\rho(z, \delta, w) = & \, c(z) + \mathcal{I}_{\mathcal{D}}(z) + \mathcal{I}_{\mathcal{C}}(z) \\ & + \sum_{k=0}^{N-1} \big(\mathcal{I}_{\mathcal{H}_k}(\delta_k) + \rho(r_k^T G_k r_k - w_k^T G_k w_k)\big), \end{aligned} \tag{4.6}$$

where $\rho > 0$ is the penalty parameter, $w^T = [w_0^T, w_1^T, \ldots, w_{N-1}^T]$, w_k are scaled dual variables, $r_k = z_k - \delta_k + w_k$, G_k is a positive definite matrix. Observe that the standard augmented Lagrangian (Park and Boyd, 2017) is recovered for $G_k = I$ for all k.

[1]The cost function has the form $c(z) = \sum_{k=0}^{N-1}(x_k^T Q_k x_k + u_k^T R_k u_k) + ||Q_N^{1/2}(Ax_{N-1} + Bu_{N-1} + D\lambda_{N-1} + d)||_2^2$.

In order to solve (4.5), we apply the ADMM algorithm, consisting of the following operations (Appendix for details):

$$z^{i+1} = \text{argmin}_z \mathcal{L}_\rho(z, \delta^i, w^i), \tag{4.7}$$

$$\delta_k^{i+1} = \text{argmin}_{\delta_k} \mathcal{L}_\rho^k(z_k^{i+1}, \delta_k, w_k^i),\ \forall k, \tag{4.8}$$

$$w_k^{i+1} = w_k^i + z_k^{i+1} - \delta_k^{i+1},\ \forall k, \tag{4.9}$$

where $\mathcal{L}_\rho^k(z_k, \delta_k, w_k) = \mathcal{I}_{\mathcal{H}_k}(\delta_k) + \rho(r_k^T G_k r_k - w_k^T G_k w_k)$. Here, (4.7) requires solving a quadratic program, (4.8) is a projection onto the LCP constraints and (4.9) is a dual variable update. Next, we analyze these operations in the given order.

Quadratic Step

Equation (4.7) can be represented by the convex quadratic program

$$\begin{aligned} \min_z \quad & c(z) + \sum_{k=0}^{N-1} (z_k - \delta_k^i + w_k^i)^T \rho G_k (z_k - \delta_k^i + w_k^i) \\ \text{s.t.} \quad & z \in \mathcal{D} \cap \mathcal{C}. \end{aligned} \tag{4.10}$$

The linear dynamics constraints are captured by the set $\mathcal{D}$, and the convex inequality constraints on states, inputs, contact forces are captured by $\mathcal{C}$. The complementarity constraints do not explicitly appear, but their influence is found, iteratively, through the variables δ_k^i.

The QP in (4.10) can be solved quickly via off-the-shelf solvers and is analogous to solving the MPC problem for a linear system without contact.

Projection Step

This step requires projecting onto the LCP constraints $\mathcal{H}_k$ and is the most challenging part of the problem. (4.8) can be represented with the following quadratic program with a non-convex constraint:

$$\begin{aligned} \min_{\delta_k} \quad & (\delta_k - (z_k^{i+1} + w_k^i))^T \rho G_k (\delta_k - (z_k^{i+1} + w_k^i)) \\ \text{s.t.} \quad & \delta_k \in \mathcal{H}_k, \end{aligned} \tag{4.11}$$

where $\delta_k = [(\delta_k^x)^T, (\delta_k^\lambda)^T, (\delta_k^u)^T]$. In our setting, we will consider three different projections. Two are approximate projections, common for minimization problems over non-convex sets (Diamond et al., 2018).

MIQP Projection The projection can be calculated by exactly formulating (4.11) as a small-scale MIQP

$$\begin{aligned} \min_{\delta_k, s_k} \quad & (\delta_k - (z_k^{i+1} + w_k^i))^T U (\delta_k - (z_k^{i+1} + w_k^i)) \\ \text{s.t.} \quad & M s_k \geq E\delta_k^x + F\delta_k^\lambda + H\delta_k^u + c \geq 0, \\ & M(\mathbf{1} - s_k) \geq \delta_k^\lambda \geq 0, \\ & s_k \in \{0,1\}^{n_\lambda}, \end{aligned} \tag{4.12}$$

where U is a positive semi-definite matrix. For $U = \rho G_k$, one recovers the problem in (4.11); however, in our experience, we found significantly improved performance using alternate, but fixed, choices for U. Observe that while (4.12) is non-convex, it is written only in terms of variables corresponding to a single time step k. While the original MIQP formulation in (4.4) has Nn_λ binary variables, here we have N independent problems, each with n_λ binary variables. This decoupling leads to dramatically improved performance (worst-case $N2^{n_\lambda}$ vs 2^{Nn_λ}).

LCP Projection In cases where (4.12) cannot be solved quickly enough, we propose two approximate solutions with faster run-time. Consider the limiting case where U has no penalty on the force elements. Here, (4.12) can be solved with optimal objective value of 0 by setting $\delta_k^x = z_k^{(i+1),x} + w_k^{(i),x}$ and $\delta_k^u = z_k^{(i+1),u} + w_k^{(i),u}$. Then, δ_k^λ can be found by solving $\text{LCP}(E\delta_k^x + H\delta_k^u + c, F)$. We note that this projection is different than shooting based methods since we are simulating the $z_k^{(i+1),x,u} + w_k^{(i),x,u}$ instead of $z_k^{(i+1),x,u}$.

ADMM Projection We note that (4.12) has an equivalent quadratically constrained quadratic program (QCQP) representation, where prior work has solved problems of this form using ADMM ((Park and Boyd, 2017), Section 4.4). This approach relies on the fact that QCQPs with a single constraint are solvable in polynomial time via the bisection method ((Park and Boyd, 2017), Appendix B). Performing the projection step via this method leads to nested ADMM algorithms, but this formulation can produce faster solutions than MIQP solvers without guarantees that it produces

Algorithm 1 Consensus Complementarity Control (C3)

Require: $\theta = \{Q_k, R_k, Q_N, \delta_k^0, w_k^0, G_k, s, \rho, \rho_s, N\}$, $\mathcal{L}_{\Delta t}$, x_0

Initialization : $i = 1$

1: **while** $i \leq s$ **do**
2: Compute z^{i+1} via (4.10)
3: Compute δ_k^{i+1} via (4.12), $\forall k$
4: $w_k^{i+1} \leftarrow w_k^i + z_k^{i+1} - \delta_k^{i+1}, \forall k$
5: $\rho \leftarrow \rho_s \rho$
6: $w_k^{i+1} \leftarrow w_k^{i+1}/\rho_s, \forall k$
7: $i \leftarrow i + 1$
8: **end while**
9: **return** u_0 from z^{s+1}

a feasible or optimal value. Note that this formulation fared poorly for high dimensional projections or when applied directly to the original problem (4.3), rarely satisfying the complementarity constraints.

Solving the full MIQP (4.12) typically is the best at exploring a wide range of modes, while the LCP projection is usually the fastest (depending on matrix F). The ADMM projection can be viewed as a middle ground. We underline that both MIQP and LCP projections produce feasible solutions (for the projection step) whereas ADMM projection has no such guarantee. For frictional contact problems (Sections 4.4 and 4.5), we observed that the MIQP projection performed better than the others.

After discussing each of the individual steps, we present the full C3 algorithm (Algorithm 1). Here θ represents the C3 parameters. Both δ_k^0 and w_k^0 are usually initialized as zero vectors. G_k are positive definite matrices, $s > 0$ is the number of ADMM steps, $\rho_s > 0$ is the scaling parameter for ρ, and N is the number of planning steps. Furthermore $\mathcal{L}_{\Delta t}$ is the LCS model and x_0 is an initial state. Given this information, the algorithm returns u_0 as in standard model-predictive control frameworks.

4.4. Illustrative Examples

In this section, we demonstrate the effectiveness of C3 presenting multiple simulation results and an hardware experiment with an under-actuated multi-contact system. Concrete steps of method

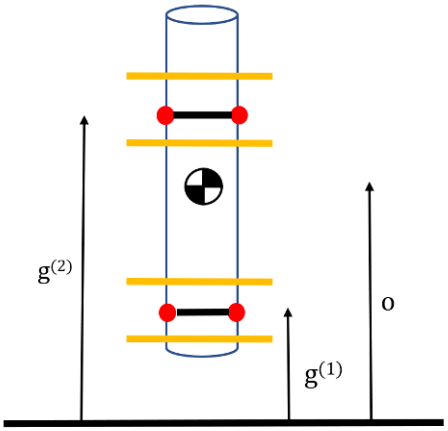

Figure 4.1: Lifting an object using two grippers indicated by red circles. The soft limits where the grippers should not cross are indicated by yellow lines.

Algorithm 2

Require: $\mathcal{M}, \theta, \Delta t, \hat{u}$

1: Estimate state as $\hat{x}$
2: Obtain LCS model $\mathcal{L}_{\Delta t}$ around $(\hat{x}, \hat{u})$ using $\mathcal{M}$
3: Run Algorithm 1 with $\mathcal{L}_{\Delta t}$, θ, $\hat{x}$ and obtain u_0

used throughout this section is shown in Algorithm 2. Given a multi-contact model $\mathcal{M}$, we obtain an LCS model $\mathcal{L}_{\Delta t}$ around the estimated state $\hat{x}$ and nominal input $\hat{u}$. Then, we run Algorithm 1 to compute u_0 and that is directly applied to the system.

For these results, OSQP (Stellato et al., 2020) is used to solve quadratic programs and Gurobi (Gurobi Optimization, LLC, 2021) is used for mixed integer programs. PATH (Dirkse and Ferris, 1995) and Lemke's algorihm have been used to solve LCP's. SI units (meter, kilogram, second) are used. The experiments are done on a desktop computer with the processor Intel *i7-11800H* and *16GB RAM*. Reported run-times include all steps in the algorithm. The code for all examples is available[2]. Experiments are also shown in the supplementary video[3].

4.4.1. Finger Gaiting

In this subsection, we want to show that our algorithm is capable of mode exploration. Our goal is to lift a rigid object upwards using four fingers. The setup for this problem is illustrated in Figure

[2]https://github.com/AlpAydinoglu/coptimal
[3]https://youtu.be/XBzlyNhKl8w

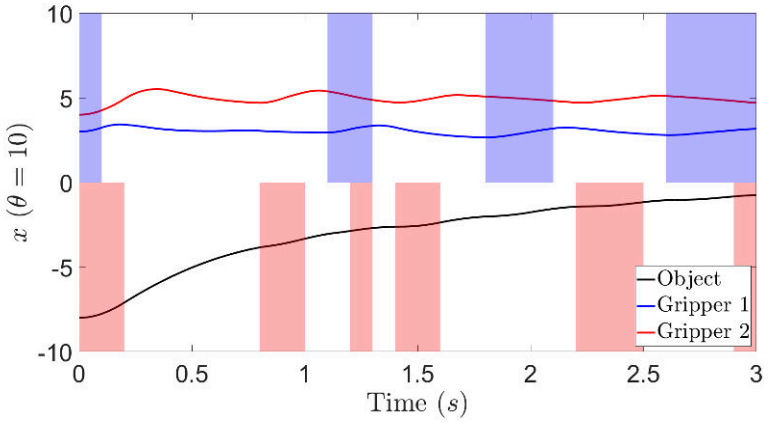

Figure 4.2: Finger gaiting with $s = 10$. Blue shading implies that gripper 1 is applying normal force to the object whereas red shading implies that gripper 2 is applying normal force.

4.1. The red circles indicate where the grippers interact the object and we assume that the grippers are always near the surface of the object and the force they apply on the object can be controlled. This force affects the friction between the object and grippers. Since the grippers never leave the surface, we assume that there is no rotation. The goal of this task is to lift the object vertically, while the fingers are constrained to stay close to their original locations (soft constraints shown in yellow). This task, therefore, requires finger gaiting to achieve large vertical motion of the object.

We use the formulation in (Stewart and Trinkle, 2000) for modeling the system and denote the positions of the grippers as $g^{(1)}$, $g^{(2)}$ respectively and position of the object as o. We choose $g = 9.81$ as gravitational acceleration and $\mu = 1$ is the coefficient of friction for both grippers.

We design a controller based on Algorithm 1 where $G_k = I$, $N = 10$. The controller uses the MIQP projection method since other two projection methods failed to find stable motions. For this example, we use $s = 10$ and $\rho_s = 1.2$. Parallelization leads to ≈ 2.5x speedup for this particular example and we are able to run at 30.3 Hz. We also enforce limits:

$$1 \leq g^{(1)} \leq 3,\ \forall k,$$

$$3 \leq g^{(2)} \leq 5,\ \forall k.$$

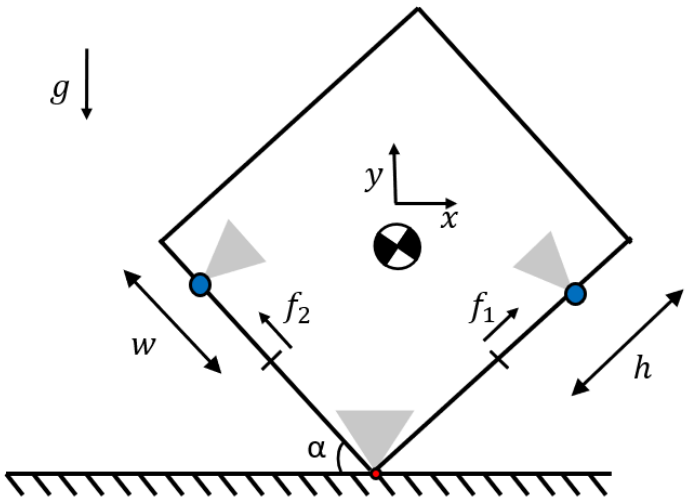

Figure 4.3: Pivoting a rigid object with two fingers (blue). The object can make and break contact with the ground and gray areas represent the friction cones.

We performed 100 trials starting from different initial conditions where $o(0) \sim U[-6, -8]$, $g^{(1)}(0) \sim U[2, 3]$, $g^{(2)}(0) \sim U[3, 4]$ are uniformly distributed and both the grippers and the object have zero initial speed. The controller managed to lift the object in all cases. We present a specific example in Figure 4.2 where the grippers first throw the object into the air and then catch it followed by some finger gaiting.

4.4.2. Pivoting

Here, we want to demonstrate that our algorithm is capable of making decisions about stick-slip transitions as well as making/breaking contact. We consider pivoting a rigid object that can make and break contact with the ground inspired by Hogan et. al (Hogan et al., 2020). Two fingers (indicated via blue) interact with the object as in Figure 4.3. The goal is to balance the rigid-object at the midpoint.

The positions of the fingers with respect to the object are described via f_1, f_2 respectively. The normal force that the fingers exert onto the box can be controlled. The center of mass position is denoted by x and y respectively, α denotes the angle with the ground and $w = 1$, $h = 1$ are the dimensions of the object. The coefficient of friction for the fingers are $\mu_1 = \mu_2 = 0.1$, and the

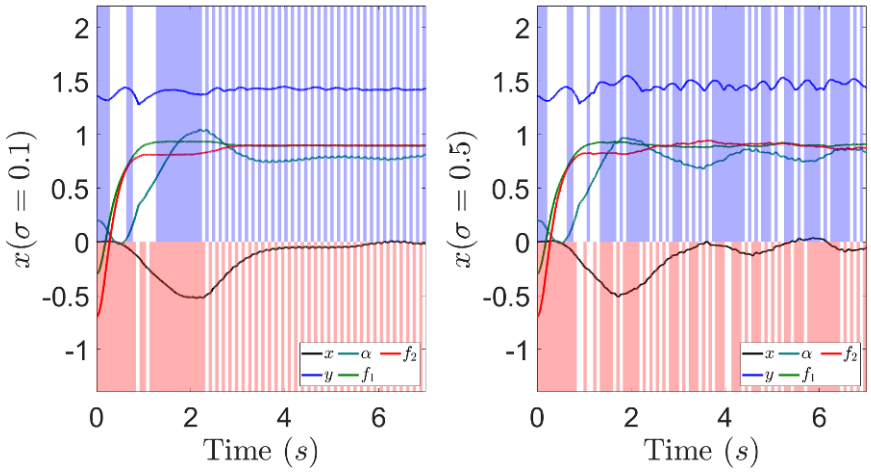

Figure 4.4: Pivoting example, Gaussian disturbances with standard deviations σ. Blue shading implies that gripper 1 is applying normal force to the object whereas red shading implies that gripper 2 is applying normal force.

coefficient of friction with the ground is $\mu_3 = 1$. We take the gravitational acceleration as $g = 9.81$ and mass of the object as $m = 1$. We model the system using an implicit time-stepping scheme (Stewart and Trinkle, 2000). The system has 3 contacts, where the finger contacts are represented by 3 complementarity variables each, and the ground contact is modeled via 4 complementarity variables. The system has ten states ($n_x = 10$), ten complementarity variables ($n_\lambda = 10$) and 4 inputs ($n_u = 4$). More concretely, it has 36 hybrid modes.

For this example, as the LCS-representation is only an approximation, we compute a new local LCS approximation at every time step k. We pick $s = 5$, $N = 10$, $\rho_s = 1.1$ and use the local LCS approximation given at time-step k while planning. Parallelization leads to $\approx$ 4x speedup for this particular example and controller can run around 43.4 Hz.

We present an example where the fingers start close to the pivot point where $f_1 = -0.3$, $f_2 = -0.7$ and the objects configuration is given by $x = 0$, $y = 1.36$ and $\alpha = 0.2$. The goal is to balance the object at the midpoint ($x = 0$, $y = \sqrt{2}$, $\alpha = \pi/4$) while simultaneously moving the fingers towards the end of the object ($f_1 = f_2 = 0.9$). Figure 4.4 demonstrates the robustness of the controller for different Gaussian disturbances (added to dynamics) with standard deviations (for $\sigma = 0.1, 0.5$).

Table 4.1: Projection Run-time and averaged cost-to-go

Projection Method	**Mean** $\pm$ **Std** (s)	**Cost**
LCP	$1.4 \cdot 10^{-5} \pm 1.8 \cdot 10^{-6}$	22.09
MIQP	$1 \cdot 10^{-3} \pm 1.2 \cdot 10^{-4}$	24.98
ADMM	$5.3 \cdot 10^{-4} \pm 3.3 \cdot 10^{-5}$	35.74

Note that at every time step, all positions and velocities (including angular) are affected by the process noise. The object approximately reaches the desired configuration (midpoint where $\alpha = \frac{\pi}{4}$) for $\sigma = 0, 0.05, 0.1$ and starts failing to get close to the desired configuration for $\sigma = 0.5$. Plots with $\sigma = 0$ and $\sigma = 0.05$ are omitted as those were similar to the one with $\sigma = 0.1$. We emphasize that the process noise causes unplanned mode changes and the controller seamlessly reacts. This example demonstrates that our method works well with successive linearizations as many multi-contact systems can not be captured via a single LCS approximation.

4.4.3. Cart-pole with Soft Walls

In this subsection, we focus on an under-actuated, multi-contact system and demonstrate the effectiveness of our approach via hardware experiments. We consider a cart-pole that can interact with soft walls as in Figure 4.5. This is a benchmark in contact-based control algorithms

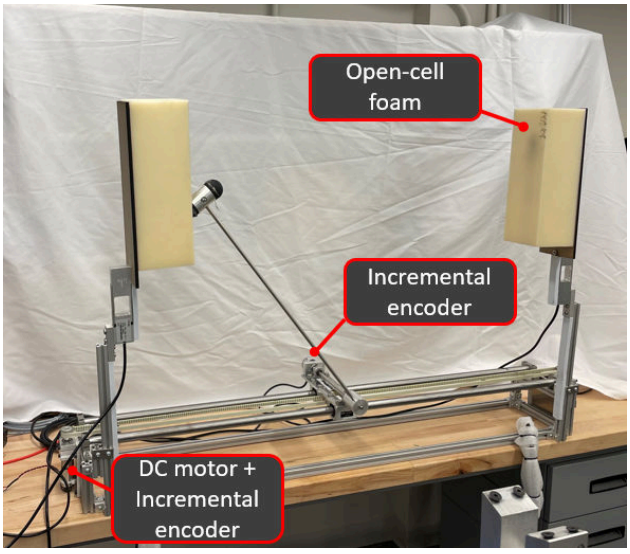

Figure 4.5: Experimental setup for cart-pole with soft walls.

(Marcucci and Tedrake, 2020; Aydinoglu et al., 2021b). In the hardware setup, a DC motor with a belt drive generates the linear motion of the cart. Soft walls are made of open-cell polyurethane foam. The experiments in this subsection are done on a desktop computer with the processor Intel *i7-6700HQ* and *16GB RAM*.

First, numerical experiments are presented. Here, $x^{(1)}$ represents the position of the cart, $x^{(2)}$ represents the position of the pole and $x^{(3)}, x^{(4)}$ represent their velocities respectively. The forces that affect the pole are described by $\lambda^{(1)}$ and $\lambda^{(2)}$ for right and left walls respectively.

The model is linearized around $x^{(2)} = 0$ and $m_c = 0.978$ is the mass of the cart, $m_p = 0.411$ is the combined mass of the pole and the rod, $l_p = 0.6$ is the length of the pole, $l_c = 0.4267$ is the length of the center of mass position, $k_1 = k_2 = 50$ are the stiffness parameter of the walls, $d = 0.35$ is the distance between the origin and the soft walls. Dynamics are discretized using the explicit Euler method with time step $T_s = 0.01$ to obtain the system matrices and use the model in (Aydinoglu et al., 2021b). We note that the MIQP formulation (as in (4.4)) runs at approximately 10 Hz.

We design a controller where $s = 10$, $\rho = 0.1$, $G_k = I$, $\rho_s = 2$, and $N = 10$. There is a clear trade-off between solve time and planning horizon (Li et al., 2021). We test all three projection methods described in Section 4.3 and report run-times (single solve of (4.12)) on Table 4.1 averaged for 800 solves. We also report the average of cost-to-go value assuming all of the methods can run at 100 Hz. Even though the MIQP projection is usually better at exploring new contacts, it does not always result in a better cost. We note that the controller can run slightly faster than 240 Hz if the LCP-based projection is used. For this example, the QP in (4.10) has no inequality constraints and can therefore the KKT conditions can be directly solved.

Next, we test the multi-contact MPC algorithm on the experimental setup shown in Figure 4.5. We consider the same LCS model with $k_1 = k_2 = 100$ as the stiffness parameters of the walls and $d = 0.39$ as the distance between the origin and walls.

For the actual hardware experiments, the parameters of the MPC algorithm are $N = 10$, $\rho_s = 2.3$,

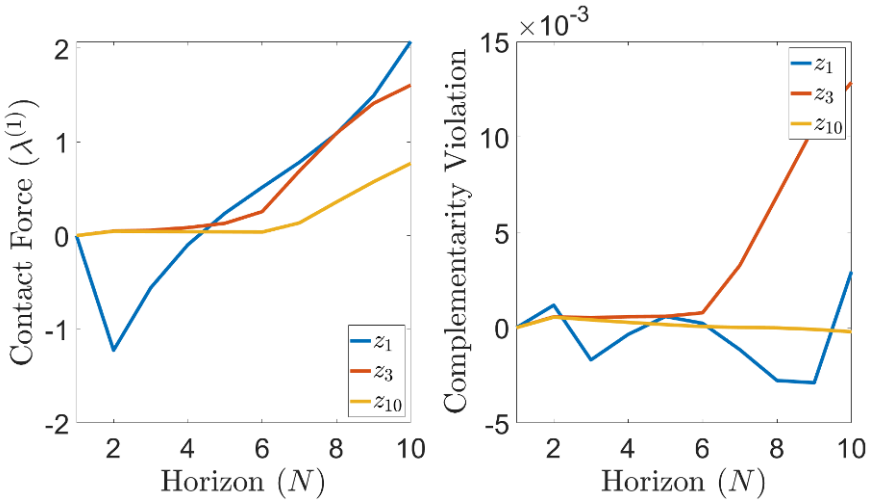

Figure 4.6: Evolution of contact force and complementarity violation during ADMM iteration when the cart is close to a contact surface.

$s = 10$, and $G_k = I$, $\rho = 0.5$. We use the LCP-based projection method, and solve the quadratic programs via utilizing the KKT system. While the algorithm is capable of running faster than 240 Hz, due to the limited bandwidth between our motor controller and computer, we run the system at 100 Hz. In Figure 4.6, we illustrate, for one particular state, the evolution of the contact force throughout the ADMM process (at ADMM steps 1, 3, and 10). Notice that as the the algorithm progresses, complementarity violation decreases.

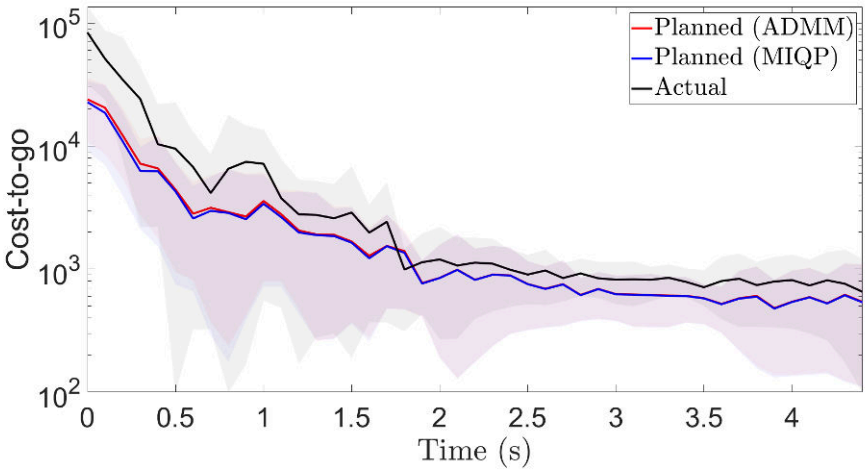

Figure 4.7: Approximate cost-to-go values for the cart-pole experiment.

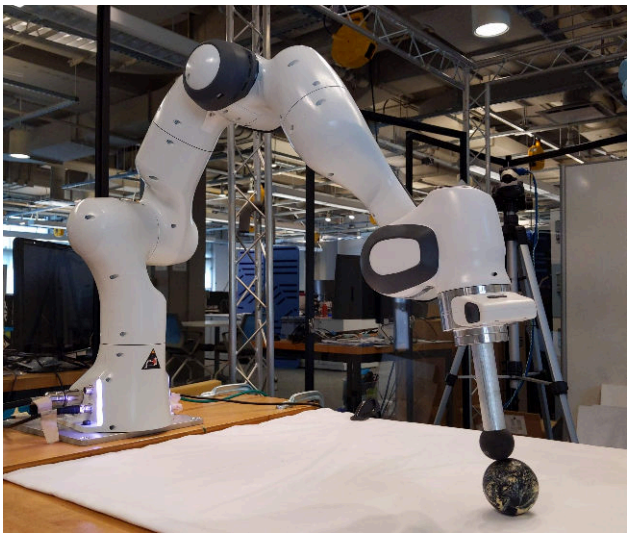

Figure 4.8: Manipulating a rigid object (sphere) with a spherical end-effector attached to the Franka Emika Panda arm.

We initialize the cart at the origin and introduce random perturbations to cover a wide range of initial conditions that lead to contact events. Specifically, we start the cart-pole at the origin where our controller is active. Then, we apply an input disturbance $u_d \sim U[10, 15]$ for 250 ms to force contact events. We repeated this experiment 10 times and our controller managed to stabilize the system in all trials.

To empirically evaluate the gap between C3, which is sub-optimal, and true solutions to (4.3), and to assess the impact of modeling errors, we report the approximate cost-to-go values for our method, MIQP solution (as in (4.4)), and the actual observed states. More precisely, given the current state of the nonlinear plant, we calculate the cost as in (4.3) using the inputs recovered from both the C3 algorithm and MIQP algorithm. Since C3 algorithm is not guaranteed to produce a (x, u, λ) that strictly satisfies complementarity, the predicted cost may not match the simulated cost once u is applied. For the actual plant, we use the data from N steps into the future and calculate the same cost. Notice that even though the cost-to-go of MIQP solution is always lower, as expected, the optimality gap is fairly small. This highlights that C3 finds near-optimal solutions, at least for this particular problem. Also, notice that the actual cost-to-go matches the planned one which further motivates the applicability of LCS representations in model-based control for

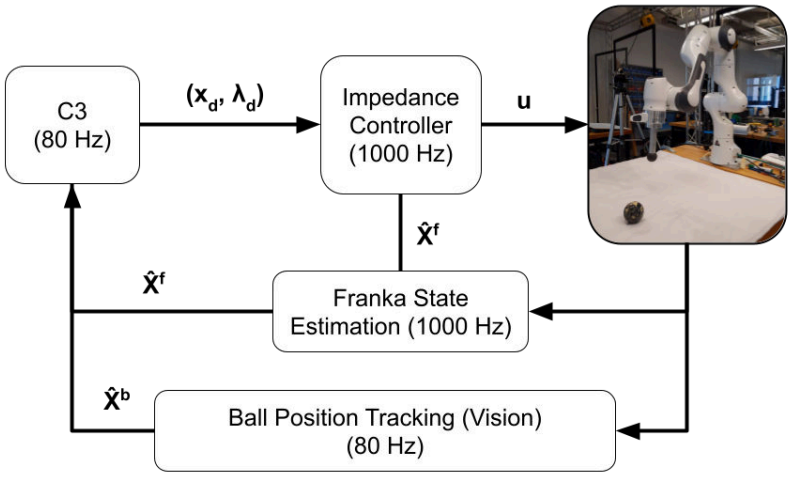

Figure 4.9: Key elements of the controller diagram.

nonlinear multi-contact systems.

4.5. Robot Arm Manipulation

In this section, we demonstrate the effectiveness of our method on multiple robot arm manipulation tasks. In addition to that, we show that C3 can reliably be used as a high-level, real-time controller for multi-contact manipulation tasks that require high-speed reasoning about contact events. For these tasks, our goal is moving one or more rigid spheres along a desired trajectory using a Franka Emika Panda Arm with a spherical end-effector (Figure 4.8). We note that the tasks involve multi-contact interaction between the arm and sphere(s). The robot has to decide on which side/location to make contact, and, in the multi-sphere case, which sphere to touch. Similar to the finger gaiting task in Section 4.4, and in contrast with comparatively simpler planar pushing tasks, these sphere-rolling problems require constantly making and breaking contact to reposition the hand. In addition to that the spheres are quite rigid, and they roll with very little dissipation. Hence it is critical for the controller to make rapid decisions to prevent them from escaping.

Towards this goal, we use the control architecture in Figure 4.9. In this scheme, C3 block acts as a high-level controller and provides the desired state, contact force pair as proposed in Algorithm 3 (often times via using a simplified model of the robot). Then, a low-level impedance controller

Algorithm 3

Require: $\mathcal{M}, \theta, \Delta t, \hat{u}$
1: Estimate state as $\hat{x}$
2: Obtain LCS model $\mathcal{L}_{\Delta t}$ around $(\hat{x}, \hat{u})$ using $\mathcal{M}$
3: Run Algorithm 1 with $\mathcal{L}_{\Delta t}$, θ, $\hat{x}$ and obtain u_{C3}
4: Compute Δt_c: Time spent during steps 1,2,3
5: **return** $(x_d, \lambda_d) = \mathcal{L}_{\Delta t_c}(\hat{x}, u_{\text{C3}})$

tracks these desired values. Please check the Appendix for details on Algorithm 3, the simplified model and the impedance control scheme.

Both C3 and the impedance controller were implemented using the Drake toolbox (Tedrake and the Drake Development Team, 2019), and simulations were performed in Drake environment. OSQP (Stellato et al., 2020) is used to solve quadratic programs and Gurobi (Gurobi Optimization, LLC, 2021) is used for mixed integer programs. The experiments are done on a desktop computer with the processor Intel *i7-11800H* and *16GB RAM*. Reported run-times include all steps in the algorithm.

4.5.1. Simulation example: Trajectory tracking with a single ball

In this subsection, our goal is to move a rigid ball in a circular path inspired by works such as (Kurtz and Lin, 2022). We also aim to make our simulation experiments as close to our hardware experiments (Section 4.5.3) as possible. For the hardware experiments, the vision setup tracks position of the ball, but not orientation. To emulate this, we measure the translational ball state $x^b_{x,y,z}$ at 80 Hz rate and estimate translational velocity of the ball via finite differencing. Similarly, angular velocity is estimated by assuming that the ball is always rolling without slipping. The angular displacement is calculated by integrating the estimated angular velocity.

To track a circular path, we form an LCS approximation around the current state and a quadratic cost for distance to a target state. The target (desired) state for the ball x^b_d is calculated according to its current state x^b. Let $(x_c, y_c) = (0.55, 0)$ be the center of the desired ball path and $r = 0.1$ be its radius. We define α^b as the current phase angle of the ball along its circular path, measured clockwise from the positive y-axis of Franka's base frame. More concretely, $\alpha^b = \text{atan2}(x^b_x - x_c, x^b_y - y_c)$, where

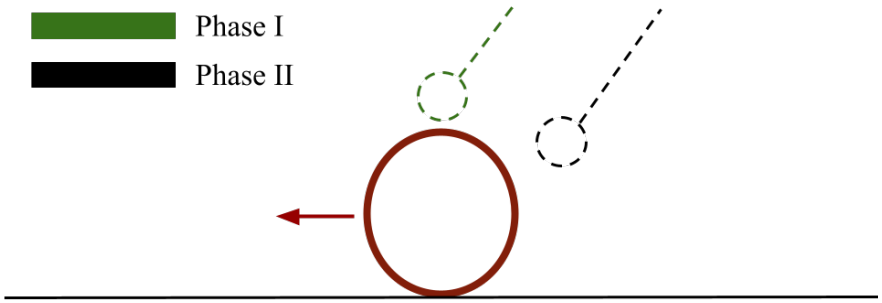

Figure 4.10: 2D toy model to explain the time-based heuristic: Green represents the end-effector bias in Phase I and black represents the end-effector bias in Phase II.

x_x^b and x_y^b are the current x and y coordinates of the ball respectively. Given the center of the path (x_c, y_c), the radius of the path r, and the phase angle of the ball α^b, we generate the next desired ball state x_d^b as:

$$\begin{aligned} x_{d,x}^b &= x_c + r sin(\alpha^b + \alpha^l), \\ x_{d,y}^b &= y_c + r cos(\alpha^b + \alpha^l), \end{aligned} \tag{4.13}$$

where $x_{d,x}^b$ is the desired x coordinate of the ball, $x_{d,y}^b$ is the desired y coordinate of the ball and $\alpha^l = 20^o$ is a design parameter that controls how far ahead the desired ball state should be along the circle with respect to its current position.

One might easily warm-start C3 with an initial trajectory, and corresponding nominal hybrid plan. Here, to demonstrate that the algorithm can reliably discover trajectories in real-time from scratch, we explicitly do not provide such a warm-start. Instead, we add a time-based heuristic as shown in Figure 4.10. In Phase I (1 s long), we pick the end-effector bias slightly above the ball's current state (shown in green). In Phase II (0.5 s long), we pick a set of end-effector biases so that it ends up slightly behind the ball with respect to the desired ball location (shown in black) and also increase the cost on end-effector deviation. It is extremely important to note that we do not specify when and how the contact interactions should happen. All the end-effector biases are above the ball, and algorithm decides that the end-effector should interact with the ball on its own (as there

is also penalty on ball location error). Moreover, C3 can decide that an interaction should happen in either Phase I and Phase II, but we softly encourage interactions to happen in Phase I and the end-effector to reposition in Phase II.

The simplified model, $\mathcal{M}_s$, has nineteen states ($n_x = 19$), twelve complementarity variables ($n_\lambda = 12$) and three inputs ($n_u = 3$). We design a high-level controller where $s = 2$, $\rho = 0.1$, $G_k = I$, $\rho_s = 3$, $\Delta t = 0.1$ and $N = 5$. The controller, C3, can run around 73 Hz. We strictly follow Algorithm 3 where a different LCS model $\mathcal{L}$ is generated using $\mathcal{M}_s$ to generate the desired state, contact force pair, (x_d, λ_d). Afterwards, the impedance controller (B.5) is used to track the the state, contact force pair as discussed.

We performed two different trials to demonstrate the accuracy of our approach as well its robustness against disturbances. For the first (shown as blue in Figure 4.11), we assume that there is no noise on ball position estimation. It is important to emphasize that there are still errors caused due to imperfect state estimation (as we emulate the hardware vision setup) but C3 is robust to these errors and the ball stays within the 2 cm error band.

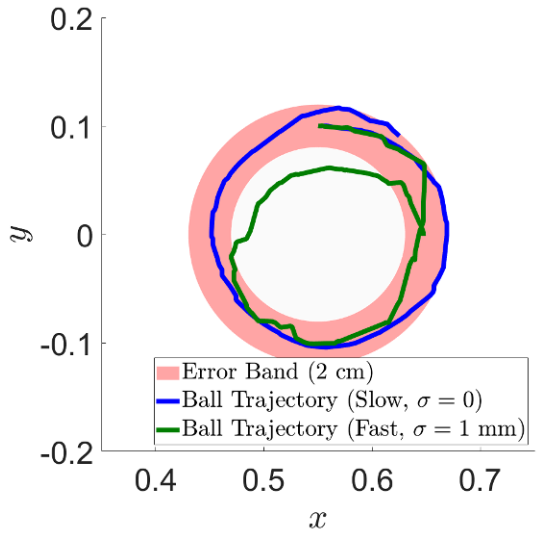

Figure 4.11: Trajectory tracking with single ball: blue represents the slow trajectory ($\sim$ 30 s for full circle) and green represents the fast trajectory ($\sim$ 21 s for full circle).

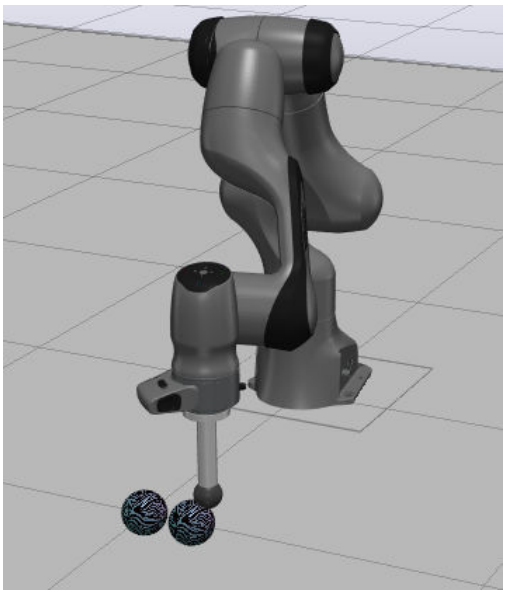

Figure 4.12: The Franka Emika Panda manipulating multiple rigid spheres.

For the second example (shown in green), our goal is to demonstrate that C3 is robust and can recover even if things go wrong (e.g. ball goes out of 2 cm error band). We add noise in state estimation of the ball (Gaussian noise with 1 mm standard deviation) and increase the cost on desired ball location error to achieve faster motions. Since we increase the cost on ball position error, C3 gives more aggressive commands to the low-level controller. Regardless, the ball stays in 2 cm error band in roughly 75% of the motion. Even if the ball rolls away from the 2 cm error band, the controller manages to recover and bring it back into the band utilizing its real-time planning capabilities. This further demonstrates the ability of C3 to recover from perturbations via re-planning contact sequences. In the supplementary video, we present multiple circular trajectories tracked in a row (for both cases) to demonstrate the reliability of our approach.

4.5.2. Simulation example: Trajectory tracking with multiple balls

We consider a similar scenario to the previous example but with multiple balls (Figure 4.12). The goal is to manipulate all of the balls to follow a line. Since the number of potential interactions (ball-ball and end-effector-ball interactions) are higher than the previous example, the hybrid decision making problem is significantly harder to solve. As the number of balls increase, the problem becomes extremely high dimensional (with respect to both the state and the complementarity vari-

able). To slightly reduce the dimension of this problem, we assume that the balls are always on the table and do not slide. We have tried this task both with 2 balls and 3 balls. The reduced-order model for 2-ball setup has fourteen states ($n_x = 14$), eighteen complementarity variables ($n_\lambda = 18$) and three inputs ($n_u = 3$). For the 3-ball setup, the reduced-order model has eighteen states ($n_x = 18$), thirty six complementarity variables ($n_\lambda = 36$) and three inputs ($n_u = 3$).

For i^{th} ball, x^{b_i}, we pick the desired ball state as:

$$x_{d,x}^{b_i} = x_c,$$
$$x_{d,y}^{b_i} = x_y^{b_i} - d_y,$$

where x_c is a constant value and d_y is the desired translation on y-direction. We also follow the same heuristic as the previous section and set the end-effector bias with respect to the ball that has made the least progress in the y direction.

For the 2-ball setup, we design a controller that can run around 40 Hz where $s = 2$, $\rho = 0.1$, $G_k = I$, $\rho_s = 3$, $\Delta t = 0.1$ and $N = 5$. Similar to the previous section, we follow Algorithm 3 to generate desired states and forces (x_d, λ_d). Impedance controller (B.5) is used to track the desired states, and feedforward torque term is taken as zero ($\tau_{ff} = 0$) for this specific example. For the 3-ball setup, everything else is the same but the controller can only run around 20 Hz.

It is important to emphasize that we do not warm-start our algorithm with an initial trajectory and C3 decides on contact interactions (e.g. which ball to interact with, the interaction between balls themselves) on its own. We have successfully accomplished the task with 2 balls where the controller runs around 40 Hz. For the 3-ball setup, C3 ran at 20 Hz, which was empirically too slow to stably complete the task. By adjusting the simulation speed, we can artificially increase the control rate to 40 Hz, where it was successful. This demonstrates the current limits of real-time performance, and also illustrates what might be possible with increased compute and further code optimization. A full video of both experiments are provided in the supplementary material.

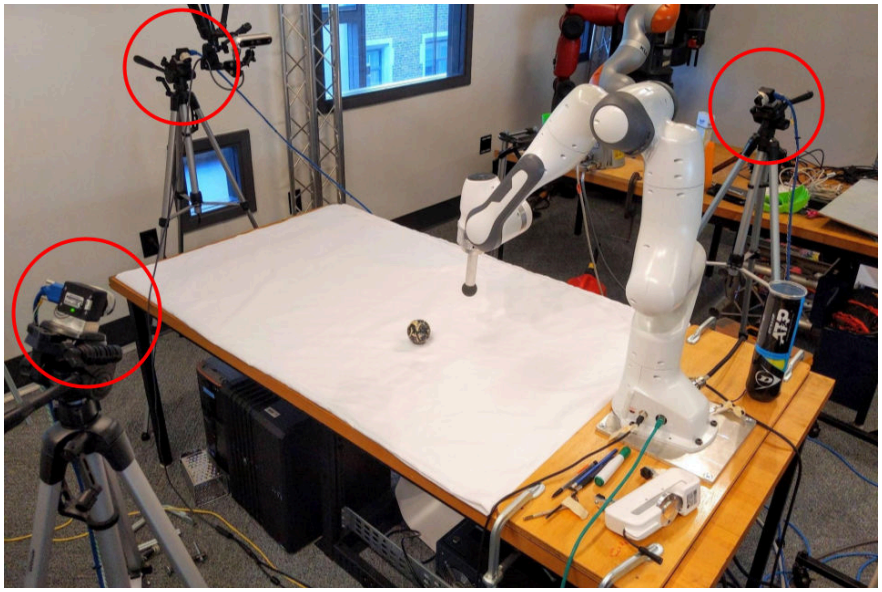

Figure 4.13: Camera setup for hardware experiments. The 3 PointGrey cameras are circled in red.

4.5.3. Hardware experiment: State-based trajectory

In addition to providing numerical examples, we also performed the experiment from Section 4.5.1 on hardware. As before, our goal is to roll the rigid ball in Figure 4.8 along a circular path of radius 0.1 m, centered at (x_c, y_c) in Franka's base frame. We use the same low-level impedance controller presented in (B.5) and C3 as a high-level controller. The next desired state for the ball is computed using the same state-based method described in equation (4.13).

In order to complete this experiment on hardware, we need a vision pipeline to estimate the translational state of the ball, $x^b_{x,y,z}$. We accomplish this by positioning 3 PointGrey cameras around the robot as shown in Figure 4.13. Each camera locates the ball using the Circle Hough Transform algorithm (Bradski, 2000),(Duda and Hart, 1972b), and computes an estimate of $x^b_{x,y,z}$ according to its extrinsic and intrinsic calibration. Next, the position estimates from all 3 cameras are compared, outliers are rejected, and the remaining estimates are averaged. The averaged measurements pass through a low-pass filter to produce the final estimate of the ball's translational state, $\hat{x}^b_{x,y,z}$. The ball's translational velocity, orientation, and angular velocity are estimated from $\hat{x}^b_{x,y,z}$ using the same procedure described in Section 4.5.1. The full vision pipeline operates at roughly 80-90 Hz

and the position estimates typically vary between a range of ±1 mm.

The desired end-effector positions (x_d^e) that C3 generates as the high-level controller and the tracking error of the low-level impedance controller are shown in Figure 4.14. Note that the magnitude of the end-effector's translational error is consistently lower than 0.003 m. These plots suggests that C3 produces effective trajectories that a low-level controller, such as our impedance controller, can realistically track.

The trajectory of the ball is shown in Figure 4.16, where the black line represents the ball path and the red line represents the desired trajectory. Each full rotation around the circle takes 35 s on average to complete. We allow the robot to complete multiple loops to demonstrate the consistency of our approach. A full video of this experiment is provided in the supplementary material.

In this experiment, we use the same time-based heuristic for end-effector biasing shown in Figure 4.10. Although Phase II of this heuristic encourages the robot to roll the ball from the back, C3 can roll from the front instead (adapting to errors) during the actual experiments (see Figure 4.15). This re-emphasizes the fact that we do not specify when or how the robot should interact with the ball, or predefine any trajectory for the end-effector to follow. It also demonstrates that C3 does not

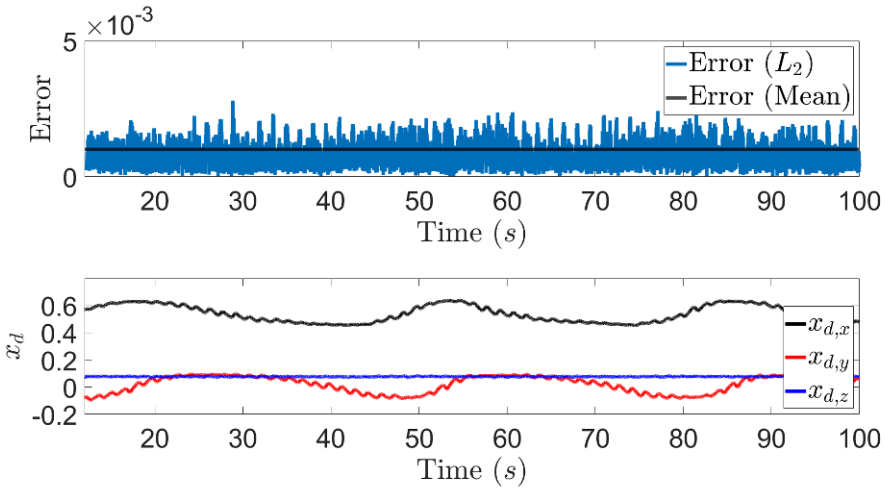

Figure 4.14: End-effector position tracking error ($\sqrt{\left(x_x^e\right)^2 + \left(x_y^e\right)^2 + \left(x_z^e\right)^2}$) and desired end-effector locations given by C3 (x_d) .

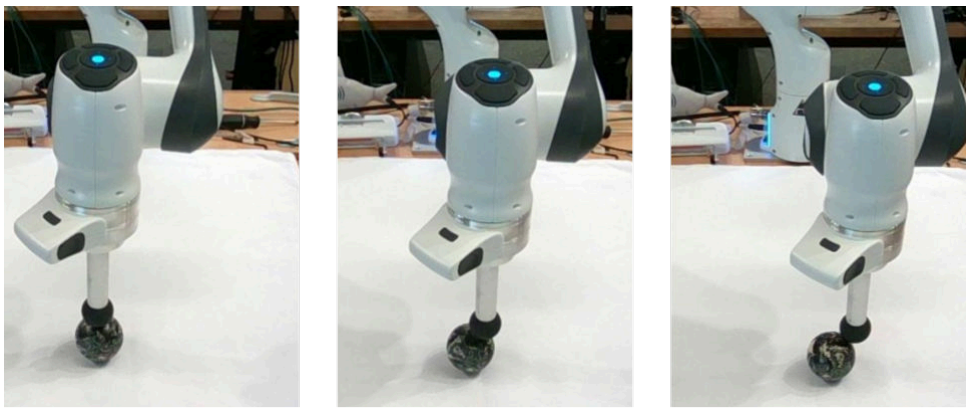

Figure 4.15: An example of a left to right roll using C3 as a high-level planner. Despite the provided heuristic, C3 positions the end-effector directly above the ball at the start of the motion (left image) and rolls from the front.

require perfectly tailored a priori heuristics to perform well. This provides two major advantages. Firstly, it enables C3 to recover, even when its plans deviate from the provided heuristic. Secondly, it eliminates the need for heavily-engineered heuristics, which may be difficult or time-consuming to develop in practise.

Since C3 plans in real-time ($70-80$ Hz), it also exhibits some robustness to model error. Recall that C3 plans using the simplified model, $\mathcal{M}_s$. This model assumes that the rolling surface is perfectly

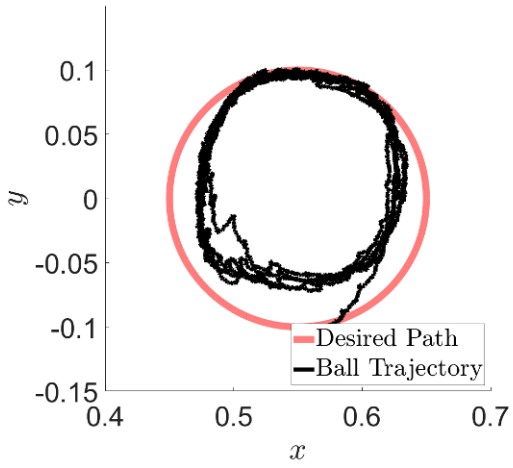

Figure 4.16: Hardware experiment: Tracking a circular path, pink region is the desired path and black demonstrates the actual trajectory.

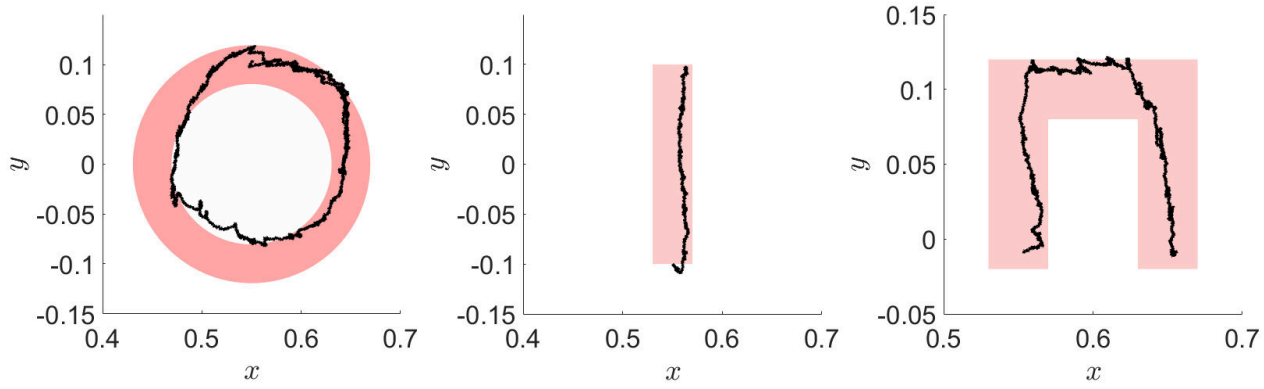

Figure 4.17: Hardware experiment: Tracking time-based trajectories of different shapes. Pink region is the 2 cm error band and black lines represent the states of the ball during the motion.

horizontal; however, the table in the hardware setup is slightly tilted.

This tilt is observable in the supplementary video: note that the ball tends to roll towards the right side of the camera frame whenever it is not in contact with the robot. As a result, the overall shape of the ball's trajectory in Figure 4.16 is circular, but it is shifted in the positive y direction. Another potential source of model error in $\mathcal{M}_s$ is the friction coefficient between the ball and end-effector, which was simply set to $\mu = 1$. Despite these model errors, C3 was still able to generate plans through contact to successfully and consistently complete the task.

Furthermore, the real-time planning capabilities of C3 enable it to continually correct any errors in the ball's trajectory. This may occur if the high-level controller generates an ineffective plan or if the low-level controller momentarily fails to track the end-effector trajectory. For instance, on one of the loops in Figure 4.16, the robot erroneously rolls the ball to $(x_x^b, x_y^b) \approx (0.5, -0.02)$, nearly 5 cm away from the desired path. Nonetheless, C3 is able to correct this mistake on the subsequent rolls and return the ball to its nominal trajectory. This demonstrates C3's self-correction abilities.

Lastly, we also demonstrate that C3 can account for external disturbances where we manually disturb the ball during the experiment. C3 accomplishes the task remarkably well even with such disturbances (shown in supplementary video).

4.5.4. Hardware experiment: Time-based trajectory

In this section, we conduct hardware experiments where the desired ball trajectory is parameterized as a function of time rather than the ball's current state. All other aspects of the experimental setup (the system models, high-level control algorithm, low-level controller, and vision pipeline) remain unchanged from Section 4.5.3.

We experimented with the 3 different time-based trajectories, $x_d^b = x_d^b(t)$ shown in Figure 4.17. For all 3 time-based trajectories, C3 successfully and consistently discovers strategies to roll the ball within the 2 cm error band. Additionally, the shape of the ball's trajectory closely matches the desired trajectory. These experiments demonstrate that C3 can also track time-based trajectories. Note that the same model errors (e.g. table slant, friction coefficient estimates, etc) from Section 4.5.3 are present in these experiments as well; however, since the high-level control algorithm has not changed, C3 remains sufficiently robust to successfully execute the tasks. The full videos of these experiments are available in the supplementary material.

4.6. Adaptive MPC for Manipulation

In the previous sections of Chapter 4, we assumed that we have reasonable estimates of model parameters (e.g. gap function, ϕ). But this assumption does not always hold and the modeling error can be relatively large. It is also important to note that we have empirically observed that if there is a modeling error for the gap function (e.g. ≈ 0.5 cm), C3 can fail to accomplish the task (trajectory tracking with a single ball) in Section 4.5.1. As an example, if the gap function estimates are smaller than the real gap function values, the planner will assume that the robotic arm is/will be in contact with the object whereas in reality no contact interaction will happen (because the planner predicts that the arm is closer to the object than it actually is). On the other hand, such modeling errors can easily occur (e.g. imprecise geometry information) and it is important to overcome such issues.

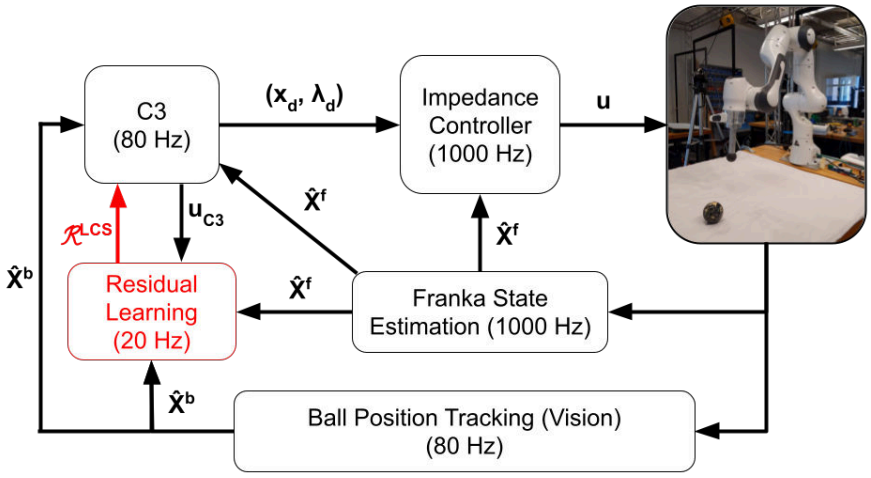

Figure 4.18: Key elements of the controller diagram with residual learning.

To alleviate this problem, we further improve our method so that we can handle relatively large modeling errors. We extend our framework (C3) and combine it with model learning (Jin et al., 2022). The proposed framework is demonstrated in Figure 4.18 and we highlight that this is a modification of our previous approach (shown in Figure 4.9) where the difference is the residual learning module. Unlike our previous work (Jin et al., 2022), this approach focuses on adaptive updates (i.e. collect few data points, calculate gradients and update parameters at real-time rates) and is closer to adaptive control rather than offline system identification.

Towards this direction, instead of assuming that the LCS model C3 uses ($\mathcal{L}_{\Delta t}$) only consists of only the known dynamics, we modify $\mathcal{L}_{\Delta t}$ to include a residual part as:

$$\begin{aligned} x_{k+1} &= (A(x_k) + A_r)x_k + (B(x_k) + B_r r)u_k + (D(x_k) + D_r)\lambda_k + (d(x_k) + d_r), \\ 0 &\leq \lambda_k \perp (E(x_k) + E_r)x_k + (F(x_k) + F_r)\lambda_k + (H(x_k) + H_r)u_k + (c(x_k) + c_r) \geq 0, \end{aligned} \tag{4.14}$$

and we denote the residual parameters as $\mathcal{R}^{\text{LCS}} = \{A_r, B_r, D_r, d_r, E_r, F_r, H_r, c_r\}$. Similarly, we compactly represent the known part of the dynamics as $\mathcal{G}^{\text{LCS}}_{\Delta t}(x_k) = \{A(x_k), B(x_k), D(x_k), d(x_k), E(x_k), F(x_k), H(x_k), c(x_k)\}$ which are directly derived from $\mathcal{M}$ (Instead of Stewart and Trinkle formulation (3.4.2), we use the Anitescu formulation (3.4.2) because the latter returns a positive semi-definite matrix $F(x_k)$ for any x_k). Unlike the previous sections, we

Algorithm 4

Require: $\mathcal{M}, \mathcal{R}^{\text{LCS}}, \theta, \Delta t, \hat{u}$
1: Estimate state as $\hat{x}$
2: Obtain parameters $\mathcal{G}^{\text{LCS}}_{\Delta t}$ around $(\hat{x}, \hat{u})$ using $\mathcal{M}$ (known dynamics)
3: Obtain parameters $\mathcal{R}^{\text{LCS}}$ from the residual learning module
4: Obtain LCS model $\mathcal{L}_{\Delta t}$ combining $\mathcal{G}^{\text{LCS}}_{\Delta t}$ and $\mathcal{R}^{\text{LCS}}$
5: Run Algorithm 1 with $\mathcal{L}_{\Delta t}$, θ, $\hat{x}$ and obtain u_{C3}
6: Compute Δt_c: Time spent during steps 1,2,3
7: Obtain parameters $\mathcal{G}^{\text{LCS}}_{\Delta t_c}$ around $(\hat{x}, \hat{u})$ using $\mathcal{M}$
8: Obtain $\mathcal{R}^{\text{LCS}}_{\text{scaled}}$ from $\mathcal{R}^{\text{LCS}}$ by scaling the terms with $\frac{\Delta t}{\Delta t_c}$
9: Obtain LCS model $\mathcal{L}_{\Delta t_c}$ combining $\mathcal{G}^{\text{LCS}}_{\Delta t_c}$ and $\mathcal{R}^{\text{LCS}}_{\text{scaled}}$
10: **return** $(x_d, \lambda_d) = \mathcal{L}_{\Delta t_c}(\hat{x}, u_{\text{C3}})$

assume that the LCS model C3 uses to plan with, $\mathcal{L}_{\Delta t}$, consists of both the known dynamics $\mathcal{G}^{\text{LCS}}_{\Delta t}$ and residual dynamics $\mathcal{R}^{\text{LCS}}$, which is provided via the residual learning module. We provide the complete algorithm (Algorithm 4) that demonstrates how C3 block in Figure 4.18 functions with the residual learning module involved (for details about update rates and connection of modules, please check Figure 4.18). We underline that it is similar to Algorithm 3 but here C3 also uses the information provided by residual learner, $\mathcal{R}^{\text{LCS}}$, in addition to the known dynamics.

So far, we have detailed our new LCS model that includes the residual parameters $\mathcal{R}^{\text{LCS}}$ and also described how C3 block functions with the residual learning module, but we have not explained the details about the residual learning block. Next, we explain the residual learner further. Once an experiment starts, we start collecting data of the form $\mathcal{B} = \{x^*_{k+1}, x^*_k, u^*_k\}_{k=n_{ct}}^{n_{ct}+n_b}$ in real-time into our buffer where n_{ct} represents the current time-step and n_b is the number of data points stored. For each data point, we also calculate the corresponding matrices, $\mathcal{G}^{\text{LCS}}_{\Delta t}(x_k)$, and define the buffer appended with those matrices as $\mathcal{B}^A = \{x^*_{k+1}, x^*_k, u^*_k, \mathcal{G}^{\text{LCS}}_{\Delta t}(x^*_k)\}_{k=n_{ct}}^{n_{ct}+n_b}$. Next, we introduce the implicit loss function that was defined in (Jin et al., 2022):

$$L_\epsilon(\mathcal{R}^{\text{LCS}}, \mathcal{B}^A) = \sum_{k=n_{ct}}^{n_{ct}+n_b} l_\epsilon(\mathcal{R}^{\text{LCS}}, \mathcal{G}^{\text{LCS}}_{\Delta t}(x^*_k), x^*_{k+1}, x^*_k, u^*_k), \tag{4.15}$$

Algorithm 5

Require: $\mathcal{B}, \mathcal{R}^{\text{LCS}}, \epsilon, \gamma, \xi$
1: Compute $\mathcal{G}^{\text{LCS}}_{\Delta t}(x^*_k)$ for each data point and construct $\mathcal{B}^A$
2: Compute gradient of parameters with respect to $L_\epsilon(\mathcal{R}^{\text{LCS}}, \mathcal{B}^A)$ (e.g. $\nabla_{A^r} L_\epsilon(\cdot)$)
3: Update the residual parameters via gradient step (e.g. $A^r = A^r - \xi \nabla_{A^r} L_\epsilon$)
4: **return** $\mathcal{R}^{\text{LCS}}$ (updated residual parameters)

where the function $l_\epsilon(\mathcal{R}^{\text{LCS}}, \mathcal{G}^{\text{LCS}}_{\Delta t}(x^*_k), x^*_{k+1}, x^*_k, u^*_k)$ is defined as:

$$l_\epsilon(\cdot) = \min_{\lambda_k \geq 0, \eta_k \geq 0} \frac{1}{2} \Big|\Big|(A(x^*_k) + A_r)x^*_k + (B(x^*_k) + B_r)u^*_k + (D(x^*_k) + D_r)\lambda_k + (d(x^*_k) + d_r) - x^*_{k+1}\Big|\Big|^2 \tag{4.16}$$

$$+ \frac{1}{\epsilon}\left(\lambda_k^T \eta_k + \frac{1}{2\gamma}\Big|\Big|(E(x^*_k) + E_r)x^*_k + (F(x^*_k) + F_r)\lambda_k + (H(x^*_k) + H_r)u^*_k + (c(x^*_k) + c_r) - \eta_k\Big|\Big|^2\right). \tag{4.17}$$

Here, γ is a constant such that $0 < \gamma < \sigma_{\min}\left((F(x^*_k) + F_r)^T + F(x^*_k) + F_r\right)$ where $\sigma_{\min}(\cdot)$ denotes the smallest singular value. It is important to note that we parameterize F_r such that $F_r = \epsilon_r I + \tilde{F}_r$ where $\epsilon_r > 0$. As the chosen contact model (Anitescu, 2006) returns $F(x_k) \succeq 0$ for all x_k, we can always find a γ that satisfies the given equality. Similarly, ϵ is a constant such that $\epsilon > 0$. Under these conditions, we can calculate the gradient of the loss function with respect to the residual parameters (e.g. $\nabla_{A_r} L_\epsilon$, $\nabla_{B_r} L_\epsilon$, $\nabla_{D_r} L_\epsilon$, ...) following our previous work (Jin et al., 2022) and the approach requires solving a single QP per data point in the batch (hence is relatively fast). After gradient calculations, one can straightforwardly update the residual parameters taking a simple gradient step (via learning rate $\xi > 0$). Following this discussion, Algorithm 5 summarizes how the residual learning block in Figure 4.18 functions. For the following results, we will focus on learning a residual that is of the form $c_r \in \mathbb{R}^{n_\lambda}$ as we have observed high performance in our experiments with this choice, but our framework is easily adaptable to the more general setting (using other terms in $\mathcal{R}^{\text{LCS}}$).

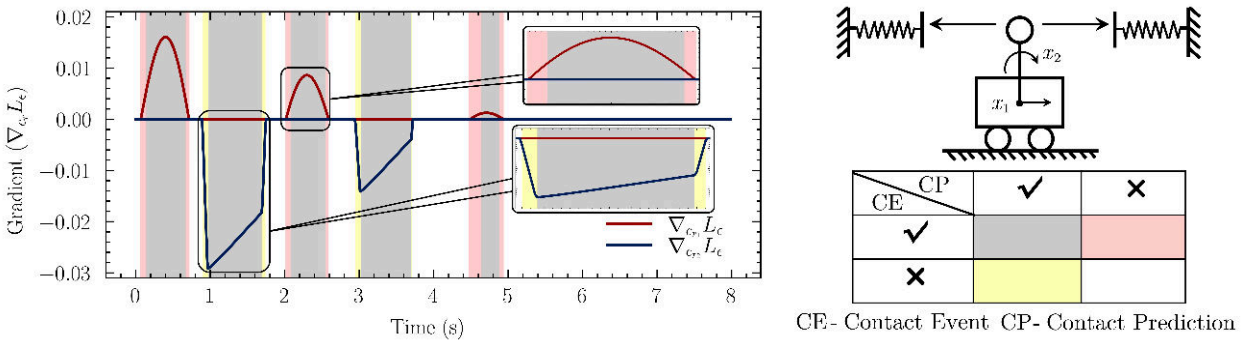

Figure 4.19: A “contact event" refers to a situation where actual physical contact is occurring, while “contact prediction" pertains to instances where the model anticipates contact, potentially inaccurately. Our method can produce meaningful gradients, even when there is no actual contact event (yellow region). The only scenario in which a zero gradient is produced is when the model and data both agree that there is no contact (white region).

4.6.1. Simulation example: Cart-pole with Soft Walls

We consider a classical cart-pole underactuated system which has been augmented with two soft walls (Figure 4.19). The pole can contact these walls, requiring contact-aware control to stabilize the system:

$$x_{k+1} = Ax_k + Bu_k + D\lambda_k + d,$$

$$0 \leq \lambda_k \perp Ex_k + F\lambda_k + c \geq 0,$$

where we have perfect knowledge of the parameters except for c. We write $c = \hat{c} + \Delta\phi$, where $\hat{c}$ is our initial guess of the parameter. For the purposes of illustration, we begin with an initial error of $\Delta\phi = [-.15, .15]$. Notice that the model error that is related to each contact (λ_k) has a different sign. Our deliberate choice of this setup was made to create a range of scenarios, allowing the contact model to predict both situations: contact and no contact, as well as the presence or absence of actual contact events during those situations. In Figure 4.19, it can be seen that our method successfully returns useful gradient information in all of those scenarios. We highlight that our approach is capable of adapting even when there is no actual contact events (Figure 4.19). We also show that our framework successfully stabilizes the system as well as learning the true residual values (Figure 4.20).

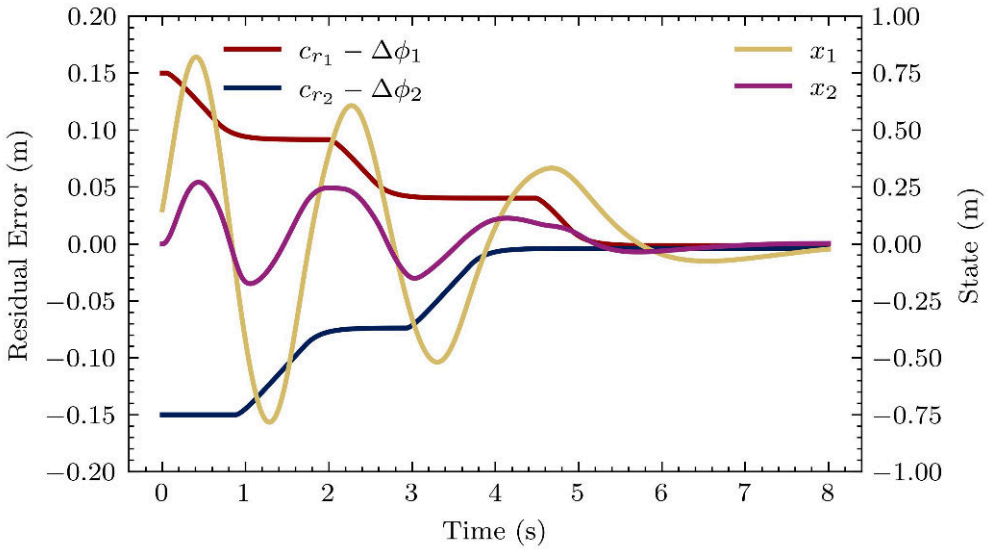

Figure 4.20: Stabilization of the cart-pole system and convergence of residual.

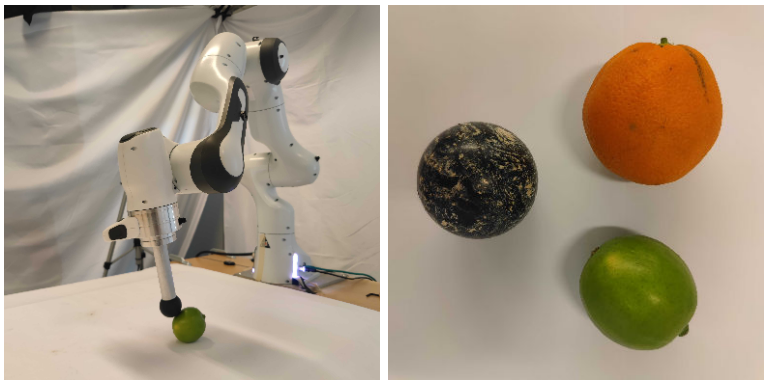

Figure 4.21: Given an initial guess that the object is a rigid sphere, the controller adapts its model of the governing contact dynamics to roll and push real fruits (orange, lime) with a Franka Emika Panda arm, tracking a desired motion.

4.6.2. Hardware Experiment: Adaptive Trajectory Tracking

Here, we show that our real-time adaptive MPC framework can reliably be used for multi-contact manipulation tasks that require high-speed reasoning about contact events. Our goal is to roll a rigid ball along a circular trajectory using a Franka Emika Panda Arm (Figure 4.21). We perform vision-based estimation for the ball using Hough transform (Duda and Hart, 1972a) and utilize an impedance controller (Hogan, 1985; Khatib, 1987) to track high-level commands that our adaptive MPC produces (Figure 4.18). For adaptive MPC, we simplify the arm as a point contact.

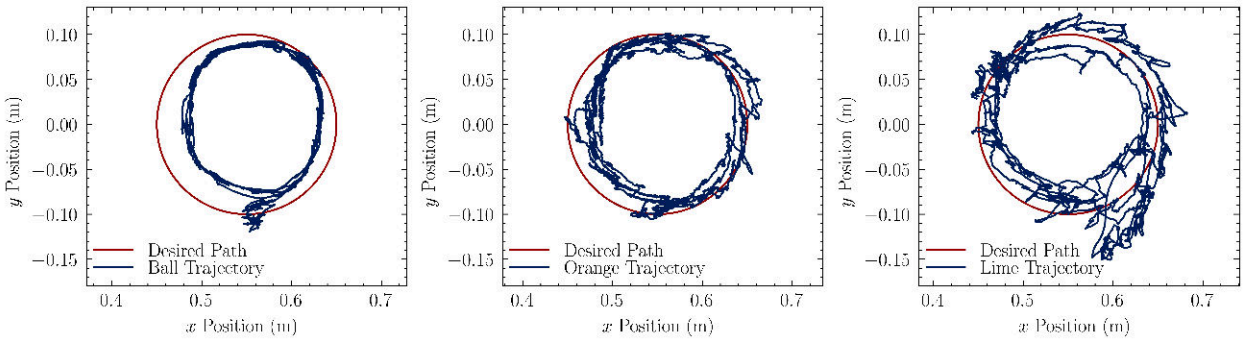

Figure 4.22: Rolling a rigid ball and fruits (left: ball, middle: orange, right: lime) starting with an inaccurate model.

In previous sections, careful manual identification of model parameters was required to achieve success. Here, via our adaptive MPC approach, we do not need to assume access to an accurate model. For this experiment, the actual radius of the ball is 5 mm smaller than our parameter estimation. The state estimation for the ball is noisy (vision-based, 80 Hz) and we do not have accurate estimation of many model parameters such as coefficient of friction. As a result, MPC with this incorrect model fails to interact with the ball, and therefore fails to accomplish the task. Due to the stiff, hybrid nature of contact dynamics, we have found MPC to be highly sensitive to modeling errors that effect the contact/no-contact transition.

Even though the purely model-based approach fails, our adaptive MPC framework successfully accomplishes the task. The residual learning module (Algorithm 5, 20 Hz), in combination with the C3 module (80 Hz), identifies the discrepancy at real-time rates. During the experiments, it can be observed that at the beginning the end-effector misses the ball due to inaccurate radius estimation. The C3 module then tries, and fails, to initiate contact. As motivated earlier, this discrepancy creates a non-zero gradient and therefore the residual values that correspond to end-effector ball contact pair gradually increase (shown in Figure 4.23).

In the end, with the learned residual compensating for the inaccurate model, our approach successfully rolls the ball for 4 successive circles. Figure 4.23 shows the loss, $\mathcal{L}_\epsilon$, as well as the residual, c_r, for this experiment. The residual for the end-effector and ball contact shows convergence as the rolling proceeds and the loss curve shows a decreasing trend, proving the effectiveness of our

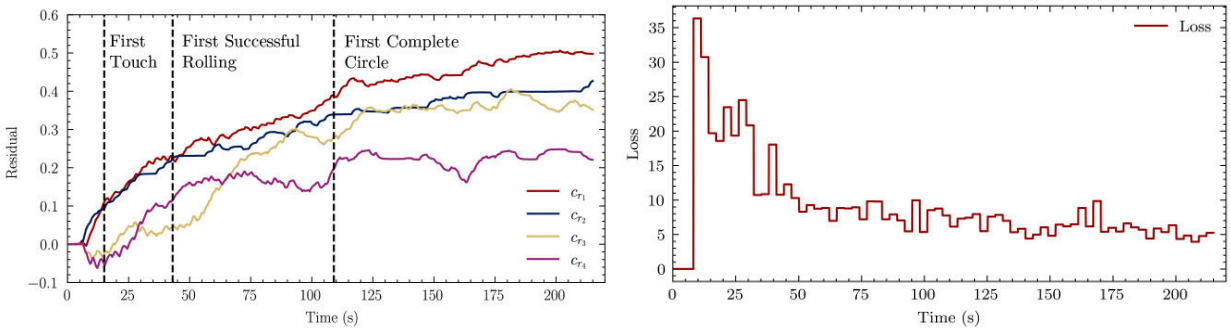

Figure 4.23: Hardware Experiment: Residual evolution of end-effector and ball contact pair, and loss curve of the learning process. The full experiment lasts for 315 s. We highlight the adaptation process until loss and residual converge.

method. We note that in this experiment, the LCP violation rate is $\epsilon = 10^{-7}$, the terms in Q_d related to balls translational velocity are set to 10^5 (others are set to zero), learning rate is $\xi = 10^{-3}$, stiffness parameter is $\gamma = 10^{-2}$ and we have a batch data size of $n_b = 10$. We demonstrate the trajectories of the ball with respect to the desired path in Figure 4.22.

4.6.3. Hardware Experiment: Objects of Non-smooth Surface

We repeat the previous hardware experiment (Section 4.6.2) with slightly deformable objects that have non-smooth surface geometries (Figure 4.21). We do not have accurate initial estimates of the geometry as our estimate of a fruit's geometry is simply a sphere. For each experiment experiments, we assign an approximate radius for the given fruit and set the sphere radius in our prior model accordingly.

Due to the bulges and dents on the fruit's surface, initiating rolling can be hard since the fruit can roll back even after a push in the correct direction. MPC without adaptation gets stuck in the beginning and fails to even initiate rolling. In contrast, our method quickly adapts and starts making stronger pushes to initiate rolling (as shown in the supplementary video). In 90% of the 20 experiments, our method successfully initiated rolling and started tracking the circular path, while MPC without adaptation was never successful (0%).

Fruits tend to produce unpredictable motions at times due to its non-smooth surface geometry, but our method can adapt and accomplish the task. Our approach has managed to track at least one

circle for 70% of the trials with orange and 50% of trials with lime (20 experiments for each case). In Figure 4.22, we report the long-term tracking (4 successive circles) performance of our method with multiple different fruits.

4.7. Conclusion and Discussion

In this work, we present an algorithm, C3, for model predictive control of multi-contact systems. The algorithm relies on solving QP's accompanied by projections, both of which can be solved efficiently for multi-contact systems. The effectiveness of our approach is verified on six numerical examples and our results are validated on two different experimental setups. We demonstrate that C3 can be reliably used both as a high-level and a low-level controller. For fairly complex examples with frictional contact, our method has a fast run-time. We also demonstrate through experiments that our heuristic can find close-to-optimal strategies. For cases where reasonable estimates of model parameters are not available, we combine C3 with a residual learner to learn such parameters at real-time rates and accomplish the given tasks.

The framework tackles the hybrid MPC problem by shifting the complexity to the projection sub-problems. These projections can be difficult to solve, especially as we rely on the MIQP-based projection method for problems with frictional contact. Exploring alternate heuristics for the projection step is of future interest. Similarly, it would be interested to leverage learning-based approaches (Cauligi et al., 2020) to speed up the process.

As discussed in Section 4.5, C3 generates desired contact forces but these have been used in a feedforward manner. It would be interesting to utilize tactile sensors and tactile feedback controllers (Aydinoglu et al., 2021b; Kim et al., 2023) that track these desired contact forces. We believe that this can improve the robustness of our approach and is one of the crucial steps moving forward.

The choice of parameters (such as C3 parameters θ) greatly affects the performance of the algorithm and exploring different approaches for deciding on parameters is of future interest too.

CHAPTER 5

STABILIZATION OF COMPLEMENTARITY SYSTEMS VIA CONTACT-AWARE CONTROLLERS

Parts of this chapter were previously published as parts of Alp Aydinoglu, Victor M Preciado, and Michael Posa. Contact-Aware Controller Design for Complementarity Systems. In 2020 IEEE International Conference on Robotics and Automation (ICRA), pages 1525–1531, ©2020 IEEE. And also as parts of Alp Aydinoglu, Philip Sieg, Victor M Preciado, and Michael Posa. Stabilization of Complementarity Systems via Contact-Aware Controllers. IEEE Transactions on Robotics, 38(3), ©2022 IEEE.

5.1. Introduction

In Chapter 4, we focused on discrete-time models and now we shift our focus to LCS approximations of continuous-time models $\mathcal{M}$ (Section 3.5). Focusing on LCS approximations, we propose a control framework which can utilize tactile information by exploiting the complementarity structure of contact dynamics. Since many robotic tasks, like manipulation and locomotion, are fundamentally based in making and breaking contact with the environment, state-of-the-art control policies struggle to deal with the hybrid nature of multi-contact motion. Such controllers often rely heavily upon heuristics or, due to the combinatorial structure in the dynamics, are unsuitable for real-time control. Principled deployment of tactile sensors offers a promising mechanism for stable and robust control, but modern approaches often use this data in an ad hoc manner, for instance to guide guarded moves. This framework can close the loop on tactile sensors and it is non-combinatorial, enabling optimization algorithms to automatically synthesize provably stable control policies. We demonstrate this approach on multiple numerical examples, including quasi-static friction problems and a high dimensional problem with ten contacts. We also validate our results on an experimental setup and show the effectiveness of the proposed method on an underactuated multi-contact system.

5.2. Linear Complementarity Systems with Tactile Feedback

We present a tactile feedback controller where the input is dependent both on the state and the contact force $\big(u = u(x, \lambda)\big)$, unlike the common approach of designing controllers only using the state feedback, $\big(u = u(x)\big)$. The section concludes with a description of complementarity models with such tactile feedback controllers.

5.2.1. Tactile Feedback and Related Complementarity Models

We introduce the tactile feedback controller:

$$u(\bar{x}, \lambda) \;=\; K\bar{x} + L\lambda, \tag{5.1}$$

where $K \in \mathbb{R}^{n_k \times n_x}$ and $L \in \mathbb{R}^{n_k \times m}$. Using this control law, (3.3) can be transformed into the following LCS:

$$\begin{aligned} \dot{x} &= Ax + D\lambda + a, \\ 0 \leq \lambda \perp Ex + F\lambda + c &\geq 0, \end{aligned} \tag{5.2}$$

where $A \in \mathbb{R}^{n \times n}$, $D \in \mathbb{R}^{n \times m}$, $a \in \mathbb{R}^n$, $E \in \mathbb{R}^{m \times n}$, $F \in \mathbb{R}^{m \times m}$, $c \in \mathbb{R}^m$.

If the contact force does not depend on the input ($H = 0$), then application of the control law (5.1) trivially produces (5.2) with $A \;=\; \bar{A} + BK$, $D = \bar{D} + BL$, $E = \bar{E}$, and $F = \bar{F}$. In this case, note that $x = \bar{x}$ and $n = n_x$.

Next, consider the case where the contact force depends on the input ($H \neq 0$). Since the input $u = u(\bar{x}, \lambda)$ similarly depends on the contact force, this introduces an algebraic loop. One might attempt to resolve this loop by simultaneously solving for both u and λ, leading to the closed-loop LCS:

$$\begin{aligned} \dot{x} &= (\bar{A} + BK)x + (\bar{D} + BL)\lambda + a, \\ 0 \leq \lambda \perp Ex + F\lambda + c &\geq 0, \end{aligned}$$

where $x = \bar{x}$, $E = \bar{E} + HK$ and $F = \bar{F} + HL$. Observe that the matrix F depends on the choice of the contact gain matrix L. Due to this dependency, the cardinality of the solution set

$\mathrm{SOL}(Ex + c, F)$ for a given x might change depending on the value of L. This is illustrated via an example.

Example 6. *Consider the complementarity constraint:*

$$0 \leq \lambda \perp x + u + \lambda \geq 0,$$

where $x, u, \lambda \in \mathbb{R}$. If u is independent of λ, observe that $SOL(x + u, F)$ is a singleton for all pairs (x, u) since $F = [1]$ is a P-matrix. In this case, the contact force $\lambda_o(x, u)$ is equal to

$$\lambda_o(x, u) = \max\{0, -x - u\}.$$

However, for some choices of force-dependent inputs, this is no longer the case. From $u = L\lambda$, it follows that $F = [1 + L]$. For the case $L = -1$, the LCP for the closed-loop system is

$$0 \leq \lambda \perp x \geq 0.$$

The solution set is then:

$$SOL(x, F = 0) = \begin{cases} \{0\} & \text{if} \quad x > 0, \\ [0, \infty) & \text{if} \quad x = 0, \\ \emptyset & \text{if} \quad x < 0. \end{cases}$$

There are infinitely many solutions for $x = 0$ and no solutions for $x < 0$.

Furthermore, resolving the algebraic loop by solving simultaneously for the contact force and the input is not physically realistic since control policies can not instantaneously respond to tactile measurements. As illustrated in Example 6, it is also mathematically problematic. Therefore, we will use the standard approach of modeling delay. Specifically, the following low-pass filter model captures the input delay:

$$\dot{\tau} = \kappa(u - \tau), \tag{5.3}$$

where $\kappa \in \mathbb{R}^+$ is the rate parameter. Using the low-pass filter model, we obtain the linear complementarity system:

$$\begin{aligned} \dot{\bar{x}} &= \bar{A}\bar{x} + B\tau + \bar{D}\lambda + a, \\ \dot{\tau} &= \kappa(u - \tau), \\ 0 \leq \lambda \perp \bar{E}\bar{x} + F\lambda + H\tau + c &\geq 0. \end{aligned} \tag{5.4}$$

Observe that the LCS model in (5.4) has the same form with (5.2) with the input (5.1):

$$\begin{aligned} \dot{x} &= \begin{bmatrix} \bar{A} & B \\ \kappa K & -\kappa I \end{bmatrix} x+, \begin{bmatrix} \bar{D} \\ \kappa L \end{bmatrix} \lambda + \begin{bmatrix} a \\ 0 \end{bmatrix}, \\ 0 \leq \lambda \perp &\begin{bmatrix} \bar{E} & H \end{bmatrix} x + F\lambda + c \geq 0, \end{aligned}$$

where $x = \begin{bmatrix} \bar{x}^T & \tau^T \end{bmatrix}^T$. Also, we highlight that the delay decouples u and λ so the matrix F does not depend on the contact gain matrix L. Notice that the state is augmented ($n > n_x$) to obtain (5.2). Alternatively, one could add delay to the sensor dynamics:

$$\dot{\tau}_s = \kappa_s(\lambda - \tau_s).$$

While this approach would similarly resolve the algebraic loop, in this work we found out that modeling input delay produced better numerical results when combined with the algorithmic approach in Section 5.4.

Using the control format in (5.1), for notational compactness, we will now exclusively consider closed-loop LCS in the form of (5.2). As a result of filtering, the matrix F will be independent of the tactile feedback gain L.

5.2.2. Solution Concept

We introduce a solution concept for complementarity systems (3.4) and (5.2) similar to ((Çamlibel et al., 2002), Definition 3.6).

Definition 7. *A pair of functions* $(x(t), \lambda(t))$ *is a solution of the complementarity system,*

$$\dot{x} = f(x, \lambda),$$
$$0 \leq \lambda \perp \Phi(x, \lambda) \geq 0,$$

where $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ *and* $\Phi : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^m$ *with the initial condition* $x(0) = x_0$ *if:*

$$x(t) \in AC([0, T]),\ \forall\, T \geq 0,$$
$$\dot{x}(t) = f(x(t), \lambda(t)) \text{ for almost all } t \in \mathbb{R}^+,$$
$$0 \leq \lambda(t) \perp \Phi(x(t), \lambda(t)) \geq 0 \text{ for almost all } t \in \mathbb{R}^+,$$
$$\lambda(t) \text{ is almost everywhere differentiable.}$$

It is important to note that we restrict ourselves to complementarity systems where the state, $x(t)$, is absolutely continuous which is well-studied in the literature (Shen and Pang, 2007; Brogliato and Thibault, 2010; Çamlıbel et al., 2002). The models considered in this work have no jumps (e.g. impact). Observe that $\lambda(t)$ can be discontinuous and it is assumed that $\lambda(t)$ is almost everywhere differentiable moving forward. Consider the following proposition by Camlibel et al. (Camlibel et al., 2006):

Proposition 8. *For every* x_0*, the LCS* (5.2) *has a unique* C^1 *(hence absolutely continuous) trajectory* $x(t)$ *defined for all* $t \geq 0$ *if and only if the set* $DSOL(Ex + c, F)$ *is a singleton for every* $x \in \mathbb{R}^n$*.*

Throughout this chapter, focus is on LCS models where $\bar{D}\text{SOL}(Ex + c, F)$ is a singleton. Note that unlike $x(t)$, the trajectory $\lambda(t)$ is not necessarily unique which is observed in friction models (Section 5.5). Recent results indicate that it may, ultimately, be possible to eliminate this assumption on $\bar{D}\text{SOL}(Ex + c, F)$ (Brogliato and Thibault, 2010; Brogliato and Tanwani, 2020), though such exploration is outside the scope of this chapter.

In Sections 5.3 and 5.4, this structure is leveraged to similarly ensure that $L\text{SOL}(Ex + c, F)$ is

a singleton for all x. Since we consider models such that $\bar{D}\text{SOL}(Ex + c, F)$ is a singleton and F is independent of the controller (Section 5.2), the condition that $L\text{SOL}(Ex + c, F)$ is a singleton suffices to ensure that $D\text{SOL}(Ex + c, F)$ is also a singleton. The trajectories of the closed-loop system (5.2) remain absolutely continuous as long as $L\text{SOL}(Ex + c, F)$ is a singleton for all x (following Proposition 6).

Moving forward, denote $\mathcal{S}(t_0, x_0)$ as the set of all trajectories $x(t)$, with $t \geq t_0$, such that $x(0) = x_0$. The dependency on the LCS parameters is suppressed for ease of notation. Observe that $S(t_0, x_0)$ is also a singleton following Proposition 8 if $D\text{SOL}(Ex + c, F)$ is a singleton for every x.

5.3. Stabilization of the Linear Complementarity System

In this section, conditions for stabilization using non-smooth monontonic Lyapunov functions and contact-aware controllers are constructed.

Notions of stability from (Smirnov, 2002) are adopted. If F is a P-matrix, these are equivalent to the notions of stability for differential equations where the right-hand side is Lipschitz continuous, though possibly non-smooth (Camlibel et al., 2006), (Khalil, 2002).

Definition 9. *The equilibrium x_e of LCS* (5.2) *is*

1. *stable in the sense of Lyapunov if, given any $\epsilon > 0$, there exists $\delta > 0$ such that*

$$||x_e - x_0|| < \delta \implies ||x(t) - x_e|| < \epsilon \; \forall t \geq 0$$

   for any x_0 and $x(t) \in \mathcal{S}(0, x_0)$.

2. *asymptotically stable if it is stable and $\delta > 0$ exists s.t.*

$$||x_e - x_0|| < \delta \implies \lim_{t \to \infty} x(t) = x_e$$

   for any x_0 and $x(t) \in \mathcal{S}(0, x_0)$.

5.3.1. Non-Smooth Lyapunov Function

In Lyapunov based analysis and synthesis methods, one desires to search over a wide class of functions. Here, piecewise quadratic Lyapunov functions are considered. They are more expressive than a Lyapunov function common to all modes (as was used in (Posa et al., 2015)), which makes it a more powerful choice than a single quadratic Lyapunov function (Johansson and Rantzer, 1997). Towards this direction, consider a variant of the Lyapunov function introduced in (Camlibel et al., 2006):

$$V(x,\lambda) = x^T P x + 2x^T Q\lambda + \lambda^T R\lambda + p^T x + r^T\lambda + z, \tag{5.5}$$

where $P \in \mathbb{R}^{n\times n}$, $Q \in \mathbb{R}^{n\times m}$, $R \in \mathbb{R}^{m\times m}$, $p \in \mathbb{R}^n$, $r \in \mathbb{R}^m$, and $z \in \mathbb{R}$. The Lyapunov function (5.5) is quadratic in terms of the pair (x,λ). If F is a P-matrix, it is piecewise quadratic in x since $\lambda = \lambda(x)$ is a piecewise affine function. For example, if all contact forces are inactive, $\lambda = 0$, then $V(x,\lambda(x)) = x^T Px + p^T x + z$. Even though V is non-smooth, it is locally Lipschitz continuous with respect to x if F is a P-matrix (Camlibel et al., 2006).

If $\mathrm{SOL}(Ex+c,F)$ is not a singleton then $\lambda(t)$ can be discontinuous in t due to the multi-valued nature of $\mathrm{SOL}(Ex(t)+c,F)$. Similarly, $V(x(t),\lambda(t))$ can be discontinuous due to the terms $Q\mathrm{SOL}(Ex(t)+c,F)$, $\mathrm{SOL}(Ex(t)+c,F)^T R\mathrm{SOL}(Ex(t)+c,F)$ and $r^T\mathrm{SOL}(Ex(t)+c,F)$. Next, it is shown that without appropriate restrictions, these discontinuities imply that such functions V cannot be valid monotonic Lyapunov functions.

Example 10. *Consider the LCS:*

$$\dot{x} = -x + \lambda_1 + \lambda_2,$$
$$0 \le \lambda_1 \perp x + \lambda_1 + \lambda_2 \ge 0,$$
$$0 \le \lambda_2 \perp x + \lambda_1 + \lambda_2 \ge 0,$$

where $x, \lambda_1, \lambda_2 \in \mathbb{R}$. As seen in Figure 5.1, the Lyapunov function jumps at $t = t^*$ for a solution $(x(t), \lambda^{dec}(t))$ and the Lyapunov function decreases ($V(t_+^*) > V(t_-^*)$). Then, consider $\lambda^{inc}(t)$ that jumps at $t = t^*$ where $\lambda_-^{inc} = \lambda_+^{dec}$ and $\lambda_+^{inc} = \lambda_-^{dec}$. The Lyapunov functions value increases after

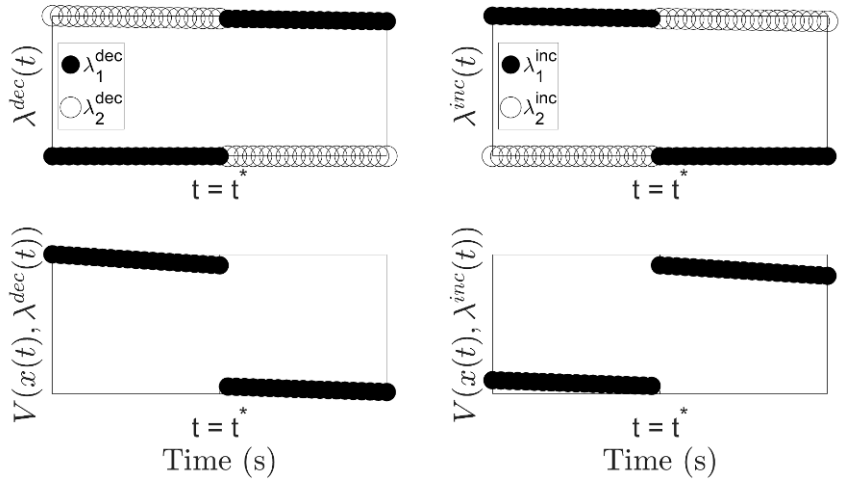

Figure 5.1: Two different solutions for the Lyapunov function. For the solution λ^{dec}, λ_1^{dec} and λ_2^{dec} represent the first and second elements of the solution vector respectively (similarly for λ^{inc}).

the jump at $t = t^*$ for the solution $(x(t), \lambda^{inc}(t))$ by construction as seen in Figure 5.1.

If the function V jumps at a time t^* for one solution and its value decreases, then another solution exists where V jumps at t^* and the function's value increases as illustrated in Example 10. Hence, the Lyapunov function cannot decrease monotonically along all solutions due to the multi-valued nature of $\text{SOL}(Ex+c, F)$. For this reason, we focus on mappings W such that $W\text{SOL}(Ex+c, F)$ is single-valued. Using such mappings, one can parameterize Q, R and r such that $Q\text{SOL}(Ex(t)+c, F)$, $\text{SOL}(Ex(t)+c, F)^T R\text{SOL}(Ex(t)+c, F)$ and $r^T\text{SOL}(Ex(t)+c, F)$ are single-valued for all t even when F is not a P-matrix.

Proposition 11. *((Camlibel et al., 2006), Proposition 3.9) Assume that $W\,SOL(q, F)$ is a singleton for all q where $W \in \mathbb{R}^{n_w \times m}$. Then, the map $q \mapsto W\,SOL(q, F)$ is a continuous piecewise linear function of q.*

One can construct a Lyapunov function using a matrix W as in Proposition 11 where $Q\text{SOL}(Ex+c)$,

$R\text{SOL}(Ex+c)$, and $r^T\text{SOL}(Ex+c)$ are singletons for all $x \in \mathbb{R}^n$ as follows:

$$V(x,\lambda) = x^T P x + 2x^T \tilde{Q} W\lambda + \lambda^T W^T \tilde{R} W\lambda + p^T x + \tilde{r}^T W\lambda + z, \tag{5.6}$$

where $W \in \mathbb{R}^{n_w \times m}$, $\tilde{Q} \in \mathbb{R}^{n \times n_w}$, $\tilde{R} \in \mathbb{R}^{n_w \times n_w}$, $\tilde{r} \in \mathbb{R}^{n_w}$ and $W\text{SOL}(Ex+c, F)$ is a singleton. Next, it is shown that V is a non-smooth, continuous piecewise quadratic function in x and is locally Lipschitz continuous which are helpful properties in establishing stability results.

Lemma 12. *The Lyapunov function*[4] $V(x, \lambda(x))$ *as in* (5.6) *is locally Lispchitz continuous in* x. *Furthermore,* $\bar{V}(t) = V(x(t), \lambda(t)) \in AC([0,T])$ *for all* $T \geq 0$ *for the solutions as in Definition 7.*

Proof. Since $W\lambda(x)$ is Lipschitz continuous in x, $V(x, \lambda(x))$ is locally Lispchitz continuous in x. Because $x(t)$ is absolutely continuous and $V(x, \lambda(x))$ is locally Lipschitz continuous, V is absolutely continuous in time. □

From this point onward, without loss of generality, the Lyapunov function as defined in (5.6) is used. Observe that if F is a P-matrix, one can trivially choose $W = I$. For many practical examples, it is possible to find such a matrix W. However, such a W is not guaranteed to exist when F is not a P-matrix. In Section 5.4-C, it is shown how to generate W algorithmically.

Remark 13. *Similar to the Lyapunov function, the input* (5.1) *is not necessarily continuous in time. If one desires a controller that is continuous in time, then the parameterization*

$$u(\bar{x}, \lambda) = K\bar{x} + \tilde{L} W\lambda, \tag{5.7}$$

leads to a controller u *that is continuous in time. As discussed in Section 5.5-B, it is required that* $DSOL(Ex+c, F)$ *be a singleton for all* x. *Therefore, we restrict the controller to be of the form* (5.7). *For all of the examples in Section 5.5, the parameterization* $L = \tilde{L}W$ *is used.*

[4] $\lambda(x)$ is the set-valued function $\lambda(x) = \text{SOL(Ex+c,F)}$.

5.3.2. Conditions for Stabilization

Now, conditions for stability in the sense of Lyapunov are constructed with the controller gains K and L as in (5.1), and the piecewise quadratic Lyapunov function V. These conditions will be the building blocks for the controller design method proposed in Section 5.4.

Theorem 14. *Consider the linear complementarity system* (5.2)*, and the Lyapunov function* (5.6) *with W such that $WSOL(Ex + c, F)$ is a singleton for all x. Assume there exists a solution for every x_0 and $x_e = 0$ is an equilibrium. If for all solutions $(x(t), \lambda(t))$[5], there exist strictly positive constants γ_1, γ_2, matrices K, L[6] and a function V such that*

$$\gamma_1 ||x(t)||_2^2 \leq V(x(t), \lambda(t)) \leq \gamma_2 ||x(t)||_2^2,$$

and $\frac{d\bar{V}(t)}{dt} \leq 0$ for almost all t, then $x_e = 0$ is Lyapunov stable. Furthermore, if there exists a strictly positive constant γ_3 such that $\frac{d\bar{V}(t)}{dt} \leq -\gamma_3 ||x(t)||_2^2$ for almost all t, then $x_e = 0$ is exponentially stable.

Proof. Let the solution $(x(t), \lambda(t))$ be arbitrary. Following Lemma 12, $\bar{V}(t)$ is absolutely continuous and almost everywhere differentiable on $[0, T]$ for all T. Then we have

$$\bar{V}(t) = \bar{V}(0) + \int_0^t \dot{\bar{V}}(s) ds \leq \bar{V}(0),$$

since $\dot{\bar{V}} \leq 0$ for almost all $t \in \mathbb{R}^+$. Since V is bounded and non-increasing, the rest follows from standard arguments for Lyapunov stability.

In order to prove exponential stability, observe that $\dot{\bar{V}}(t) \leq -\gamma_3 ||x(t)||_2^2$. Hence, it follows that

$$||x(t)||_2^2 \leq \frac{1}{\gamma_1} \bar{V}(0) - \frac{\gamma_3}{\gamma_1} \int_0^t ||x(s)||_2^2 ds.$$

[5] Dependence on x_0 and LCS parameters is suppressed.

[6] $\frac{d\bar{V}(t)}{dt}$ depends on K and L since $\dot{x}$ is a function of K and L.

Using Grönwall's inequality, it follows that

$$||x(t)||_2^2 \leq \frac{1}{\gamma_1}\bar{V}(0)e^{-\frac{\gamma_3}{\gamma_1}t} \leq \frac{\gamma_2}{\gamma_1}||x_0||_2^2 e^{-\frac{\gamma_3}{\gamma_1}t}.$$

Hence we conclude that the equilibrium is exponentially stable. □

Theorem 14 establishes sufficient conditions to stabilize the LCS in (5.2). In Section 5.4, it is shown how Theorem 14 can be used to algorithmically synthesize a controller. Next, observe that an upper-bound always exists under certain assumptions.

Remark 15. *If $c \geq 0$ and $p = r = z = 0$, there exists a γ_2 such that $V(x, \lambda) \leq \gamma_2||x||_2^2$ since $W\lambda(x) \leq \rho||x||_2$ for all x for some ρ.*

For this special case, an upper-bound always exists and one does not need to verify that V is upper-bounded when algorithmically synthesizing a controller (Section 5.4).

5.4. Controller Design as a Bilinear Matrix Inequality Feasibility Problem

Controller design for complementarity systems in the form of BMI's have been explored before for the case where F in (5.2) is zero and without tactile feedback ($L = 0$) (Brogliato et al., 2007). In this section, these results are extended and it is shown how Theorem 14 can be used to algorithmically synthesize a controller when there is tactile feedback and the matrix F is non-zero. Then, the controller design problem is converted into a bilinear matrix inequality (BMI) feasibility problem. A convex optimization program is proposed to find a matrix W such that $W\text{SOL}(Ex + c, F)$ is a singleton for all x.

5.4.1. Sufficient Conditions for Stabilization

The sufficient conditions in Theorem 14 need to be satisfied for all solutions of the LCS (5.2). Now, we will transform them into matrix inequalities over two basic semialgebraic sets $\Gamma_{\text{SOL}}(E, F, c)$ and $\Gamma'_{\text{SOL}}(E, F, c)$. Define the set:

$$\Gamma_{\text{SOL}}(E, F, c) = \{(\boldsymbol{x}, \boldsymbol{\lambda}) : 0 \leq \boldsymbol{\lambda} \perp E\boldsymbol{x} + c + F\boldsymbol{\lambda} \geq 0\},$$

where $(\boldsymbol{x}, \boldsymbol{\lambda}) \in \Gamma_{\mathrm{SOL}}(E, F, c)$ are represented as quadratic inequalities. Similarly, define the following set:

$$\begin{aligned}\Gamma'_{\mathrm{SOL}}(E, F, c) = \{(\boldsymbol{x}, \boldsymbol{\lambda}, \dot{\boldsymbol{\lambda}}) \big| \exists \boldsymbol{\rho}, \boldsymbol{\mu} : \boldsymbol{\lambda} \in SOL(E\boldsymbol{x} + c, F),\\ E\dot{x} + F\dot{\boldsymbol{\lambda}} + \boldsymbol{\rho} = 0,\ \boldsymbol{\lambda}_i\boldsymbol{\rho}_i = 0, \dot{\boldsymbol{\lambda}}_i + \boldsymbol{\mu}_i = 0,\\ (E_i^T\boldsymbol{x} + F_i^T\boldsymbol{\lambda} + c_i)\boldsymbol{\mu}_i = 0, \boldsymbol{\mu}_i\boldsymbol{\rho}_i = 0\},\end{aligned} \tag{5.8}$$

where $\dot{x} = A\boldsymbol{x} + D\boldsymbol{\lambda} + a$ and $\boldsymbol{\mu}, \boldsymbol{\rho}$ are slack variables. Here $\dot{\boldsymbol{\lambda}}$ expresses the time derivative of the force. Next, we express the matrix inequalities over semialgebraic sets where a method similar to construction of contact LCP's in ((Brogliato, 2016), Section 5.1.2.1) is used.

Proposition 16. *If the inequalities*

$$\gamma_1||\boldsymbol{x}||_2^2 \leq V(\boldsymbol{x}, \boldsymbol{\lambda}) \leq \gamma_2||\boldsymbol{x}||_2^2,\ \forall(\boldsymbol{x}, \boldsymbol{\lambda}) \in \Gamma_{SOL}, \tag{5.9}$$

$$\nabla_{\boldsymbol{x}} V(\boldsymbol{x}, \boldsymbol{\lambda})^T\dot{x} + \nabla_{\boldsymbol{\lambda}} V(\boldsymbol{x}, \boldsymbol{\lambda})^T\dot{\boldsymbol{\lambda}} \leq 0,\ \forall(\boldsymbol{x}, \boldsymbol{\lambda}, \dot{\boldsymbol{\lambda}}) \in \Gamma'_{SOL}, \tag{5.10}$$

hold for the LCS (5.2) *where* $\dot{x} = A\boldsymbol{x} + D\boldsymbol{\lambda} + a$*, then the following inequalities hold for all solutions* $(x(t), \lambda(t))$ *of the LCS*

$$\gamma_1||x(t)||_2^2 \leq V(x(t), \lambda(t)) \leq \gamma_2||x(t)||_2^2, \tag{5.11}$$

$$\frac{d}{dt}V(x(t), \lambda(t)) \leq 0, \tag{5.12}$$

for almost all $t \geq 0$*.*

Proof. Consider an arbitrary solution, $(x(t), \lambda(t))$ of (5.2). First we will show that (5.9) implies (5.11). From Definition 7, it follows that $\lambda(t) \in \mathrm{SOL}(Ex + c, F)$ and $(x(t), \lambda(t)) \in \Gamma_{\mathrm{SOL}}(E, F, c)$ for almost all $t \geq 0$. The result follows from (5.9).

Next, we show that (5.10) implies (5.12). We show that

$$\lambda(t) \in \mathrm{SOL}(Ex(t) + c, F), \tag{5.13}$$

$$\lambda_i(t) > 0 \implies E_i^T \dot{x}(t) + F_i^T \dot{\lambda}(t) = 0, \tag{5.14}$$

$$E_i^T x(t) + F_i^T \lambda(t) + c_i > 0 \implies \dot{\lambda}_i(t) = 0, \tag{5.15}$$

$$\left.\begin{array}{r} \lambda_i = 0 \\ E_i^T x + F_i^T \lambda + c = 0 \end{array}\right\} \implies \begin{array}{c} \dot{\lambda}_i = 0 \quad \text{or} \\ E_i^T \dot{x} + F_i^T \dot{\lambda} = 0, \end{array} \tag{5.16}$$

hold for almost all $t \geq 0$ where dependency on t in (5.16) is suppressed for space limitations, $\dot{\lambda}(t) = \frac{d\lambda}{dt}$, and (5.13) directly follows from the definition of solution.

We define $n_i(t) = E_i^T x(t) + F_i^T \lambda(t) + c$ for notational simplicity. To prove (5.14)-(5.16), observe that for almost all $t \geq 0$, there exists an $\epsilon > 0$ such that both $\lambda_i(t)$ and $n_i(t)$ are continuous in the interval $[t - \epsilon, t + \epsilon]$. For almost all $t \geq 0$ if $\lambda_i(t) > 0$, then $n_i(t) = 0$ for a neighborhood around t hence $E_i^T \dot{x}(t) + F_i^T \dot{\lambda}(t) = 0$ and (5.14) follows. Similarly observe that if $n_i(t) > 0$, then $\lambda(t) = 0$ and $\dot{\lambda}(t) = 0$ as in (5.15). (5.16) follows from the the fact that both $\lambda_i(t)$ and $n_i(t)$ cannot be positive at the same time.

Suppose (5.13)-(5.16) hold at some time t^* and consider $x_* = x(t^*)$, $\lambda_* = \lambda(t^*)$, $\dot{\lambda}_* = \lambda(t^*)$ and $\dot{x}_* = \dot{x}(t^*)$. We will show that $(x_*, \lambda_*, \dot{\lambda}_*) \in \Gamma'_{\mathrm{SOL}}(E, F, c)$.

There are 3 cases. First consider the case where $\lambda_{i,*} > 0$ and therefore $(E_i^T x_* + F_i^T \lambda_* + c_i) = 0$. Observe that all equalities in (5.8) are satisfied with $\rho_{i,*} = 0$ and $\mu_{i,*} = -\dot{\lambda}_{i,*}$.

For the case where $\lambda_{i,*} = 0$ and $(E_i^T x_* + F_i^T \lambda_* + c_i) > 0$, all equalities are satisfied with $\rho_{i,*} = -E_i^T \dot{x}_* - F\dot{\lambda}_{i,*}$ and $\mu_{i,*} = 0$.

For the last case where both $(E_i^T x_* + F_i^T \lambda_* + c_i) = \lambda_{i,*} = 0$, the equalities are satisfied with either $\rho_{i,*} = 0$, $\mu_{i,*} = -\dot{\lambda}_{i,*}$ or $\rho_{i,*} = -E_i^T \dot{x}_* - F\dot{\lambda}_{i,*}$, $\mu_{i,*} = 0$.

Since the implications hold for almost all t, we conclude that $(x(t), \lambda(t), \dot{\lambda}(t)) \in \Gamma'_{\mathrm{SOL}}$ for almost all

$t \geq 0$. The result follows from (5.10). $\square$

Following Proposition 16 and Theorem 14, if the matrix inequalities (see Appendix for the full derivation) over basic semialgebraic sets (5.9), (5.10) are satisfied, one can conclude that the equilibrium x_e is Lyapunov stable. Similarly, one can show that the equilibrium is exponentially stable if the left side of (5.10) is upper-bounded by $-\gamma_3||\boldsymbol{x}||_2^2$ as in Theorem 14.

5.4.2. Control Design

Now, the sets $\Gamma_{\text{SOL}}(E,F,c)$, $\Gamma'_{\text{SOL}}(E,F,c)$ are defined and it is assumed that there is access to a matrix W. The BMI feasibility problem with strictly positive constants γ_1, γ_2 and non-negative γ_3 can be formulated as:

$$\begin{aligned} &\text{find} \quad V(\boldsymbol{x},\boldsymbol{\lambda}), K, L \\ &\text{s.t.} \quad \gamma_1||\boldsymbol{x}||_2^2 \leq V(\boldsymbol{x},\boldsymbol{\lambda}) \leq \gamma_2||\boldsymbol{x}||_2^2,\ (\boldsymbol{x},\boldsymbol{\lambda}) \in \Gamma_{\text{SOL}}(E,F,c), \\ &\qquad \frac{dV}{dt} \leq -\gamma_3||\boldsymbol{x}||_2^2,\ (\boldsymbol{x},\boldsymbol{\lambda},\dot{\boldsymbol{\lambda}}) \in \Gamma'_{\text{SOL}}(E,F,c), \end{aligned} \tag{5.17}$$

with the function $V(\boldsymbol{x},\boldsymbol{\lambda})$ as in (5.6) and

$$\begin{aligned} \frac{dV}{dt} &= 2\boldsymbol{x}^T P(A\boldsymbol{x} + D\boldsymbol{\lambda} + a) + 2(A\boldsymbol{x} + D\boldsymbol{\lambda} + a)^T \tilde{Q} W \boldsymbol{\lambda} \\ &\quad + 2\boldsymbol{x}^T \tilde{Q} W \dot{\lambda} + 2\boldsymbol{\lambda}^T W^T \tilde{R} W \dot{\boldsymbol{\lambda}} + p^T \dot{x} + \tilde{r} W_i^T \dot{\boldsymbol{\lambda}}, \end{aligned}$$

with the controller as in (5.7). Here, V encodes the non-smoothness of the problem structure, mirroring the structure of the LCS, and allow tactile feedback design without exponential enumeration. This is an appealing middle ground between the common Lyapunov function of our prior work (Posa et al., 2015), and purely hybrid approaches (Papachristodoulou and Prajna, 2009; Marcucci and Tedrake, 2019). Whereas methods like our prior work are more conservative than the proposed method (Example 3.3., (Camlibel et al., 2006)), purely hybrid methods are less conservative at the cost of additional computation. Here, it is possible to assign a different Lyapunov function and a control policy for each mode while avoiding mode enumeration so the approach can

scale to large number of contacts m unlike purely hybrid approaches.

Notice that the inequality with $\frac{dV}{dt}$ in (5.17) is a bilinear matrix inequality because of the bilinear terms such as PA where A as in (5.2) depends on the gain matrix K as discussed in Section 5.2. In (5.17), the problem of designing a control policy is formulated as finding a feasible solution for a set of bilinear matrix inequalities. The sets $\Gamma_{\mathrm{SOL}}(E, F, c)$ and $\Gamma'_{\mathrm{SOL}}(E, F, c)$ are incorporated via the S-procedure.

5.4.3. Computing W via Polynomial Optimization

Until this point, it is assumed that there is access to a W such that $W\mathrm{SOL}(Ex+c, F)$ is a singleton for all $x \in \mathbb{R}^n$. If F is a P-matrix, one can always pick $W = I$ since $\mathrm{SOL}(Ex + c, F)$ is a singleton for any x as discussed earlier. For the non-P case, one can always trivially pick $W = 0$ which turns the Lyapunov function (5.6) into a common Lyapunov function, and controller (5.1) into a non-switching state feedback controller. On the other hand, it is clearly better to search over a wider range of Lyapunov functions and controllers (Johansson and Rantzer, 1997), (Camlibel et al., 2006). Hence it is desired to maximize the rank of W. More precisely, consider the following optimization problem:

$$\begin{aligned} \max_{W} \quad & \mathrm{rank}(W) \\ \text{subject to} \quad & W\mathrm{SOL}(q, F) \text{ is a singleton for all } q. \end{aligned}$$

To solve this problem, an algorithm based in a sequence of convex optimization problems is proposed. First, consider the following sub-problem:

$$\operatorname*{find}_{w} \quad \text{s.t.} \quad w^T\mathrm{SOL}(q, F) \text{ is a singleton for all } q, \tag{5.18}$$

where $w \in \mathbb{R}^m$. Using this sub-problem, we will construct an algorithm to find matrices W. Notice that a w such that $|w^T(\lambda_{1,q} - \lambda_{2,q})| = 0$ holds for all q, and all $\lambda_{1,q}, \lambda_{2,q} \in \mathrm{SOL}(q, F)$ satisfies (5.18). Next, we demonstrate that it is sufficient to satisfy $|w^T(\lambda_{1,q} - \lambda_{2,q})| \leq \eta$ for any $\eta > 0$ to satisfy (5.18).

Proposition 17. *Suppose that for some w, the following inequalities hold for all q, all $\lambda_{1,q}, \lambda_{2,q} \in SOL(q, F)$:*

$$\begin{aligned}(\eta + w^T(\lambda_{1,q} - \lambda_{2,q}))(\lambda_{1,q}^T\lambda_{1,q} + \lambda_{2,q}^T\lambda_{2,q}) \geq 0,\\ (\eta - w^T(\lambda_{1,q} - \lambda_{2,q}))(\lambda_{1,q}^T\lambda_{1,q} + \lambda_{2,q}^T\lambda_{2,q}) \geq 0,\end{aligned} \tag{5.19}$$

where $\eta > 0$ is a constant slack parameter. Then, $w^T SOL(q, F)$ is a singleton for all q.

Proof. Observe that

$$\lambda \in \text{SOL}(q, F) \implies \alpha\lambda \in \text{SOL}(\alpha q, F),$$

for all $\alpha \geq 0$. We will show that the positive homogeneity property leads to:

$$|w^T(\lambda_{1,q} - \lambda_{2,q})| \leq \eta \; \forall q \implies |w^T(\lambda_{1,q} - \lambda_{2,q})| = 0 \; \forall q. \tag{5.20}$$

Assume that there exists $\eta^* > 0$ such that $|w^T(\lambda_{1,q^*} - \lambda_{2,q^*})| = \eta^*$ for some q^*. Pick $\alpha^* > 0$ such that $\alpha^*\eta^* > \eta$ and $|w^T(\alpha^*\lambda_{1,q^*} - \alpha^*\lambda_{2,q*})| = \alpha^*\eta^* > \eta$. Due to the positive homogeneity property, there exists $\lambda_{1,\alpha q^*}$ and $\lambda_{2,\alpha q^*}$ such that $|w^T(\lambda_{1,\alpha q^*} - \lambda_{2,\alpha q*})| = \alpha^*\eta^* > \eta$. This leads to a contradiction.

Next, we consider q such that $(\lambda_{1,q}^T\lambda_{1,q}+\lambda_{2,q}^T\lambda_{2,q}) > 0$. It follows from (5.19) that $|w^T(\lambda_{1,q}-\lambda_{2,q})| \leq \eta$ hence $|w^T(\lambda_{1,q} - \lambda_{2,q})| = 0$. If $(\lambda_{1,q}^T\lambda_{1,q} + \lambda_{2,q}^T\lambda_{2,q}) = 0$, then $\lambda_{1,q} = \lambda_{2,q} = 0$ and it trivially holds that $w^T\lambda_{1,q} = w^T\lambda_{2,q}$. Therefore, for any w such that (5.19) holds, $w^T\lambda_{1,q} = w^T\lambda_{2,q}$ also holds for all q. Hence, $w^T SOL(q, F)$ is a singleton for all q. □

Finding a w such that (5.19) holds can be reduced to a polynomial optimization program (see Appendix, (A.1)) and any vector w that satisfies (5.19) also satisfies (5.18). Experimentally, we found that use of the slack variable η was helpful to avoid numerical difficulties in the solvers. The solvers (Mosek (Mosek, 2010), SeDuMi (Sturm, 1999)) had trouble verifying the status of the problem (feasible or infeasible) when $\eta = 0$. The $(\lambda_{1,q}^T\lambda_{1,q} + \lambda_{2,q}^T\lambda_{2,q})$ terms in (5.19) are introduced because there are degree two S-procedure terms (as shown in (A.1)) and the inequalities must be at least degree two to use such S-procedure terms.

Algorithm 6 Find W

Require: F

Initialization : $N \leftarrow I$, $W = [\,]$, $r = \mathbf{1}$, $w = \mathbf{1}$

1: **while** $\min r^T N^T w \neq 0$ **do**
2: $r \sim U(0,1)$
3: Solve (5.21) and obtain w
4: **if** $\min r^T N^T w < 0$ **then**
5: $W \leftarrow \begin{bmatrix} W \\ w^T \end{bmatrix}$
6: Calculate N based on $\mathcal{N}(W)$
7: **end if**
8: **end while**
9: **return** W

Given a matrix $W_d \in \mathbb{R}^{s\times m}$, one can utilize Proposition 17 in order to find a vector w such that $w^T \mathrm{SOL}(q, F)$ is a singleton (for all q) and w^T is linearly independent with the rows of W_d. Consider the optimization problem:

$$\begin{aligned} \min_{w,\eta} \quad & r^T N^T w \\ \text{subject to} \quad & (\eta + w^T(\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_2))(\boldsymbol{\lambda}_1^T\boldsymbol{\lambda}_1 + \boldsymbol{\lambda}_2^T\boldsymbol{\lambda}_2) \geq 0, \\ & (\eta - w^T(\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_2))(\boldsymbol{\lambda}_1^T\boldsymbol{\lambda}_1 + \boldsymbol{\lambda}_2^T\boldsymbol{\lambda}_2) \geq 0, \\ & \text{for } \boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2 \in \mathrm{SOL}(\boldsymbol{q}, F), \\ & |w_i| \leq 1,\ \forall i,\ \eta \geq 0, \end{aligned} \tag{5.21}$$

where r is a random vector with entries sampled from uniform distribution ($r_i \sim U(0,1)$), N is a basis for the nullspace of W_d ($\mathcal{N}(W_d)$), $\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \boldsymbol{q}$ are indeterminates, and the set inclusion is incorporated via the S-procedure for the first two inequalities (Appendix B, (A.2)). Randomness is introduced in (5.21) in order to ensure that the objective function is almost surely strictly negative. More precisely, a linear objective that is strictly negative if any N such that $Nw \neq 0$ exists is needed and projection onto a random vector r ensures this with probability one.

Proposition 18. *Consider W_d and the optimization* (5.21)*. If there exists a w such that the constraints hold, and w^T is linearly independent with the rows of W_d, then* $\min r^T N^T w < 0$ *almost*

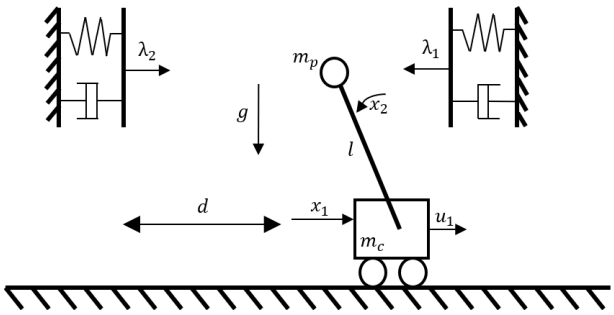

Figure 5.2: Benchmark problem: Regulation of the cart-pole system to the origin with soft walls.

surely.

Proof. Assume there exists a w that is feasible for optimization problem (5.21) and w^T is linearly independent with the rows of W_d. Then, $||N^T w|| > 0$ and $r^T N^T w \neq 0$ with probability 1. By homogeneity, an optimal w_* can be found such that $r^T N^T w_* < 0$. □

We now introduce Algorithm 6 based on Proposition 18. The algorithm almost surely finds a new linearly independent vector that satisfies the constraints in (5.21) if it exists and terminates when there are not any left. Notice that the randomization process affects the outcome (W) but we are only interested in finding a W such that $W\text{SOL}(q, F)$ is a singleton for all q hence this effect can be neglected.

5.5. Numerical Examples

In this paper, the YALMIP (Löfberg, 2004) toolbox and PENBMI (Kočvara and Stingl, 2003) are used to formulate and solve bilinear matrix inequalities. SeDuMi (Sturm, 1999) and Mosek (Mosek, 2010) are used for solving the semidefinite programs (SDPs). PATH (Dirkse and Ferris, 1995) has been used to solve the linear complementarity problems when performing simulations. MATLAB's stiff solver 'ode15s' is used while performing simulations of the LCS models and Euler's method with stepsize 10^{-4} is used for the nonlinear models. The code for all examples is available[7] and

[7] https://github.com/AlpAydinoglu/cdesign

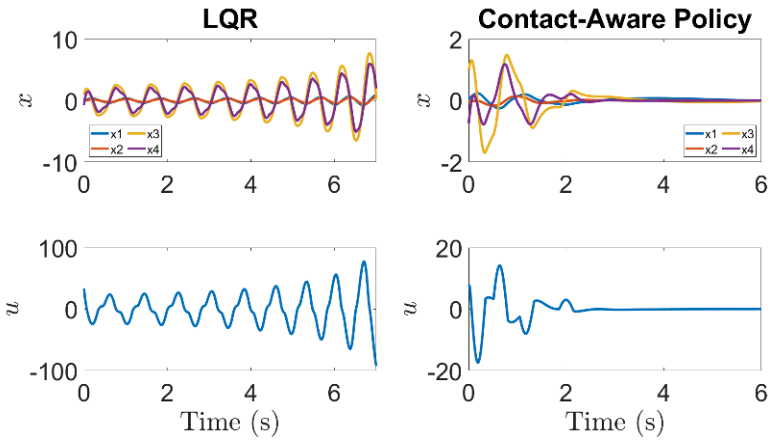

Figure 5.3: Performance of LQR and contact-aware policy starting from the same initial condition for the cart-pole with soft walls example. LQR is unstable whereas contact-aware policy is successful.

examples are provided with a video depiction[8]. The experiments are done on a desktop computer with the processor *Intel i7-9700* and *16GB RAM*. We have reported the offline computation times for all of the examples and emphasize that our controller only requires only a few addition and multiplication operations when running online and is applicable in real time context after the offline computations are done.

5.5.1. Cart-Pole with Soft Walls

Consider the cart-pole system where the goal is to balance the pole and regulate the cart to the center, where there are frictionless walls, modeled via spring contacts, on both sides. This problem, or a slight variation of it, has been used as a benchmark in control through contact (Marcucci and Tedrake, 2019), (Deits et al., 2019), (Marcucci et al., 2017) and the model is shown in Figure 5.2.

First, the model where there is no damping is analyzed. In this model, the x_1 is the position of the cart, x_2 is the angle of the pole, and x_3, x_4 are their respective time derivatives. The input u_1 is a force applied to the cart, and the contact forces of the walls are represented with λ_1 and λ_2, leading

[8]https://www.youtube.com/watch?v=CpYZcinYuQM

to the LCS

$$\dot{x}_1 = x_3, \tag{5.22}$$

$$\dot{x}_2 = x_4, \tag{5.23}$$

$$\dot{x}_3 = \frac{gm_p}{m_c}x_2 + \frac{1}{m_c}u_1, \tag{5.24}$$

$$\dot{x}_4 = \frac{g(m_c+m_p)}{lm_c}x_2 + \frac{1}{lm_c}u_1 + \frac{1}{lm_p}\lambda_1 - \frac{1}{lm_p}\lambda_2, \tag{5.25}$$

$$0 \leq \lambda_1 \perp lx_2 - x_1 + \frac{1}{k_1}\lambda_1 + d \geq 0,$$

$$0 \leq \lambda_2 \perp x_1 - lx_2 + \frac{1}{k_2}\lambda_2 + d \geq 0,$$

where $k_1 = k_2 = 10$ are stiffness parameters of the soft walls, $g = 9.81$ is the gravitational acceleration, $m_p = 0.1$ is the mass of the pole, $m_c = 1$ is the mass of the cart, $l = 0.5$ is the length of the pole, and $d = 0.1$ represents where the walls are. Observe that the model has absolutely continuous solutions following Proposition 8. For this model, we solve the feasibility problem (5.17) and find a controller of the form $u(x, \lambda) = Kx + L\lambda$ that regulates the model to the origin with $K = \begin{bmatrix} 3.69 & -46.7 & 3.39 & -5.71 \end{bmatrix}$ and $L = \begin{bmatrix} -13.98 & 13.98 \end{bmatrix}$. The algorithm succeeded in finding a feasible controller in 0.72 seconds. Additionally, we have tried to find a pure state feedback controller ($L = 0$) and, as formulated, failed to find such a controller for 1000 trials starting from different initial conditions. Due to the non-convexity of the BMI, this does not guarantee that such a controller-Lyapunov function pair does not exist, but demonstrates that the optimization problem is harder to solve in more conservative settings.

As a comparison, consider an LQR controller with penalty on the state $Q = 100I$ and penalty on the input $R = 1$ which is given as $K_{\text{LQR}} = \begin{bmatrix} 10 & -91.77 & 16.28 & -22.69 \end{bmatrix}$. Both contact-aware and LQR controllers are tested on the nonlinear plant for 100 initial conditions where $x_2(0) = 0$, and $x_1(0), x_3(0), x_4(0)$ are uniformly distributed $(10x_1(0), \frac{1}{4}x_3(0), x_4(0) \sim U[-1, 1])$. The LQR controller was successful only 31% of the time, whereas our contact-aware policy was successful 87% of the time. In Figure 5.3, an example is presented where both LQR and contact-aware policy start from the same initial conditions and LQR fails whereas our policy is successful. Our method

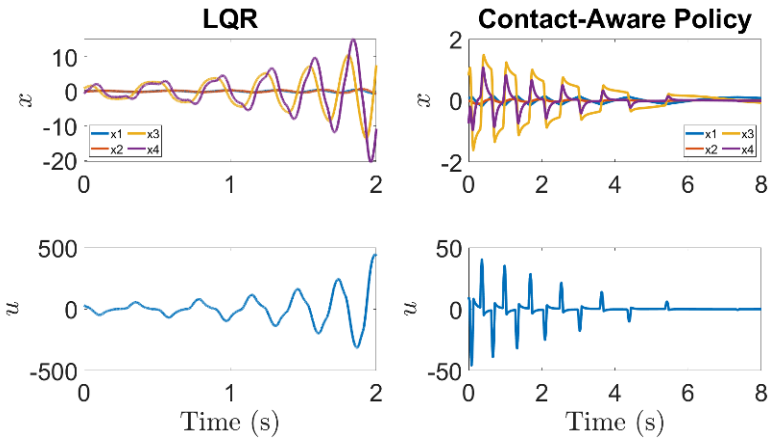

Figure 5.4: Comparison of the LQR controller and contact-aware policy for $k_1 = k_2 = 100$. The LQR controller fails to stabilize the system whereas contact-aware policy is successful.

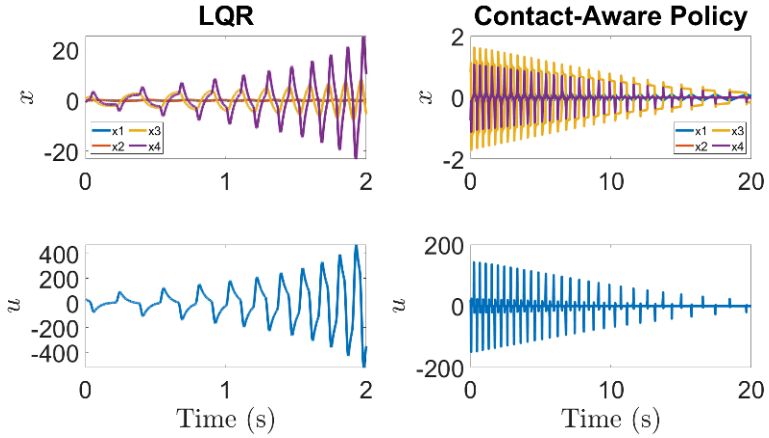

Figure 5.5: Comparison of the LQR controller and contact-aware policy for $k_1 = k_2 = 1000$. The LQR controller fails to stabilize the system whereas contact-aware policy is successful.

is compared with LQR, which uses linearization and thus ignores potential contact events. Our controller is a contact-aware analogue to non-switching state-feedback controllers such as LQR. In our comparison, the LQR controller acts as a stand-in for these methods which do not explicitly consider the non-smooth structure of the system.

Then, two different cases with higher stiffness values are explored. First, a controller is designed for

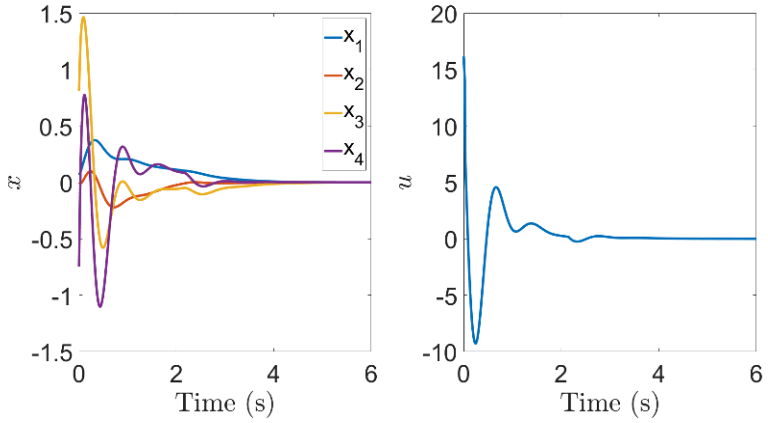

Figure 5.6: Performance of tactile feedback controller with damping. Contact-aware policy successfully stabilizes the nonlinear plant.

the case where $k_1 = k_2 = 100$ with the controller gains:

$$K_{100} = \begin{bmatrix} 3.69 & -48.78 & 2.36 & -9.96 \end{bmatrix},$$
$$L_{100} = \begin{bmatrix} -14.14 & 14.14 \end{bmatrix}.$$

The performance of the contact-aware controller is demonstrated against the previously designed LQR on the nonlinear plant in Figure 5.4.

One more set of experiments is presented where $k_1 = k_2 = 1000$ and a controller is designed with the gains:

$$K_{1000} = \begin{bmatrix} 0.45 & -40.23 & 0.86 & -25.50 \end{bmatrix},$$
$$L_{1000} = \begin{bmatrix} -14.14 & 14.14 \end{bmatrix}.$$

The performance of the contact-aware controller is demonstrated against the previously designed LQR on the nonlinear plant in Figure 5.5. Notice that it is possible to design controllers for stiffer contacts, but stiff, near-impulsive contact events resolve very quickly and that measurement and rapid response may not be practical.

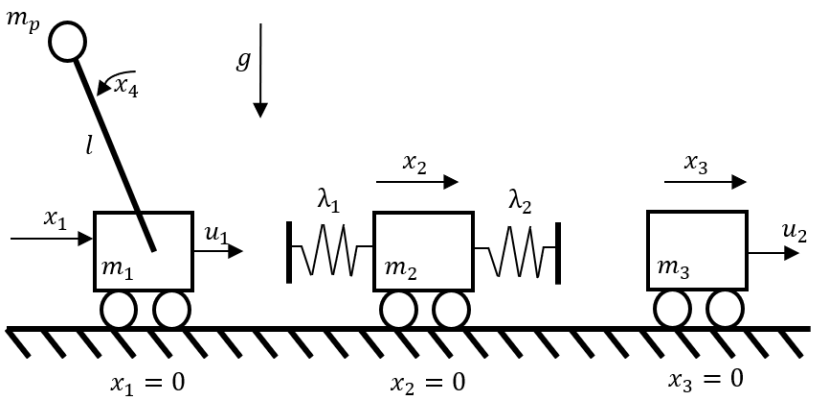

Figure 5.7: Regulation of carts to their respective origins without observation of the middle cart.

Next, inspired by (Brogliato, 2003), consider the case where damping term, b, is nonzero. The system dynamics are modeled as in (5.22)-(5.25) with the following complementarity constraints:

$$0 \leq \lambda_1 \perp -kx_1 + klx_2 - bx_3 + blx_4 + kd + \lambda_1 + \gamma_1 \geq 0,$$

$$0 \leq \gamma_1 \perp Mx_1 - Mlx_2 - Md + \gamma_1 \geq 0,$$

$$0 \leq \lambda_2 \perp kx_1 - lkx_2 + bx_3 - blx_4 + kd + \lambda_2 + \gamma_2 \geq 0,$$

$$0 \leq \gamma_2 \perp -Mx_1 + Mlx_2 - Md + \gamma_2,$$

where a large positive constant $M = 1000$ is introduced and the slack variables γ_1 and γ_2 to capture the affect of damper. As M increases the model approximates the spring-damper model (Brogliato, 2003) better. In this model, consider $b = 1$, $k = 10$, $m_p = m_c = 1$, $g = 9.81$, $l = 0.5$, $d = 0.1$. For this model, we solve the feasibility problem (5.17) and find a controller of the form $u(x, \lambda) = Kx + L\lambda$ that regulates the model to the origin with $K = \begin{bmatrix} 8.47 & -64.54 & 10.36 & -9.69 \end{bmatrix}$ and $L = \begin{bmatrix} -4.8 & 0 & 4.76 & 0 \end{bmatrix}$ in 384 seconds. The performance of the controller is demonstrated on the nonlinear plant in Figure 5.6.

5.5.2. Partial State Feedback

Consider a model that consists of three carts on a frictionless surface as in Figure 5.7. The cart on the left is attached to a pole and the cart in the middle makes contact via soft springs. In this

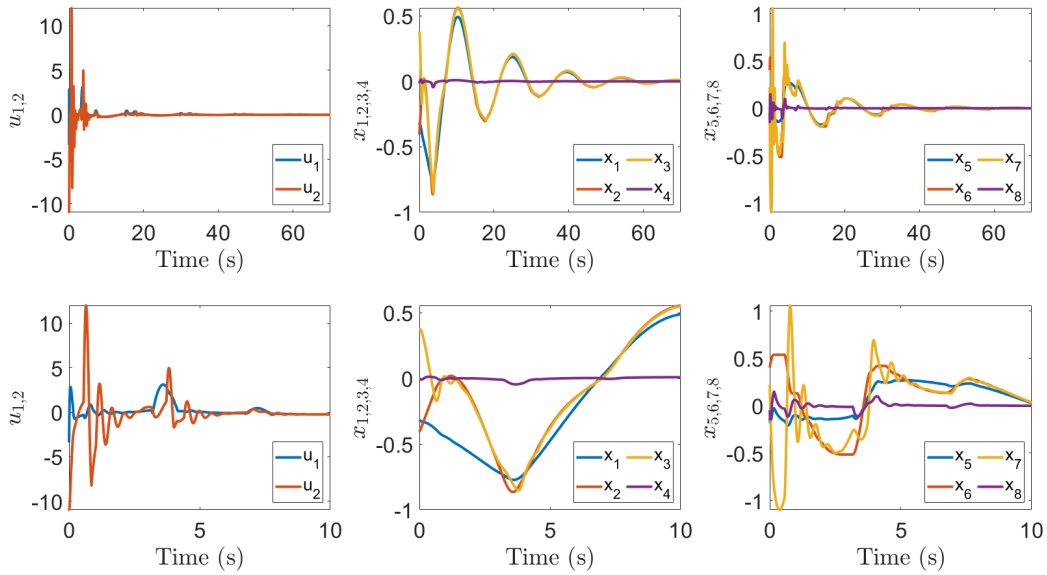

Figure 5.8: Simulation with contact-aware policy for partial state-feedback example. The plots on the top row show the input and the state variables $(u(t), x(t))$ for the time interval $t = [0\ 60]$. Second row demonstrates the time interval $t = [0\ 10]$ for the same initial condition.

model, a spring only becomes active if the distance between the outer block and the block in the middle is less than some threshold. Here, x_1, x_2, x_3 represent the positions of the carts and x_4 is the angle of the pole. The corresponding LCS is

$$
\begin{aligned}
\ddot{x}_1 &= \frac{g m_p}{m_1} x_4 + \frac{1}{m_1} u_1 - \frac{1}{m_1} \lambda_1, \\
\ddot{x}_2 &= \frac{\lambda_1}{m_2} - \frac{\lambda_2}{m_2}, \\
\ddot{x}_3 &= \frac{\lambda_2}{m_3} + \frac{u_2}{m_3}, \\
\ddot{x}_4 &= \frac{g(m_1 + m_p)}{m_1 l} x_4 + \frac{u_1}{m_1 l} - \frac{1}{m_1 l} \lambda_1, \\
0 &\leq \lambda_1 \perp x_2 - x_1 + \frac{1}{k_1} \lambda_1 \geq 0, \\
0 &\leq \lambda_2 \perp x_3 - x_2 + \frac{1}{k_2} \lambda_2 \geq 0,
\end{aligned}
$$

where the masses of the carts are $m_1 = m_2 = m_3 = 1$, $g = 9.81$ is the gravitational acceleration, $m_p = 1.5$ is the mass of the pole, $l = 0.5$ is the length of the pole, and $k_1 = k_2 = 20$ are stiffness

parameters of the springs. Observe that we have control over the outer blocks, but do not have any control over the block in the middle. Additionally, it is assumed that the middle block is not observed, and one can only observe the outer blocks and the contact forces. Notice that the model has absolutely continuous solutions following Proposition 8.

For this example, the feasibility problem (5.17) takes 9.3 seconds to solve with a controller of the form $u(x, \lambda) = Kx + L\lambda$ where

$$K = \begin{bmatrix} -2.8 & 0 & 6.6 & -263.1 & 6.4 & 0 & -2.1 & -30.2 \\ 11.5 & 0 & -12.1 & 12.1 & 2.6 & 0 & -4.7 & 6.6 \end{bmatrix},$$

$$L = \begin{bmatrix} -3.7 & -0.6 \\ -0.6 & 7.2 \end{bmatrix}.$$

Notice that we enforce sparsity on the controller K and do not use any feedback from the state x_2 or its derivative $\dot{x}_2$. This example demonstrates that tactile feedback can be used in scenarios where full state information is lacking and also impact events can be used in order to stabilize the system. In Figure 5.8, the performance of the controller is demonstrated.

5.5.3. Acrobot with Soft Joint Limits

As a third example, consider the classical underactuated acrobot, a double pendulum with a single actuator at the elbow (see (Murray and Hauser, 1991) for the details of the acrobot dynamics). Additionally, soft joint limits are added to the model. Hence we consider the model in Figure 5.9:

$$\dot{x} = Ax + Bu + D\lambda,$$

where $x = (\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2)$, $\lambda = (\lambda_1, \lambda_2)$, and $D = \begin{bmatrix} 0_{2\times 2} \\ M^{-1}J^T \end{bmatrix}$ with $J^T = \begin{bmatrix} -1 & 1 \\ 0 & 0 \end{bmatrix}$. For this model, the masses of the rods are $m_1 = 0.5$, $m_2 = 1$, the lengths of the rods are $l_1 = 0.5$, $l_2 = 1$, and the gravitational acceleration is $g = 9.81$. The soft joint limits are modeled using the following

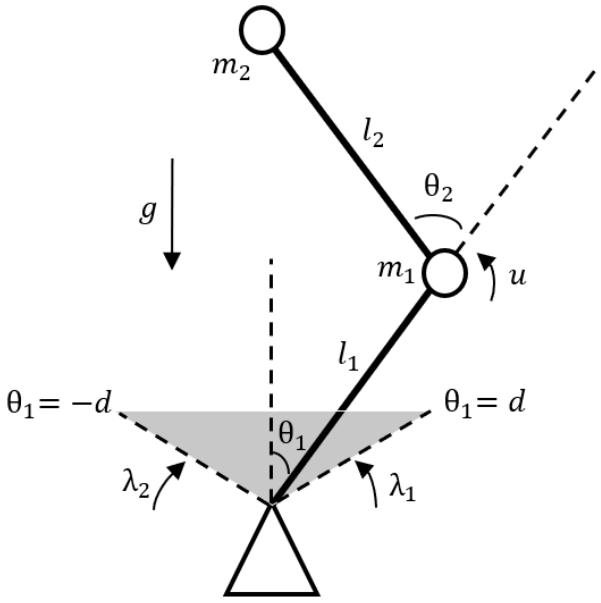

Figure 5.9: Acrobot with soft joint limits.

complementarity constraints:

$$0 \leq d - \theta_1 + \frac{1}{k}\lambda_1 \perp \lambda_1 \geq 0,$$
$$0 \leq \theta_1 + d + \frac{1}{k}\lambda_2 \perp \lambda_2 \geq 0,$$

where $k = 1$ is the stiffness parameter and $d = 0.2$ is the angle that represents the joint limits in terms of the angle θ_1. Observe that the model has absolutely continuous solutions, $x(t)$, following Proposition 8 since F is a P-matrix.

For this example, we solve the feasibility problem (5.17) and obtain a controller of the form $u(x, \lambda) = Kx + L\lambda$ in 1.18 seconds with $K = \begin{bmatrix} 73.07 & 38.11 & 30.41 & 18.95 \end{bmatrix}$ and $L = \begin{bmatrix} -4.13 & 4.13 \end{bmatrix}$. For comparison, we also designed an LQR controller for the linear system where the penalty on the state is $Q = 100I$ and the penalty on the input is $R = 1$ which is given as $K_{\text{LQR}} = \begin{bmatrix} 1476.3 & 851.68 & 548.81 & 334.43 \end{bmatrix}$. 100 trials were made on the nonlinear plant where initial conditions were sampled according to $x_1(0) = x_2(0) = 0$ and $x_3(0), x_4(0) \sim U[-0.05, 0.05]$.

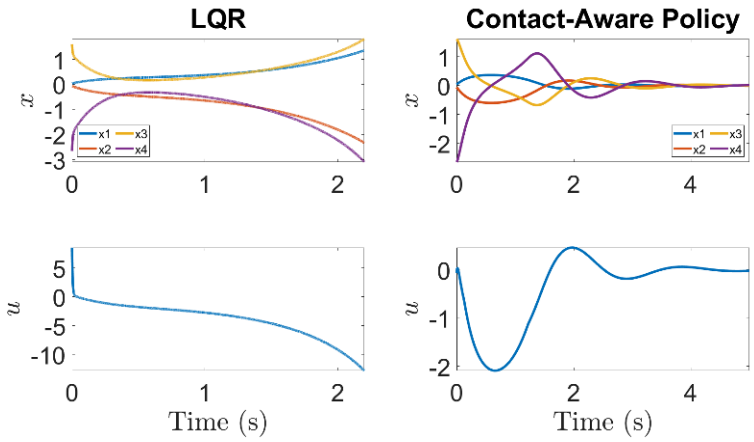

Figure 5.10: Simulation of LQR and contact-aware policy starting from the same initial condition for the acrobot with soft joint limits example. LQR is unstable whereas contact-aware policy is successful.

Out of these 100 trials, LQR was successful only 49% of the time whereas our design was successful 87% of the time. In Figure 5.10, a case where LQR fails and contact-aware policy is successful is presented.

5.5.4. Box with Friction

Consider a quasi-static model of a box on a surface, as in Figure 5.11, where μ is the coefficient of friction between the box and the ground. Newtons's second law is approximated with a force balance equation with Coulomb friction and damping. The goal is to regulate the box to the center. This simple model serves as an example where F is not a P-matrix and the complementarity constraints have a dependency on the input u ($H \neq 0$). Here, x is the position of the box, u is the input, λ_+ is the positive component of the friction force, λ_- is the negative component of the friction force and γ is the slack variable:

$$\begin{aligned}
&\alpha\dot{x} = u + \lambda_+ - \lambda_-, \\
&0 \leq \gamma \perp \mu mg - \lambda_+ - \lambda_- \geq 0, \\
&0 \leq \lambda_+ \perp \gamma + u + \lambda_+ - \lambda_- \geq 0, \\
&0 \leq \lambda_- \perp \gamma - u - \lambda_+ + \lambda_- \geq 0,
\end{aligned}$$

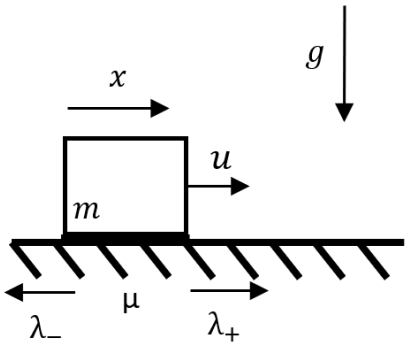

Figure 5.11: Regulation task of a box standing on a surface with Coulomb friction.

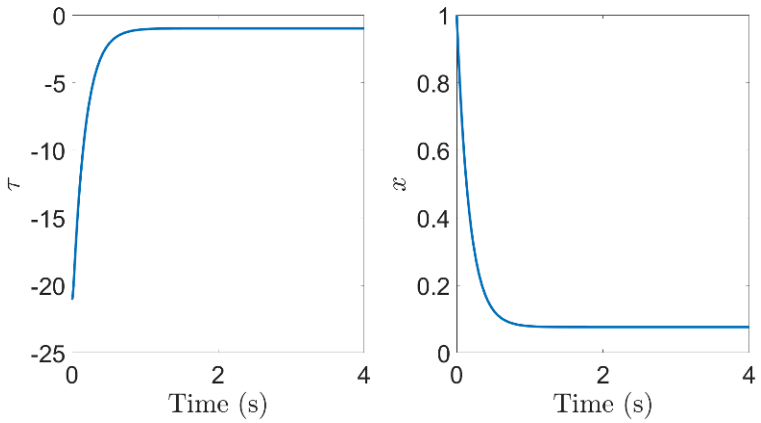

Figure 5.12: Simulation of box with friction example. The equilibrium is Lyapunov stable and the state trajectory, $x(t)$ does not reach origin because of stiction.

where $m = 1$ is the mass of the box, $g = 9.81$ is the gravitational acceleration $\mu = 0.1$ is the friction coefficient, and $\alpha = 4$ is the damping coefficient. Input delay is modeled with the low-pass filter:

$$\alpha \dot{x} = \tau + \lambda_+ - \lambda_-,$$

$$\dot{\tau} = \kappa(u - \tau),$$

$$0 \leq \gamma \perp \mu m g - \lambda_+ - \lambda_- \geq 0,$$

$$0 \leq \lambda_t^+ \perp \gamma + \tau + \lambda_+ - \lambda_- \geq 0,$$

$$0 \leq \lambda_t^- \perp \gamma - \tau - \lambda_+ + \lambda_- \geq 0,$$

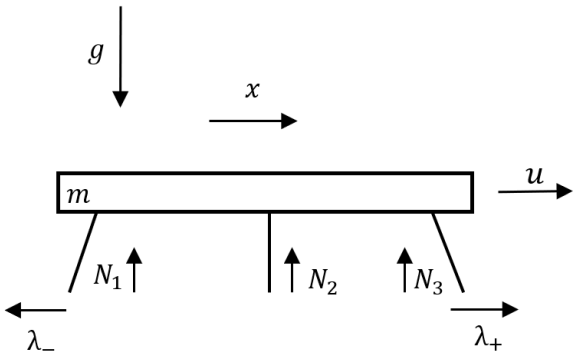

Figure 5.13: Regulation task of a 3 legged table.

where $\kappa = 100$. Since F is not a P-matrix, we use Algorithm 6 and find $W = [0\ 1\ -1]$ such that $W\text{SOL}(q, F)$ is a singleton for all q in 8.09 seconds. For this example, W shows that the net friction force, $\lambda_+ - \lambda_-$ is always unique. Notice that the x-trajectory, $x(t)$ is unique, but the λ-trajectory, $\lambda(t)$ is not. Note that the closed-loop system has absolutely continuous solutions following Proposition 8 since it is enforced that $L\text{SOL}(Ex + c, F)$ is a singleton.

We can solve the feasibility problem in (5.17) in 22 seconds and find a controller of the form $u(x, \lambda) = Kx + L\lambda$ such that the system is Lyapunov stable with $K = \begin{bmatrix} -10.58 \end{bmatrix}$ and $L = \begin{bmatrix} 0 & 0.7 & -0.7 \end{bmatrix}$. In Figure 5.12, the performance of the controller is demonstrated.

5.5.5. Three Legged Table

We consider a three legged table on a surface with Coulomb friction as in Figure 5.13. In this model, the coefficient of friction values (μ_1, μ_2, μ_3) are different for each leg of the table. The normal forces at the legs of the table are denoted by (N_1, N_2, N_3) and sum of the normal forces are equal to the mass times gravitational acceleration, mg. The net friction force is unique in static situations but it is non-unique during sliding since individual normal forces, N_i, are non-unique. The task is regulating the three legged table to the center. Newton's second law is approximated with a force balance equation with Coulomb friction and damping as in the previous example. In this model x is the position of the box, τ is the output of the low-pass filter, λ_+ is the positive component of the friction force, λ_- is the negative component of the friction force, γ is the slack variable, u is the force applied to the table:

$$\alpha\dot{x} = \tau + \lambda_+ - \lambda_-, \tag{5.26}$$

$$\dot{\tau} = \kappa(u - \tau), \tag{5.27}$$

$$0 \leq \gamma \perp \mu_1 N_1 + \mu_2 N_2 + \mu_3 N_3 - \lambda_+ - \lambda_- \geq 0, \tag{5.28}$$

$$0 \leq \lambda_+ \perp \gamma + \tau + \lambda_+ - \lambda_- \geq 0, \tag{5.29}$$

$$0 \leq \lambda_- \perp \gamma - \tau - \lambda_+ + \lambda_- \geq 0, \tag{5.30}$$

$$N_1 + N_2 + N_3 = mg, \tag{5.31}$$

$$N_1, N_2, N_3 \geq 0, \tag{5.32}$$

where $(\mu_1, \mu_2, \mu_3) = (0.1, 0.5, 1)$ are the coefficient of friction parameters for the legs of the table, $m = 1$ is the mass of the box, $g = 9.81$ is the gravitational acceleration, $\alpha = 4$ is the damping coefficient, and $\kappa = 100$ is the filter coefficient. The constraints (5.31) and (5.32) are exchanged with:

$$0 \leq N_1 \perp -mg + N_1 + N_2 + N_3 \geq 0,$$

$$0 \leq N_2 \perp -mg + N_1 + N_2 + N_3 \geq 0,$$

$$0 \leq N_3 \perp -mg + N_1 + N_2 + N_3 \geq 0,$$

to be consistent with the framework. Note that extending the framework to LCS models with additional equality and inequality constraints as in (5.26)-(5.32) is straightforward but it is omitted for brevity.

After using Algorithm 1, we find that $W = [0\ 0\ 0\ 1\ 1\ 1]$ in 242.8 minutes. Observe that one can solve a smaller sized polynomial optimization that only includes N_1, N_2, N_3 in 3.08 seconds as the variables are decoupled from $\gamma, \lambda_+, \lambda_-$ and reach the same result. W shows that $N_1 + N_2 + N_3$ is unique, as expected since $N_1 + N_2 + N_3 = mg$. Note that $W\lambda = mg$ is a constant and the Lyapunov function (5.6) reduces to a common Lyapunov function. Based on the structure of W, unlike the previous example, the net force $\lambda_+ - \lambda_-$ is not unique which is also expected due to the non-unique nature of normal forces. Notice that both the x-trajectory, $x(t)$ and λ-trajectory, $\lambda(t)$

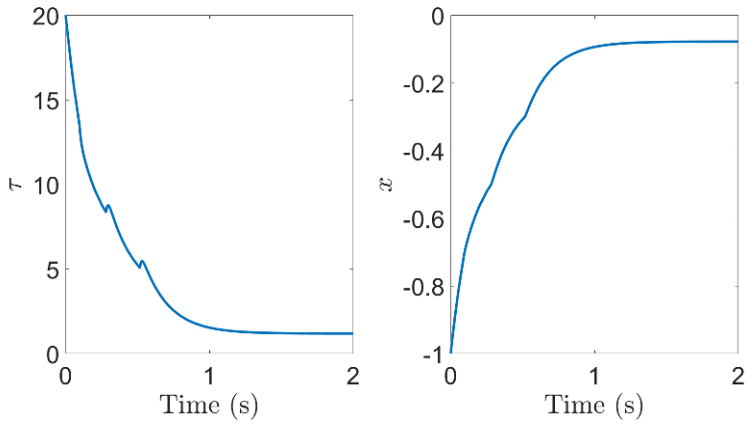

Figure 5.14: Simulation of three legged table example for the normal forces given as $N(t) = [4.0910, 4.1195, 1.5995]$ for $t \in [0, 0.2992)$, $N(t) = [5.4033, 3.1206, 1.2861]$ for $t \in [0.2992, 0.5455)$ and $N(t) = [9.4866, 0.1770, 0.1464]$ for $t \in [0.5455, \infty)$.

are non-unique. The model has absolutely continuous solutions, $x(t)$ and $\tau(t)$, for fixed N_1, N_2, N_3 following Proposition 8. For this example, we only consider the case where $N_1(t), N_2(t), N_3(t)$ are bounded piecewise constant functions with finitely many pieces hence $x(t)$ and $\tau(t)$ are absolutely continuous.

The feasibility problem (5.17) is solved in 19 seconds and a controller of the form $u(x, \lambda) = Kx + L\lambda$ is found such that the origin is Lyapunov stable with $K = \begin{bmatrix} -20.75 \end{bmatrix}$ and $L = \begin{bmatrix} 0 & 0.36 & -0.36 & 0 & 0 & 0 \end{bmatrix}$. The trajectories of the closed loop system are always absolutely continuous since we enforce that $LSOL(Ex + c, F)$ is a singleton for fixed, arbitrary N_1, N_2 and N_3. In Figure 5.14, observe that the force applied to the system (τ) is continuous even though u is not, due to the low-pass filter. The origin is Lyapunov stable and the trajectory does not reach origin because of stiction.

5.5.6. 2D Simple Manipulation

Consider a quasi-static model of a box on a surface with friction parameter μ and two robotic arms that can interact with the box as in Figure 5.15. Similar to the previous example, the force balance equation is used with Coulomb friction and damping to model the dynamics of the box. The velocity of the manipulators can be controlled directly with delayed inputs τ_1 and τ_2. In this

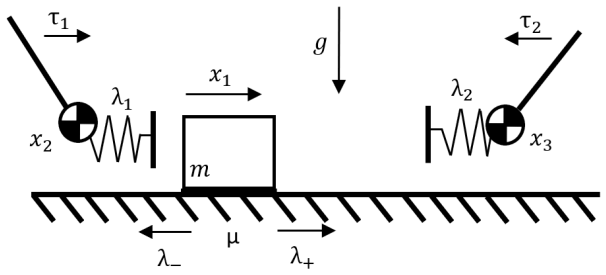

Figure 5.15: 2D manipulation task where the goal is to regulate the position of the box on a surface with friction.

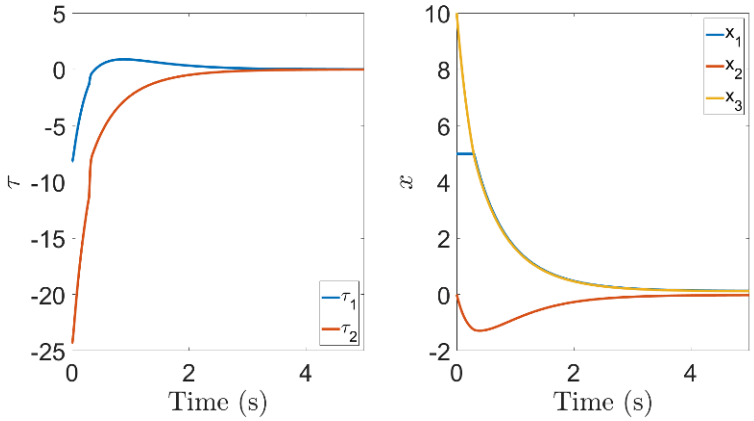

Figure 5.16: Simulation results for 2D simple manipulation example. The forces applied to the box (τ) are smooth even though u is not, due to the low-pass filter model.

model x_1, x_2, x_3 represent the positions of the box, the left manipulator and the right manipulator respectively. The contact forces λ_1 and λ_2 are non-zero if and only if the distance between the manipulators and the box is less than some threshold. The friction force consists of a positive component λ_+ and a negative component λ_-. We assume that we can not observe anything related to the box except the contact force between the manipulators and the box. The task is to regulate the box to the origin using the model:

$$
\begin{aligned}
&\alpha\dot{x}_1 = \lambda_1 - \lambda_2 + \lambda_+ - \lambda_-,\\
&\dot{x}_2 = \tau_1,\\
&\dot{x}_3 = \tau_2,\\
&\dot{\tau}_1 = \kappa(u_1 - \tau_1),\\
&\dot{\tau}_2 = \kappa(u_2 - \tau_2),\\
&0 \le \lambda_1 \perp x_1 - x_2 + \frac{1}{k}\lambda_1 \ge 0,\\
&0 \le \lambda_2 \perp x_3 - x_1 + \frac{1}{k}\lambda_2 \ge 0,\\
&0 \le \gamma \perp \mu mg - \lambda_+ - \lambda_- \ge 0,\\
&0 \le \lambda_+ \perp \gamma + \lambda_1 - \lambda_2 + \lambda_+ - \lambda_- \ge 0,\\
&0 \le \lambda_- \perp \gamma - \lambda_1 + \lambda_2 - \lambda_+ + \lambda_- \ge 0,
\end{aligned}
$$

where $\kappa = 100$, $\mu = 0.1$, $m = 1$, $g = 9.81$, $\alpha = 1$, and $k = 100$. Since, F is not a P-matrix, we can construct the W matrix using the result in Example 5.5.4 and obtain

$$
W = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}.
$$

Observing W, the net friction force, $\lambda_+ - \lambda_-$, is unique. The contact forces between the manipulators and the box, λ_1, λ_2 are also unique. Note that the closed-loop system has absolutely continuous solutions following Proposition 8 since it is enforced that $L\mathrm{SOL}(Ex + c, F)$ is a singleton.

Then we solve the optimization problem to find a controller that asymptotically stabilizes the system to a small ball around the origin $\mathcal{B} = \{x : x^T x \le 0.1\}$. The optimization problem finds a result in 7.06 minutes and a controller of the form $u(x, \lambda) = Kx + L\lambda$ that stabilizes the system is obtained

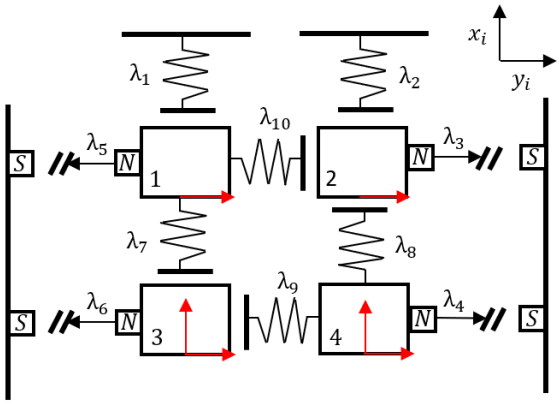

Figure 5.17: Four carts example. The inputs that are applied to carts are represented by the red arrows.

with

$$K = \begin{bmatrix} 0 & -2.33 & -0.82 \\ 0 & -0.89 & -2.44 \end{bmatrix}, \qquad L = \begin{bmatrix} -0.26 & 0.06 & 0 & 0 & 0 \\ -0.06 & 0.27 & 0 & 0 & 0 \end{bmatrix}.$$

This example shows that the contact-aware policy can be used for systems with non-unique contact forces, e.g., quasi-static friction, where we do not have full state information. In Figure 5.16, we demonstrate the performance of the controller.

5.5.7. Four Carts

As our last example, consider the system in Figure 5.17. Here, (x_i, y_i) gives the position of the cart i. We approximate Newton's second law with a force balance equation for each cart. The contact forces $\lambda_1, \lambda_2, \lambda_7, \lambda_8, \lambda_9$ and λ_{10} are soft contacts that are represented by the springs and are non-zero if the objects are closer than a threshold. The forces $\lambda_3, \lambda_4, \lambda_5$ and λ_6 approximate attractive magnetic forces between the carts and the walls and similarly are non-zero if the distance between the carts and the walls is less than a threshold. The red arrows represent the input forces that can be applied to carts. We model this system with $n = 8$ states, and $m = 10$ contacts where our goal is to show the performance of the proposed method on a high dimensional under-actuated example that

is unstable without any control action. The model parameters are $A = 0_{8\times 8}$, $c = 0_{10\times 1}$, $F = I_{10\times 10}$,

$$B = \begin{bmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} & 0_{4\times 3} \\ 0_{3\times 4} & I_{3\times 3} \end{bmatrix},$$

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \end{bmatrix},$$

$$E = \begin{bmatrix} 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Observe that the model has absolutely continuous solutions following Proposition 8. A controller

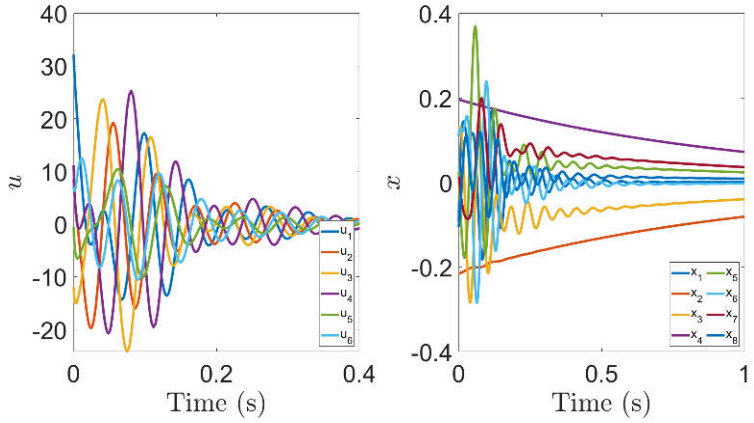

Figure 5.18: Simulation of four carts example. The state trajectory, $x(t)$, asymptotically converges to the origin.

of the form $u(x, \lambda) = Kx + L\lambda$ that stabilizes the system can be found. For this example, and higher dimensional examples in general, the initialization of the K and L matrices have a significant affect on the success of the algorithm. We initialized elements of K with sampling from uniform distribution ($K_{ij} \sim U[-100, 0]$). For one successful case, the algorithm terminates in 6 minutes and 58 seconds, but we also needed to run the algorithm approximately 20 hours with random seeds to obtain a successful result. In Figure 5.18, we present the performance of the controller.

5.6. Experimental Validation

We demonstrate our contact-aware feedback controller on an experimental cart-pole system with soft walls shown in Figure 5.19, replicating the system from Section 5.5.1. To generate linear motion of the cart, a DC motor is used with a belt drive. Linear motion of the cart was driven by torque-controlled DC motor and incremental encoders measured the positions of the cart and pole. We added soft walls and tactile sensing capabilities; open-cell polyurethane foam was used for the walls, and a '*Flintec PC42 Single Point Load Cell*' was used to measure the force exerted on the walls by the pole. Alternatively, sensors could have been placed on the pole itself.

5.6.1. System Model

We take the gravitational acceleration as $g = 9.81$, mass of the pole as $m_p = 0.35$, mass of the cart

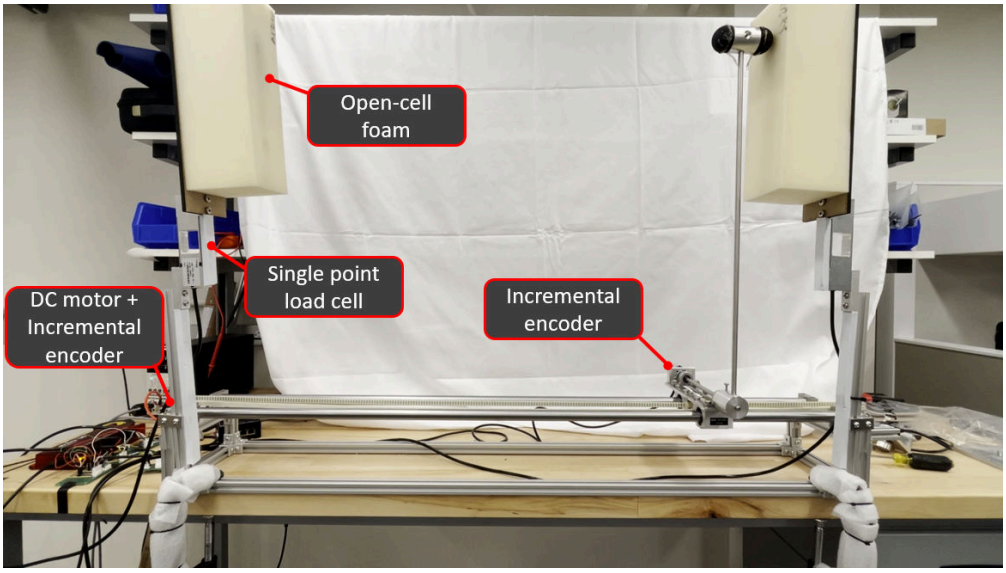

Figure 5.19: Experimental setup for cart-pole with soft walls.

as $m_c = 0.978$, length of the center of mass location as $l_{\text{CoM}} = 0.4267$, the distance to the walls as $d = 0.35$ and and length of the pole as $l_p = 0.6$. After experimental trials, we identified the spring constant $k = 700$ and neglect the damping term ($b = 0$). The parameters of the LCS model are:

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 3.51 & 0 & 0 \\ 0 & 22.2 & 0 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0 \\ 0 \\ 1.02 \\ 1.7, \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 4.7619 & -4.7619 \end{bmatrix}, \; E = \begin{bmatrix} -1 & 0.6 & 0 & 0 \\ 1 & -0.6 & 0 & 0, \end{bmatrix},$$

$$F = \begin{bmatrix} 0.0014 & 0 \\ 0 & 0.0014 \end{bmatrix}, \; c = \begin{bmatrix} 0.35 \\ 0.35 \end{bmatrix}.$$

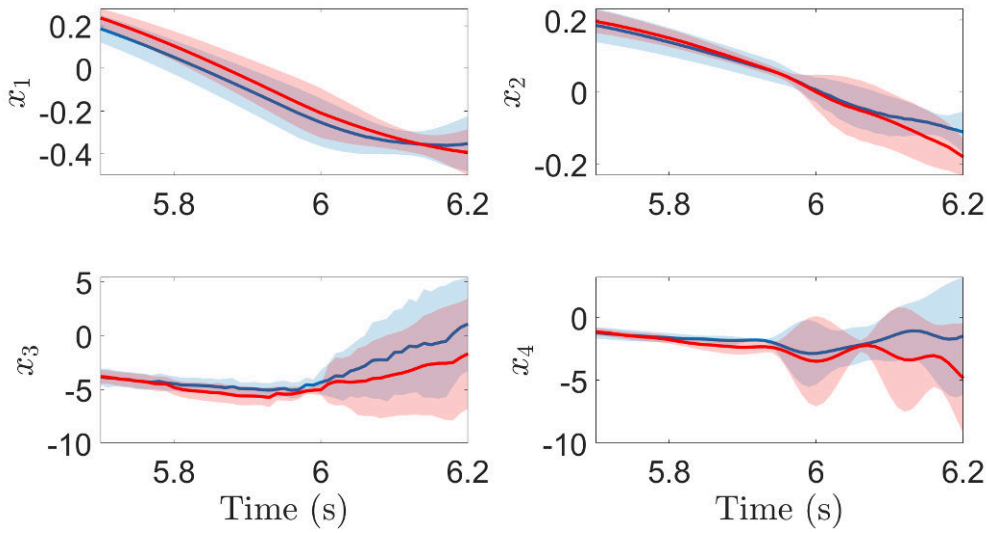

Figure 5.20: Experiment 1 - Trajectories around the impact event for all 6 trials. Blue represents contact-aware policy, red represents LQR, solid lines represent the respective means and shaded regions represent standard deviation. State distribution before the impact events are similar (≈ 2 cm difference in cart position and ≈ 1 degree difference in pole angle). The velocity of the cart (x_3) is significantly higher for contact-aware policy after the impact.

5.6.2. Experiments

Three sets of experiments are performed. For the first two, our method is compared against an LQR controller where $K_{\text{LQR}} = \begin{bmatrix} 3.16 & -40.78 & 4.3 & -7.67 \end{bmatrix}$. The contact-aware policy is designed such that $K = K_{\text{LQR}}$ is enforced and a contact gain matrix $L = \begin{bmatrix} -10.02 & 10.02 \end{bmatrix}$ is found. It may be impossible to find a contact gain L with a fixed K in general, but $K = K_{\text{LQR}}$ was enforced for a more fair comparison with LQR. If we let the BMI design search for both K and L, we can potentially get a better solution, though BMI enforces stability versus optimality. Then, we perform between 6 and 10 trial experiments depending on the setup, after which we observed deterioration of the experimental setup due to the violence of the impact events that occurred when the LQR controller failed to stabilize the system. While performing these experiments, sensor readings below 1 N are considered as 0 N to neglect the effect of oscillations that occur after the impact events. For the third experiment, we repeat Experiment 1 without thresholds on the sensor readings.

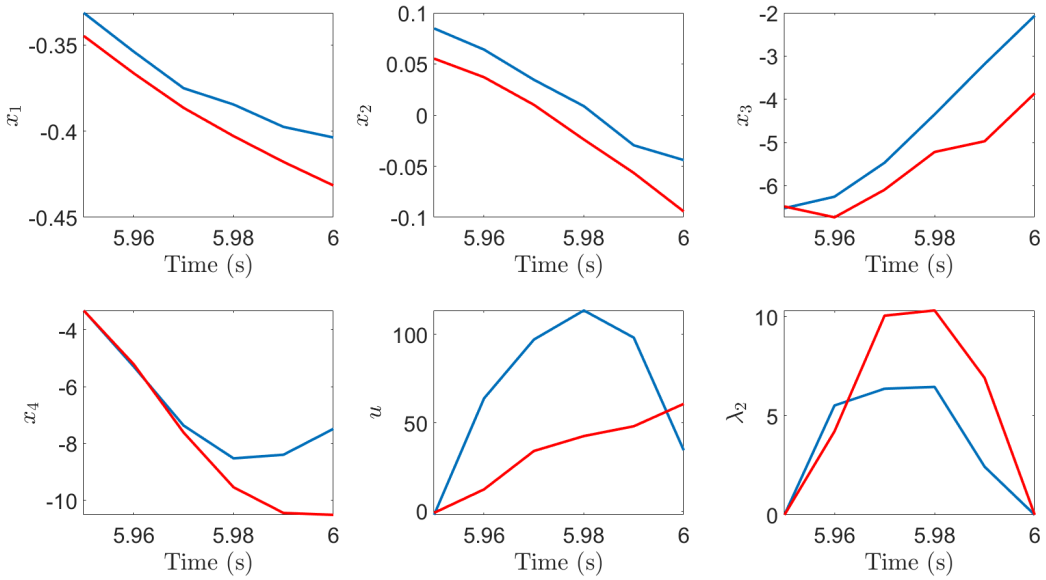

Figure 5.21: Experiment 1 - Blue represents the contact-aware policy and red represents LQR. Contact-aware policy (u) starts pushing the cart in the positive direction aggressively as soon as impact starts (λ_2) in order to catch the falling pole causing a big increase in the cart velocity (x_3). As a result of contact-aware policy, the angular speed of pole is closer to zero (x_4). The contact-aware policy also mitigates the impact (λ_2).

Experiment 1 We execute balancing controllers which attempt to stabilize the system to the origin and evaluate their performance by introducing large perturbations that lead to contact events. First, the system is started in the upright position at the right wall, $x_0^T = \begin{bmatrix} 0.35 & 0 & 0 & 0 \end{bmatrix}$. Then, a control input[9] is applied such that the pole impacts the left wall with high speed and close to upright position.

We repeated this experiment 6 times for both policies. In Figure 5.20, we demonstrate that the initial conditions for all trials (at $t = 5.7$) are similar. The mean difference is 0.05, 0.01, 0.01, 0.04 respectively for x_1, x_2, x_3, x_4. The LQR policy failed in all (0/6) of the trials whereas contact-aware policy was always successful (6/6). Since the policies are identical when not in contact, we focus our analysis and plots on the brief time windows ($t = [5.7, 6.2]$ as in Figure 5.20) which contain impact events. The contact-aware policy results in a significant increase in cart velocity (during the

[9]We apply the control input $u = K(x - x_s)$ where $x_s^T = \begin{bmatrix} 0 & 0.35 & 0 & 0 \end{bmatrix}$; hence, the cart moves towards the left wall; then, we switch back to $x_s^T = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$.

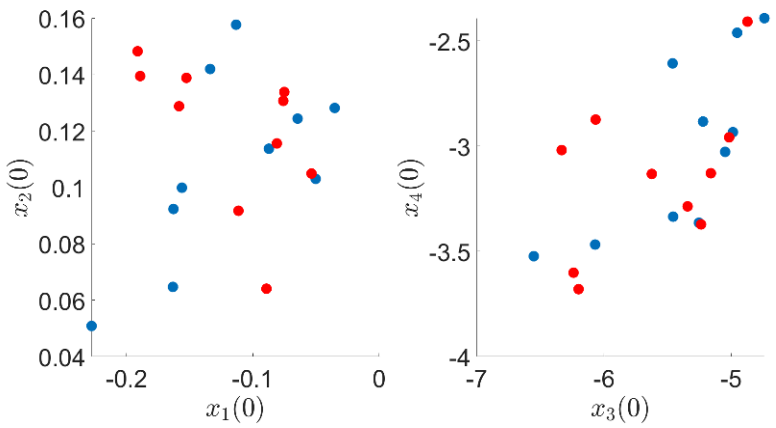

Figure 5.22: Experiment 2 - Distribution of initial conditions where blue represents contact-aware policy trials and red represents LQR trials.

impact event) in order to catch the falling pole (the mean cart velocity for contact-aware policy is 2.76 m/s higher than LQR at $t = 6.2$). Similarly, the mean angular velocity of the pole with LQR controller is -4.84 rad/s compared to -1.48 rad/s of contact-aware policy which also demonstrates that contact-aware policy is reacting better to the falling pole as expected.

We examine a specific trial (Figure 5.21) where the differences between the pre-impact states are 0.01, 0.02, 0.2 and 0.01 for x_1, x_2, x_3, x_4 respectively. Over the 50 ms impact period, change in the cart velocity with contact-aware controller is 2.2 m/s higher than LQR. Similarly, the change in angular velocity of the pole is 3.01 rad/s more than LQR. As shown in Figure 5.21, as soon as the impact event with the left wall starts the contact-aware policy tries to push the cart in the positive direction in order to catch the falling pole and stabilizes the system unlike LQR.

Experiment 2 In the first experiment, we created a consistent initial condition across all trials. Here, we introduce random perturbations to cover a broader range of initial conditions. As with the first experiment, the goal of the initialization process is to create conditions which initiate contact. First, the system is balanced at the origin. Then, we apply a control input[10] briefly to ensure that pole is close to impacting the left wall with a relatively high speed. Next, we apply a random input disturbance with uniform distribution $u_d \sim U[5, 10]$ for 100 ms. We repeat this experiment 10 times

[10]We apply the control input $u = K(x - x_s)$ where $x_s^T = \begin{bmatrix} 0 & 0.5 & 0 & 0 \end{bmatrix}$ for 0.5 seconds and switch back to $x_s^T = \begin{bmatrix} 0 & 0 & 0 & 0 \end{bmatrix}$.

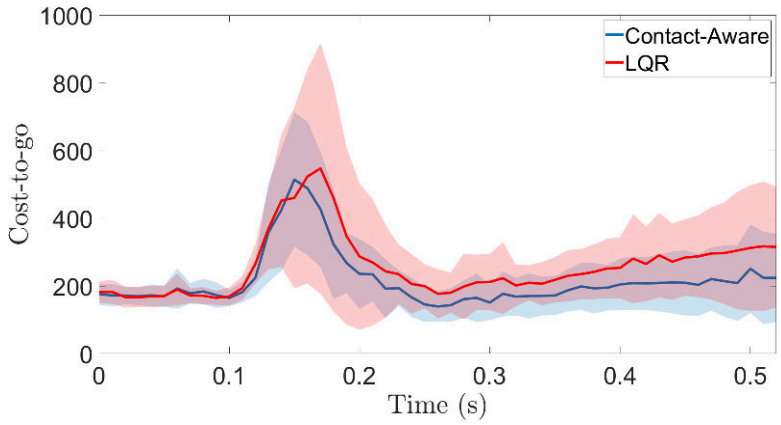

Figure 5.23: Experiment 2 - LQR cost-to-go for all trials during the impact event (impact events are aligned for all trials). Blue represents contact-aware policy, red represents LQR, solid lines represent the respective means and shaded regions represent standard deviation. Contact-aware cost-to-go surpasses LQR cost-to-go as the impact starts due to the aggressive tactile feedback but contact-aware policy ends up with a lower cost-to-go after the impact event.

for each LQR and contact-aware policy (with same seeds). After the random input disturbance is applied, the states are distributed as shown in Figure 5.22.

Out of the 10 trials, the LQR controller failed in 5/10 of the trials whereas the contact-aware policy was always successful. In Figure 5.23, we demonstrate that the contact-aware policy ends up with a lower cost-to-go than LQR after the impact event. Note that LQR cost-to-go is a useful metric, more so than the 2-norm, since it represents an approximate cost to complete the stabilization task.

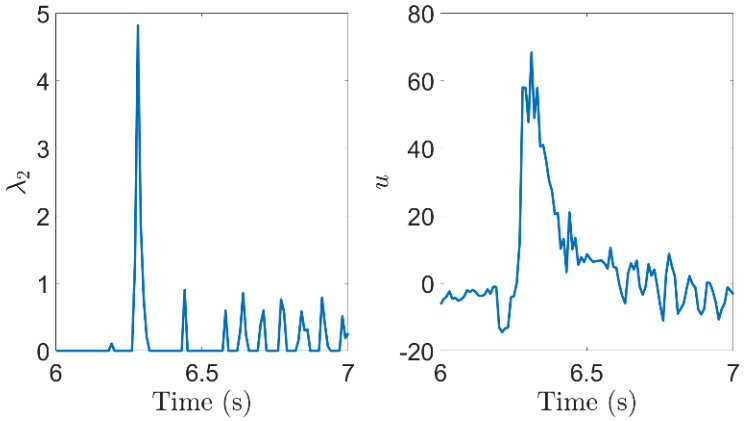

Figure 5.24: Experiment 3 - The oscillations in sensor readings (λ_2) that are caused by the impact event and the corresponding control action without heuristic-based thresholds.

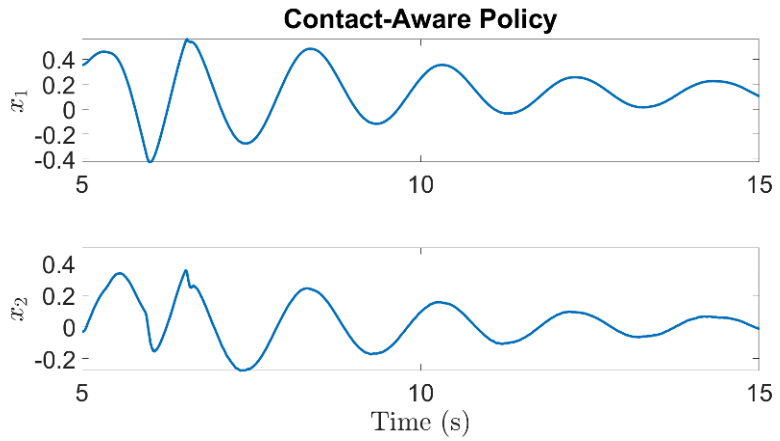

Figure 5.25: Experiment 3- Trajectory with contact-aware controller without any heuristic-based thresholds.

Experiment 3 After the impact event, the walls oscillate back and forth which causes oscillations in sensor readings as shown in Figure 5.24. In this experiment, we do not apply a threshold to the sensor readings. This enables pure feedback on sensor measurements, rather than heuristic-based thresholds or slow/inaccurate mode detection (where the state-of-the-art takes 4-5 ms (Bledt et al., 2018)) that purely hybrid approaches utilize. Hence, we repeat the procedure in Experiment 1 and observe that the controller is successful in stabilizing the system (Figure 5.25) even without heuristic-based thresholds. In Figure 5.26, model is simulated (as in Section 5.5.1) starting from the an initial condition obtained from the experiment and contact-aware policy stabilizes the system whereas LQR fails. This demonstrates that simulations capture the system qualitatively.

5.7. Conclusion and Discussion

In this work, we have introduced a controller that can utilize both state and force feedback. We have demonstrated that combining linear complementarity systems with such tactile feedback controllers might result in an algebraic loop, and discussed how one can break such algebraic loops.

We have proposed an algorithm for synthesizing contact-aware control policies for linear complementarity systems with possibly non-unique solutions. For soft contact models, we have shown that pure local, linear analysis was entirely insufficient and utilizing contact in the control design

is critical to achieve high performance. For systems with non-unique solutions, we have proposed a polynomial optimization program that can find matrices that map non-unique contact forces into a unique value, and used such mappings in our controller design algorithm. We have shown the effectiveness of our method on quasi-static friction models.

Furthermore, the proposed algorithm exploits the complementarity structure of the system and avoids enumerating the exponential number of potential modes, enabling efficient design of multi-contact control policies. Towards this direction, we have presented an example with eight states and ten contacts. In addition to incorporating tactile sensing into dynamic feedback, we provide stability guarantees for our design method and we have verified our method on an experimental setup.

The algorithm requires solving feasibility problems that include bilinear matrix inequalities and we have used PENBMI (Kočvara and Stingl, 2003). For the examples presented here, except the last one, the run time of the algorithm was short and we found solutions to the problems relatively quickly. On the other hand, it is important to note that for some parameter choices and initializations, the solver was unable to produce feasible solutions.

Interesting future work in this area will be using the controller presented here in physical exper-

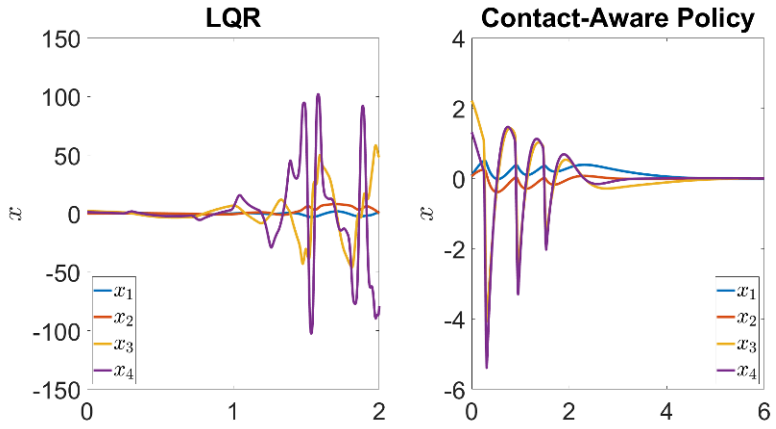

Figure 5.26: Experiment 3- Simulation results (as in Section 5.5.1) where we simulate forward from a state obtained from the experiment. Simulation captures the system response qualitatively as LQR is unstable and contact-aware policy is successful.

iments. In addition, we intend to extend these algorithms to more complex tasks. For example, quasi-static models (Halm and Posa, 2019) where the matrix F depends on the generalized coordinates q. Another direction is designing controllers for systems where there are bilinear terms $(x_i\lambda_j)$ in the dynamics, since we believe that bilinear terms are important when locally approximating a certain class of non-smooth systems. Also, works such as (Brogliato, 2016) draw connections between compliant and rigid contact models. Such approaches can help analyzing the controllers designed in this work for a range wider range of models. Lastly, it may be possible to increase the application of our method by utilizing non-monotonic and almost-decreasing Lyapunov functions (Morărescu and Brogliato, 2010).

CHAPTER 6

STABILITY ANALYSIS OF COMPLEMENTARITY SYSTEMS WITH NEURAL NETWORK CONTROLLERS

Parts of this chapter were previously published as parts of Alp Aydinoglu, Mahyar Fazlyab, Manfred Morari, and Michael Posa. Stability Analysis of Complementarity Systems with Neural Network Controllers. In Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control, pages 1–10, 2021a. DOI: https://dl.acm.org/doi/10.1145/3447928.3456651.

6.1. Introduction

In this chapter, we propose a method to analyze the stability of complementarity systems with neural network controllers. First, we introduce a method to represent neural networks with rectified linear unit (ReLU) activations as the solution to a linear complementarity problem. Then, we show that systems with ReLU network controllers have an equivalent linear complementarity system description. Using the LCS representation, we turn the stability verification problem into a linear matrix inequality (LMI) feasibility problem. We demonstrate the approach on several examples, including multi-contact problems and friction models with non-unique solutions.

6.2. Linear Complementarity Systems with Neural Network Controllers

In this section, we demonstrate that neural networks with rectified linear units (ReLU) have an equivalent LCP representation. Then, we show that an LCS combined with a neural network controller has an alternative complementarity system description.

Definition 19. *A ReLU neural network* $\phi\colon \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_\phi}$ *with* L *hidden layers is the composite function*

$$\phi(x) = (h_L \circ \lambda_{ReLU} \circ h_{L-1} \circ \ldots \circ \lambda_{ReLU} \circ h_0)(x), \tag{6.1}$$

where $\lambda_{ReLU}(x) = \max\{0, x\}$ *is the ReLU activation layer, and* $h_i(x) = \theta_i x + c_i$ *are the affine layers with* $\theta_i \in \mathbb{R}^{n_{i+1} \times n_i}$, $c_i \in \mathbb{R}^{n_{i+1}}$. *Here,* $n_i, 1 \leq i \leq L$ *denotes the number of hidden neurons in the* i*-th layer,* $n_0 = n_x$, *and* $n_{L+1} = n_\phi$. *We denote by* $n_t = \sum_{i=1}^{L} n_i$ *the total number of neurons.*

6.2.1. Representing ReLU Neural Networks as Linear Complementarity Problems

ReLU neural networks are piece-wise affine functions. Similarly, the linear complementarity problem describes a piece-wise affine function as shown in (3.1) as long as F is a P-matrix. In this section, we will explore the connection between two piece-wise affine representations.

It has been shown that ReLU neural networks can be represented with quadratic constraints (Raghunathan et al., 2018), (Fazlyab et al., 2019b). Now, our goal is to show the connection between these results and linear complementarity problems. We will describe a method to represent a multi-layered ReLU neural network as a linear complementarity problem exploring the cascade connections of complementarity problems (Brogliato, 2003).

First we consider a single ReLU unit $\lambda_{\text{ReLU}}(x) = \max(0, x)$ and its equivalent LCP representation.

Lemma 20. *(Heemels et al., 2000) Consider the following LCP for a given $x \in \mathbb{R}^d$:*

$$\begin{aligned} &\textit{find} && \lambda^{LCP} \in \mathbb{R}^d \\ &\textit{subject to} && \bar{y} = -x + \lambda^{LCP}, \\ & && 0 \leq \lambda^{LCP} \perp \bar{y} \geq 0, \end{aligned}$$

Then λ^{LCP} is unique and is given by $\lambda^{LCP} = \max\{0, x\}$.

Proof. If $x_i < 0$, then $\lambda_i^{\text{LCP}} = 0$ and if $x_i \geq 0$, then $\lambda_i^{\text{LCP}} = x_i$. □

Next, we consider a two layered neural network and transform it into an LCP using Lemma 20.

Example 21. *Consider a two layered network shown in Figure 6.1 where $\phi_{2\text{-}layer}(x) = \lambda_{ReLU} \circ h_1 \circ \lambda_{ReLU} \circ h_0(x)$ with $h_0(x) = x$ and $h_1(x) = \theta_2 x + \bar{c}_2$. An alternative representation is $\phi_{2\text{-}layer}(x) =$*

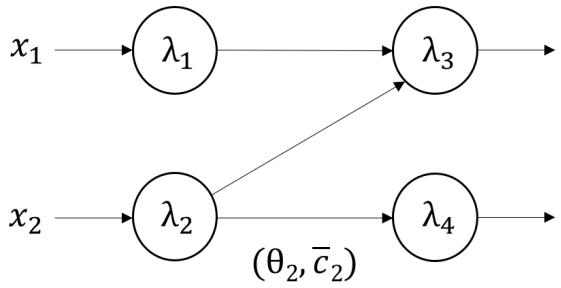

Figure 6.1: Two-layered neural network with ReLU activation functions.

$\begin{bmatrix} \lambda_3(x)^\top & \lambda_4(x)^\top \end{bmatrix}^\top$ *where*

$$\lambda_1(x) = \max\{0, x_1\}, \tag{6.2}$$

$$\lambda_2(x) = \max\{0, x_2\}, \tag{6.3}$$

$$\lambda_3(x) = \max\{0, \theta_2^{1,1}\lambda_1(x) + \theta_2^{1,2}\lambda_2(x) + \bar{c}_2^1\}, \tag{6.4}$$

$$\lambda_4(x) = \max\{0, \theta_2^{2,2}\lambda_2(x) + \bar{c}_2^2\}. \tag{6.5}$$

Here λ_i represents the output of the ith ReLU activation function, $\theta_i^{j,k}$ and $\bar{c}_i^j$ represent the coefficients of the affine function. Observe that the two-layered NN is equivalent to $\begin{bmatrix} \lambda_3 \\ \lambda_4 \end{bmatrix}$ *where λ is the unique solution of the following LCP:*

$$\begin{aligned}
&\textit{find} && \lambda \in \mathbb{R}^4 \\
&\textit{subject to} && \bar{y}_1 = -x_1 + \lambda_1, \\
&&& \bar{y}_2 = -x_2 + \lambda_2, \\
&&& \bar{y}_3 = -\theta_2^{1,1}\lambda_1 - \theta_2^{1,2}\lambda_2 - \bar{c}_2^1 + \lambda_3, \\
&&& \bar{y}_4 = -\theta_2^{2,2}\lambda_2 - \bar{c}_2^2 + \lambda_4, \\
&&& 0 \le \lambda_1 \perp \bar{y}_1 \ge 0,\ 0 \le \lambda_2 \perp \bar{y}_2 \ge 0, \\
&&& 0 \le \lambda_3 \perp \bar{y}_3 \ge 0,\ 0 \le \lambda_4 \perp \bar{y}_4 \ge 0.
\end{aligned}$$

Here, $\{\lambda_i\}_{i=1}^2$ *can be represented as* $\lambda_i = \max\{0, x_i\}$ *and* $\{\lambda_i\}_{i=3}^4$ *are as in* (6.4), (6.5) *after direct application of Lemma 20. Then, we conclude that* $\begin{bmatrix}\lambda_3\\ \lambda_4\end{bmatrix} = \phi_{\textit{2-layer}}$.

Now, we show that all neural networks of the form (6.1) have an equivalent LCP representation.

Lemma 22. *For any* x, *the ReLU neural network in* (6.1) *can be expressed as* $\phi(x) = \bar{D}\lambda(x) + \bar{z}$, *where* $\lambda(x)$ *is the unique solution of the following linear complementarity problem:*

$$\begin{aligned} &\textit{find} && \lambda \\ &\textit{subject to} && \bar{y} = \bar{E}x + \bar{F}\lambda + \bar{c}, \\ & && 0 \le \lambda \perp \bar{y} \ge 0, \end{aligned}$$

where $\bar{c} = \begin{bmatrix} -c_0 \\ -c_1 \\ \vdots \\ -c_{L-1} \end{bmatrix}$, $\bar{E} = \begin{bmatrix} -\theta_0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$, $\bar{F} = \begin{bmatrix} I & & & & & \\ -\theta_1 & I & & & & \\ 0 & -\theta_2 & I & & & \\ 0 & 0 & -\theta_3 & I & & \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & \dots & -\theta_{L-1} & I \end{bmatrix}$ *and* $\bar{z} = \bar{c}_L$, $\bar{D} = \begin{bmatrix} 0 & 0 & \dots & 0 & \theta_L \end{bmatrix}$ *where* $\bar{F}$ *is a P-matrix.*

Proof. First, we write $\lambda^\top = \begin{bmatrix} \lambda_0^\top & \lambda_1^\top & \cdots & \lambda_{L-1}^\top \end{bmatrix} \in \mathbb{R}^{1\times n_t}$ where each sub vector λ_i has the same dimension as $\bar{c}_i$. Next, we show that $\lambda_0(x) = \lambda_{\mathrm{ReLU}} \circ h_0(x)$. Observe that λ_0 is independent of $\lambda_1, \dots, \lambda_{L-1}$ since F is lower-triangular. Hence λ_0 is the unique element of the following LCP:

$$\begin{aligned} &\text{find} && \lambda_0 \\ &\text{subject to} && \bar{y}_0 = -\theta_0 x - \bar{c}_0 + \lambda_0, \\ & && 0 \le \lambda_0 \perp \bar{y}_0 \ge 0. \end{aligned}$$

Following Lemma 20, $\lambda_0(x) = \max\{0, h_0(x)\} = \lambda_{\text{ReLU}} \circ h_0(x)$. Similarly, notice that for $i > 0$, λ_i only depends on $\lambda_{i-1}(x)$ and is the unique element of:

$$\begin{aligned} \text{find} \quad & \lambda_i \\ \text{subject to} \quad & \bar{y}_i = -\theta_i \lambda_{i-1}(x) - \bar{c}_i + \lambda_i, \\ & 0 \leq \lambda_i \perp \bar{y} \geq 0, \end{aligned}$$

and is equivalent to $\lambda_i(x) = \lambda_{\text{ReLU}} \circ h_i \circ \lambda_{i-1}(x)$ as a direct application of Lemma 20. Using this equivalency recursively,

$$\bar{D}\lambda(x) + \bar{z} = h_L \circ \lambda_{\text{ReLU}} \circ h_{L-1} \circ \ldots \circ \lambda_{\text{ReLU}} \circ h_0(x).$$

Notice that $\bar{F}_{\alpha\alpha}$ is lower triangular with ones on the diagonal for any α such that $\text{card}(\alpha) \geq 2$ hence $\bar{F}$ is a P-matrix. □

Each neuron in the NN is represented with a complementarity variable, therefore the dimension of the complementarity vector (λ) is equal to the number of neurons in the network. As seen in Lemma 22, transforming a ReLU neural network into an LCP only requires concatenating vectors and matrices. Furthermore, this transormation allows us to consider the ReLU neural network as an LCP based controller (Tanwani et al., 2018).

6.2.2. Linear Complementarity Systems with Neural Network Controllers

We will use the LCP representation of the neural network (6.1) and describe an LCS with a NN controller as an A-LCS. Consider a linear complementarity system with a ReLU neural network controller $u_k = \phi(x_k)$:

$$\begin{aligned} x_{k+1} &= Ax_k + B\phi(x_k) + \tilde{D}\tilde{\lambda}_k + \tilde{z}, \\ \tilde{y}_k &= \tilde{E}x_k + \tilde{F}\tilde{\lambda}_k + H\phi(x_k) + \tilde{c}, \\ 0 &\leq \tilde{\lambda}_k \perp \tilde{y}_k \geq 0, \end{aligned} \tag{6.6}$$

where $x_k \in \mathbb{R}^{n_x}$ is the state, $\tilde{\lambda}_k \in \mathbb{R}^{n_{\tilde{\lambda}}}$ is the complementarity variable, $\phi(x) \in \mathbb{R}^{n_\phi}$ is a ReLU neural network as in (6.1) with n_t neurons. Notice that (6.6) is not in the A-LCS form. Using Lemma 22, we can transform (6.6) into an A-LCS in a higher dimensional space. To see this, observe that (6.6) is equivalent to

$$\begin{aligned}
x_{k+1} &= Ax_k + B(\bar{D}\bar{\lambda}_k + \bar{z}) + \tilde{D}\tilde{\lambda}_k + \tilde{z}, \\
\tilde{y}_k &= \tilde{E}x_k + \tilde{F}\tilde{\lambda}_k + H(\bar{D}\bar{\lambda}_k + \bar{z}) + \tilde{c}, \\
\bar{y}_k &= \bar{E}x_k + \bar{F}\bar{\lambda}_k + \bar{c}, \\
0 &\leq \tilde{\lambda}_k \perp \tilde{y}_k \geq 0, \\
0 &\leq \bar{\lambda}_k \perp \bar{y}_k \geq 0,
\end{aligned}$$

after direct application of Lemma 22 where $\bar{\lambda} \in \mathbb{R}^{n_t}$. We can write it succinctly as

$$\begin{aligned}
x_{k+1} &= Ax_k + D\lambda_k + z, \\
y_k &= Ex_k + F\lambda_k + c, \\
0 &\leq \lambda_k \perp y_k \geq 0,
\end{aligned} \tag{6.7}$$

where $\lambda_k = \begin{bmatrix} \tilde{\lambda}_k \\ \bar{\lambda}_k \end{bmatrix}$, $y_k = \begin{bmatrix} \tilde{y}_k \\ \bar{y}_k \end{bmatrix}$, $D = \begin{bmatrix} \tilde{D} & B\bar{D} \end{bmatrix}$, $E = \begin{bmatrix} \tilde{E} \\ \bar{E} \end{bmatrix}$, $F = \begin{bmatrix} \tilde{F} & H\bar{D} \\ 0 & \bar{F} \end{bmatrix}$, $c = \begin{bmatrix} \tilde{c} + H\bar{z} \\ \bar{c} \end{bmatrix}$, and $z = B\bar{z} + \tilde{z}$. Here, the size of $x_k \in \mathbb{R}^{n_x}$ does not change, but notice that now $\lambda_k \in \mathbb{R}^{n_\lambda}$ where $n_\lambda = n_t + n_{\tilde{\lambda}}$. Using controllers of the form (6.1), we will exclusively consider the linear complementarity system model (6.7) for notational compactness.

Similarly, one can consider the scenario where both the system dynamics and the controller are represented by ReLU neural networks as in $x_{k+1} = \phi_1(x_k) + B\phi_2(x_k)$, where ϕ_1 represents the autonomous part of the dynamics (obtained by, for example, system identification) and ϕ_2 is the controller. Using Lemma 22, this system has an equivalent A-LCS representation similar to (6.7), but the details are omitted for brevity.

After obtaining an A-LCS representation of the closed-loop system, one can directly use the existing tools for stability analysis of complementarity systems, such as Lyapunov functions (Camlibel et al.,

2007) and semidefinite programming (Aydinoglu et al., 2020). We will elaborate on this in the next section.

6.3. Stability Analysis of the Closed-Loop System

In this section, we provide sufficient conditions for stability in the sense of Lyapunov for an A-LCS. Then, we show that the stability verification problem is equivalent to finding a feasible solution to a set of linear matrix inequalities (LMI's). To begin, consider the following Lyapunov function candidate that was introduced in (Camlibel et al., 2007):

$$V(x_k, \lambda_k) = \begin{bmatrix} x_k \\ \lambda_k \\ 1 \end{bmatrix}^\top \underbrace{\begin{bmatrix} P & Q & h_1 \\ Q^\top & R & h_2 \\ h_1^T & h_2^T & h_3 \end{bmatrix}}_{:=M} \begin{bmatrix} x_k \\ \lambda_k \\ 1 \end{bmatrix}, \tag{6.8}$$

where $P \in \mathbb{S}^{n_x}$, $Q \in \mathbb{R}^{n_x \times n_\lambda}$, $R \in \mathbb{S}^{n_\lambda}$, $h_1 \in \mathbb{R}^{n_x}$, $h_2 \in \mathbb{R}^{n_\lambda}$, and $h_3 \in \mathbb{R}$ are to be chosen. Note that if F in (6.7) is a P-matrix, then λ_k is a piecewise affine function of x_k, implying that the Lyapunov function (6.8) is quadratic in the pair (x_k, λ_k) but it is *piecewise* quadratic (PWQ) in the state x_k. If F is not a P-matrix, then V can be set valued since there can be multiple λ_k's corresponding to each x_k. In either case, V reduces to a common quadratic Lyapunov function in the special case $Q = R = 0$. Therefore, (6.8) is more expressive than a common quadratic Lyapunov function.

In the following theorem, we construct sufficient conditions for the stability of (6.7), using the Lyapunov function (6.8). This is the discrete time version of the results in (Camlibel et al., 2007).

Theorem 23. *Consider the A-LCS in* (6.7) *with the equilibrium* $x_e = 0$*, the Lyapunov function* (6.8) *and a domain* $\mathcal{X} \subseteq \mathbb{R}^n$*. If there exist* $M \in \mathbb{S}^{n_x+n_\lambda+1}$*,* $\alpha_1 > 0$*, and* $\alpha_2 > \alpha_3 \geq 0$ *such that*

$$\alpha_1 ||x_k||_2^2 \leq V(x_k, \lambda_k) \leq \alpha_2 ||x_k||_2^2, \ \forall (x_k, \lambda_k) \in \Gamma_1,$$
$$V(x_{k+1}, \lambda_{k+1}) - V(x_k, \lambda_k) \leq -\alpha_3 ||x_k||_2^2, \ \forall (x_k, \lambda_k, \lambda_{k+1}) \in \Gamma_2,$$

where $x_{k+1} = Ax_k + D\lambda_k + z$ *and*

$$\begin{aligned}\Gamma_1 &= \{(x_k, \lambda_k) :\ 0 \le \lambda_k \perp Ex_k + F\lambda_k + c \ge 0,\ x_k \in \mathcal{X}\},\\ \Gamma_2 &= \{(x_k, \lambda_k, \lambda_{k+1}) : 0 \le \lambda_k \perp Ex_k + F\lambda_k + c \ge 0,\\ &\qquad 0 \le \lambda_{k+1} \perp Ex_{k+1} + F\lambda_{k+1} + c \ge 0,\ x_k \in \mathcal{X}\}.\end{aligned}$$

Then the equilibrium is Lyapunov stable if $\alpha_3 = 0$ *and geometrically stable if* $\alpha_2 > \alpha_3 > 0$.

Proof. Observe that for all (x_k, λ_k):

$$\alpha_1||x_k||_2^2 \le V(x_k, \lambda_k) \le V(x_0, \lambda_0) \le \alpha_2||x_0||_2^2,$$

and Lyapunov stability follows. For geometric stability, notice that Lyapunov decrease condition is equivalent to $V(x_{k+1}, \lambda_{k+1}) - \gamma V(x_k, \lambda_k) \le 0$, for some $\gamma \in (0, 1)$. Then

$$\alpha_1||x_k||_2^2 \le V(x_k, \lambda_k) \le \gamma^k V(x_0, \lambda_0) \le \alpha_2\gamma^k||x_0||_2^2.$$

The result follows. □

Note that we do not require M in (6.8) to be positive definite to satisfy the requirements of Theorem 23. In light of this theorem, we must solve the following feasibility problem to verify that if the equilibrium of the closed-loop system (6.7) is stable on $\mathcal{X}$:

$$\begin{aligned}\text{find}\quad & P, Q, R, h_1, h_2, h_3, \alpha_1, \alpha_2, \alpha_3 \\ \text{s.t.}\quad & \alpha_1||x_k||_2^2 \le V(x_k, \lambda_k) \le \alpha_2||x_k||_2^2,\ \text{for } (x_k, \lambda_k) \in \Gamma_1,\\ & \Delta V \le -\alpha_3||x_k||_2^2,\ \text{for } (x_k, \lambda_k, \lambda_{k+1}) \in \Gamma_2,\end{aligned} \tag{6.9}$$

where $\Delta V = V(x_{k+1}, \lambda_{k+1}) - V(x_k, \lambda_k)$. In the following proposition, we turn (6.9) with $\mathcal{X} = \mathbb{R}^n$ into an LMI feasibility problem using the S-procedure (Boyd et al., 1994).

Proposition 24. *The following LMI's solve* (6.9) *with* $\mathcal{X} = \mathbb{R}^n$*:*

$$T_1 - S_1^T W_1 S_1 - \frac{1}{2}(S_{3,1} + S_{3,1}^\top) \succeq 0, \tag{6.10a}$$

$$T_2 + S_1^\top W_2 S_1 + \frac{1}{2}(S_{3,2} + S_{3,2}^\top) \preceq 0, \tag{6.10b}$$

$$T_3 + S_2^T W_3 S_2 + S_5^T W_4 S_5 + \frac{1}{2}[(S_4 + S_4^\top) + (S_6 + S_6^\top)] \succeq 0, \tag{6.10c}$$

where $G_1 = D^T P z + D^T h_1 - h_2$, $G_2 = z^T P D + h_1^T D - h_2$, $G_3 = z^T Q + h_2^T$, $S_1 = \begin{bmatrix} E & F & c \\ 0 & I & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $S_2 = \begin{bmatrix} E & F & 0 & c \\ 0 & I & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, $S_{3,i} = \begin{bmatrix} 0 & 0 & 0 \\ J_i E & J_i F & J_i c \\ 0 & 0 & 0 \end{bmatrix}$, $S_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ J_3 E & J_3 F & 0 & J_3 c \\ 0 & 0 & 0 & 0 \end{bmatrix}$, $S_5 = \begin{bmatrix} EA & ED & Fc & Ecz + c \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$, $S_6 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ J_4 EA & J_4 ED & J_4 F & J_4 Ez + c \\ 0 & 0 & 0 & 0 \end{bmatrix}$, $T_1 = \begin{bmatrix} P - \alpha_1 I & Q & h_1 \\ Q^\top & R & h_2 \\ h_1^T & h_2^T & h_3 \end{bmatrix}$, $T_2 = \begin{bmatrix} P - \alpha_2 I & Q & h_1 \\ Q^\top & R & h_2 \\ h_1^T & h_2^T & h_3 \end{bmatrix}$, $T_3 = -\begin{bmatrix} A^T P A - P + \alpha_3 I & A^T P D - Q & A^T Q & A^T P z - h_1 \\ D^T P A - Q^T & D^T P D - R & D^T Q & G_1 \\ Q^T A & Q^T D & R & Q^T z + h_2 \\ z^T P A - h_1^T & G_2 & G_3 & z^T P z - h_1^T z \end{bmatrix}$.

Here, W_i *are decision variables with non-negative entries, and* $J_i = \operatorname{diag}(\tau_i)$ *where* $\tau_i \in \mathbb{R}^m$ *are free decision variables.*

Proof. First define $e_k^\top = \begin{bmatrix} x_k^\top & \lambda_k^\top & 1 \end{bmatrix}$. By left and right multiplying both sides of (6.10a) by $e_k^\top$

and e_k, respectively, we obtain

$$V(x_k, \lambda_k) - \alpha_1 \|x_k\|_2^2 \geq \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix}^\top W_1 \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix} + 2\lambda_k^\top \operatorname{diag}(\tau_1) y_k$$

The right hand side is non-negative due to the complementarity constraint $0 \leq \lambda_k \perp y_k \geq 0$. Similarly, by left and right multiplying both sides of (6.10b) by $e_k^\top$ and e_k, respectively, we obtain

$$\alpha_2 \|x_k\|_2^2 - V(x_k, \lambda_k) \geq \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix}^\top W_2 \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix} + 2\lambda_k^\top \operatorname{diag}(\tau_2) y_k$$

Again, the right hand side is non-negative due to the complementarity constraint $0 \leq \lambda_k \perp y_k \geq 0$.

Now, we define $p_k^\top = \begin{bmatrix} x_k^\top & \lambda_k^\top & \lambda_{k+1}^\top & 1 \end{bmatrix}$. Notice that if we left and right multiply both sides of (6.10c) by $p_k^\top$ and p_k, we obtain

$$\begin{aligned} -\Delta V - \alpha_3 ||x_k||_2^2 \geq & \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix}^\top W_3 \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix} + \begin{bmatrix} y_{k+1} \\ \lambda_{k+1} \\ 1 \end{bmatrix}^\top W_4 \begin{bmatrix} y_k \\ \lambda_k \\ 1 \end{bmatrix} \\ & + 2\lambda_k^\top \operatorname{diag}(\tau_3) y_k + 2\lambda_{k+1}^\top \operatorname{diag}(\tau_4) y_{k+1} \end{aligned}$$

Similarly, all the terms on the right hand side are non-negative since $0 \leq \lambda_k \perp y_k \geq 0$ for all k. This concludes the proof. □

Notice that (6.9) captures the non-smooth structure of the LCS combined with the ReLU neural network controller. In addition to that, we can assign a different quadratic function to each polyhedral partition that is created by the neural network without enumerating those partitions by exploiting the complementarity structure of the neural network. Observe that (6.10a), (6.10b) are LMI's of size $(n_x + n_\lambda + 1)$, and (6.10c) is an LMI of size $(n_x + 2n_\lambda + 1)$.

Note that Theorem 6.10 is a global result for $\mathcal{X} = \mathbb{R}^n$. We can adapt the theorem to bounded regions $\mathcal{X}$ containing the origin.

Remark 25. *For the equilibrium $x_e = 0$, the region of attraction is defined as*

$$\mathcal{R} = \{x_0 : \lim_{k\to\infty} ||x_k|| = 0\}.$$

If one adds (to the left side) $+\eta L$ to (6.10c) *where*

$$L = \begin{bmatrix} -P & -Q & 0 & h_1 \\ -Q^T & R & 0 & -h_2 \\ 0 & 0 & 0 & 0 \\ -h_1^T & -h_2^T & 0 & \xi - h_3 \end{bmatrix},$$

and η is a non-negative scalar variable, then the closed-loop system is geometrically stable and the sub-level set

$$\mathcal{V}_\xi = \{x : V(x,\lambda) \leq \xi \ \forall (x,\lambda) \in \Gamma_1\},$$

is an approximation of the ROA, i.e., $\mathcal{V}_\xi \subseteq \mathcal{R}$. To see this, note that the resulting matrix inequality would imply

$$\begin{aligned} &\alpha_1 \|x_k\|_2^2 \leq V(x_k,\lambda_k) \leq \alpha_2 \|x_k\|_2^2, \\ &V(x_{k+1},\lambda_{k+1}) - V(x_k,\lambda_k) + \alpha_3 \|x_k\|_2^2 + \eta(\xi - V(x_k,\lambda_k)) \leq 0. \end{aligned}$$

From the second inequality, if $V(x_k,\lambda_k) \leq \xi$, then for some $\gamma \in (0,1)$ we have $V(x_{k+1},\lambda_{k+1}) \leq \gamma V(x_k,\lambda_k) \leq \xi$. By induction, if $V(x_0,\gamma_0) \leq \xi$, then $\alpha_1\|x_k\|_2^2 \leq V(x_k,\gamma_k) \leq \gamma^k V(x_0,\gamma_0) \leq \gamma^k \alpha_2 \|x_0\|_2^2$.

Remark 26. *In order to prove the Lyapunov conditions over the ellipsoid $\mathcal{X} = \{x : x^T N x \leq \xi\}$,*

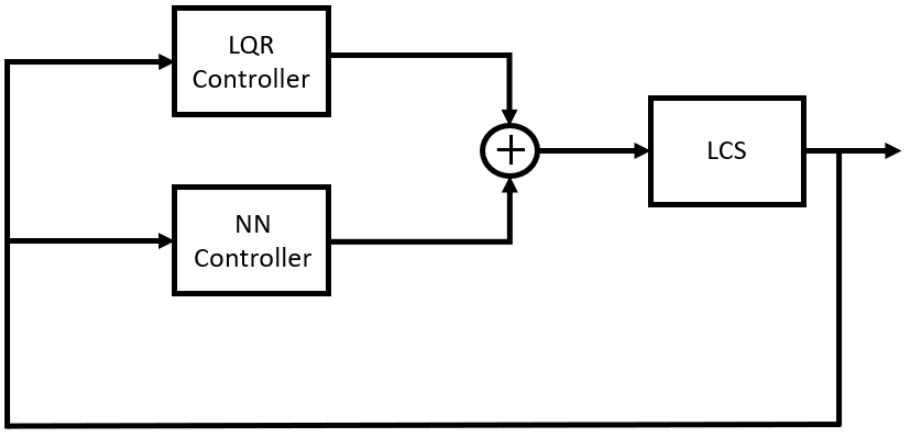

Figure 6.2: Block diagram of the closed-loop system.

one can add (to the left side) $-\beta_1 N_1$ *to* (6.10a), $+\beta_2 N_1$ *to* (6.10b) *and* $+\beta_3 N_2$ *to* (6.10c) *where*

$$N_1 = \begin{bmatrix} -N & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \xi \end{bmatrix}, N_2 = \begin{bmatrix} -N & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \xi \end{bmatrix},$$

and β_i *are non-negative scalar variables.*

6.4. Examples

We use YALMIP (Lofberg, 2004) toolbox with MOSEK (Mosek, 2010) to formulate and solve the linear matrix inequality feasibility problems. PATH (Dirkse and Ferris, 1995) has been used to solve the linear complementarity problems when simulating the system. PyTorch is used for training neural network controllers (Paszke et al., 2017). The experiments are done on a desktop computer with the processor Intel *i7-9700* and *16GB RAM* unless stated otherwise. For all of the experiments, we consider the closed-loop system in Figure 6.2 and the linear-quadratic regulator controller is designed with state penalty matrix $Q^{\text{LQR}} = 10I$ and input penalty matrix $R^{\text{LQR}} = I$ unless stated otherwise. Now, we introduce the mixed-integer problem for (3.2) that connects the complementarity constraints into equivalent big-M mixed integer constraints:

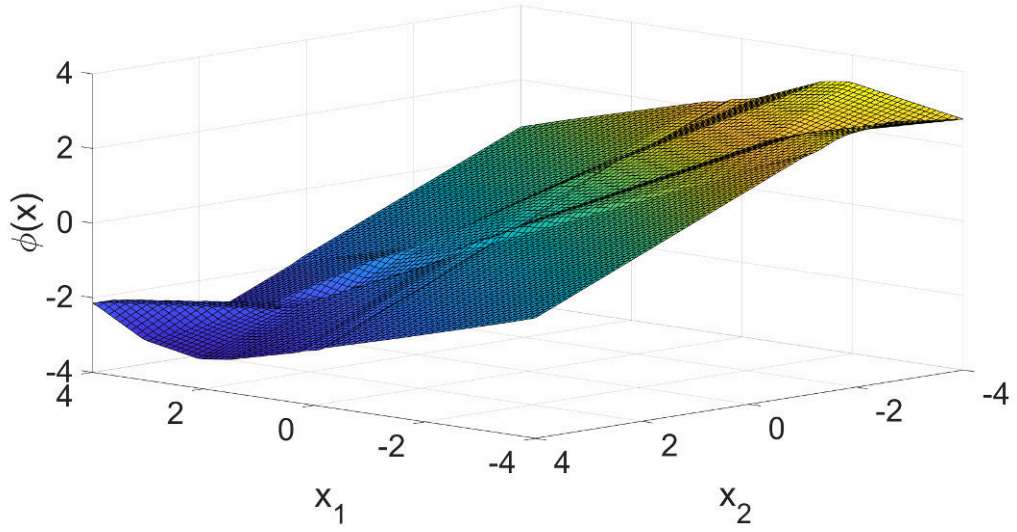

Figure 6.3: Neural network (ϕ) policy for the double-integrator example.

$$\begin{aligned}
\min_{x_k,\lambda_k,u_k} \quad & \sum_{k=0}^{N-1} x_k^T Q^{\mathrm{OPT}} x_k + u_k^T R^{\mathrm{OPT}} u_k + x_k^T Q_N^{\mathrm{OPT}} x_k \\
\text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k + D\lambda_k + z, \\
& M_1 s_k \geq Ex_k + F\lambda_k + Hu_k + c \geq 0, \\
& M_2(\mathbf{1} - s_k) \geq \lambda_k \geq 0, \\
& s_k \in \{0,1\}^m, x_0 = x(0),
\end{aligned} \tag{6.11}$$

where $\mathbf{1}$ is a vector of ones, and M_1, M_2 are scalars that are used for the big M method. Moving forward, we will consider function $\pi_{\mathrm{OPT}}(x(0))$ that returns the first element of the optimal input sequence, u_0^*, for a given $x(0)$ and learn this function using a neural network $\phi(x)$. The code for all examples is available[11].

6.4.1. Double Integrator

In this example, we consider a double integrator model:

$$x_{k+1} = Ax_k + Bu_k,$$

[11] https://github.com/AlpAydinoglu/sverification

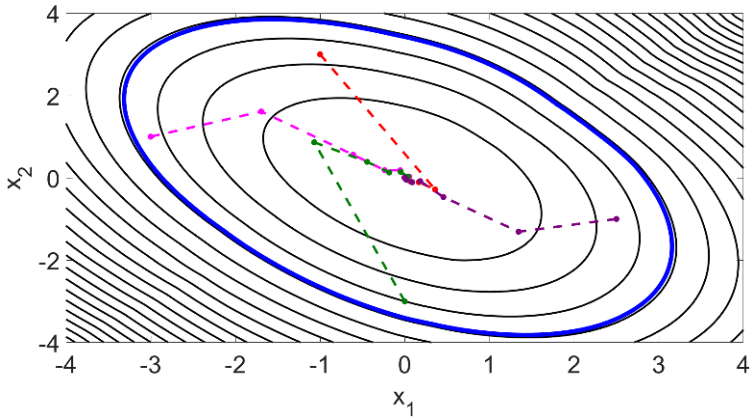

Figure 6.4: Sublevel sets of the piece-wise quadratic Lyapunov function $V(x_k, \lambda_k)$ with four different trajectories for the double integrator example. A sublevel set that lies in the constraint set is shown in blue.

where $\tilde{A} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}$, and $A = \tilde{A} + BK_{\text{LQR}}$, where LQR gains are $Q^{\text{LQR}} = 0.1I$ and $R^{\text{LQR}} = 1$. This simple model serves as an example where we approximate an explicit model predictive controller (explicit MPC) (Bemporad et al., 2002) using a neural network and verify the stability of the resulting system. We consider the state and input constraints:

$$\mathcal{X} = \{x : \begin{bmatrix} -4 \\ -4 \end{bmatrix} \leq x \leq \begin{bmatrix} 4 \\ 4 \end{bmatrix}\}, \, \mathcal{U} = \{u : -3 \leq u \leq 3\},$$

and obtain 2000 samples of the form $(x, \pi_{\text{MPC}}(x))$ with $N = 10$, $Q^{\text{MPC}} = 10I$, and $R^{\text{MPC}} = 1$ where Q^{MPC} is the penalty on the state, R^{MPC} is the penalty on the input and $\pi_{\text{MPC}}(x)$ is the first element of the optimal input sequence for a given state x (Borrelli et al., 2017). Next we approximate the explicit MPC controller using a ReLU network $\phi(x)$ with two layers and 10 neurons in each layer as in Figure 6.3. Now, consider the closed-loop system:

$$x_{k+1} = Ax_k + B\phi(x_k). \tag{6.12}$$

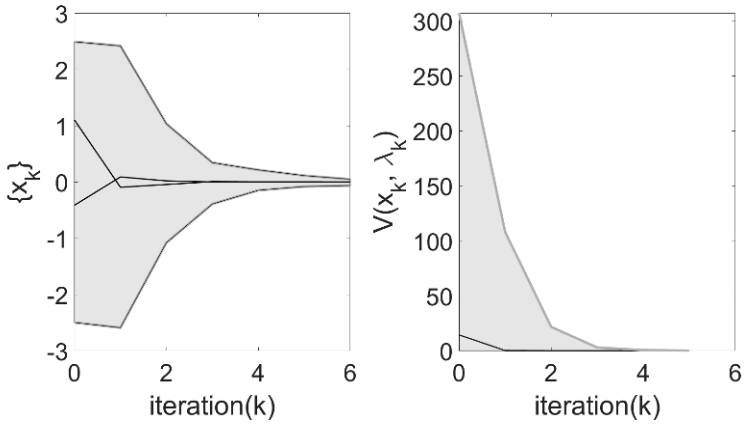

Figure 6.5: Envelopes for 1000 trajectories and their corresponding Lyapunov functions (in gray) with a sample trajectory (in black) for the double integrator example.

First, we find the equivalent LCP representation of $\phi(x)$ using Lemma 22. Then, we write the equivalent LCS representation of (6.12) as described in Section 6.2.2. We computed the piece-wise quadratic Lyapunov function of the form (6.8) and verified exponential stability in 0.81 seconds. The sublevel sets of the Lyapunov functions are plotted in Figure 6.4 and the envelopes of 1000 trajectories with their corresponding Lyapunov functions in Figure 6.5.

6.4.2. Cart-pole with Soft Walls

We consider the regulation problem of a cart-pole with soft-walls as in Figure 5.2. This problem has been studied in (Marcucci and Tedrake, 2020; Deits et al., 2019; Aydinoglu et al., 2021b) and is a benchmark in contact-based control algorithms. In this model, x_1 represents the position of the cart, x_2 represents the angle of the pole and x_3, x_4 are their time derivatives respectively. Here, λ_1 and λ_2 represent the contact force applied by the soft walls to the pole from the right and left

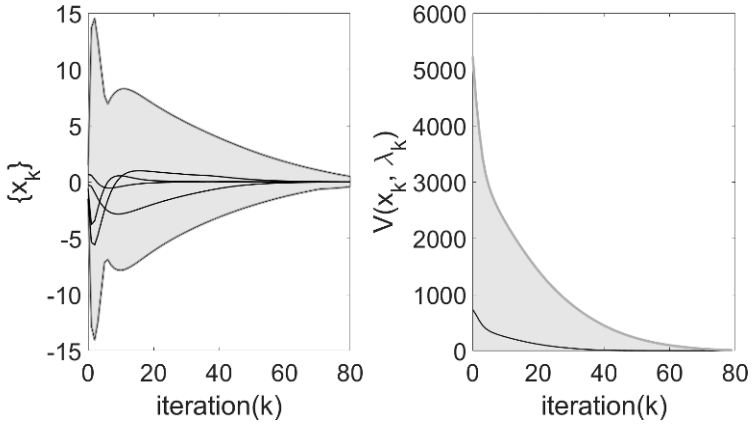

Figure 6.6: Envelopes for 1000 trajectories and the corresponding Lyapunov functions (in gray) with a sample trajectory (in black) for the cart-pole example.

walls, respectively. We consider the linearized model around $x_2 = 0$:

$$
\begin{aligned}
\dot{x}_1 &= x_3, \\
\dot{x}_2 &= x_4, \\
\dot{x}_3 &= g\frac{m_p}{m_c}x_2 + \frac{1}{m_c}u_1, \\
\dot{x}_4 &= \frac{g(m_c+m_p)}{lm_c}x_2 + \frac{1}{lm_c}u_1 + \frac{1}{lm_p}\lambda_1 - \frac{1}{lm_p}\lambda_2, \\
0 &\leq \lambda_1 \perp lx_2 - x_1 + \frac{1}{k_1}\lambda_1 + d \geq 0, \\
0 &\leq \lambda_2 \perp x_1 - lx_2 + \frac{1}{k_2}\lambda_2 + d \geq 0,
\end{aligned}
$$

where $m_c = 1$ is the mass of the cart, $m_p = 1$ is the mass of the pole, $l = 1$ is the length of the pole, $k_1 = k_2 = 1$ are the stiffness parameter of the walls, $d = 1$ is the distance between the origin and the soft walls. Then, we discretize the dynamics using the explicit Euler method with time step $T_s = 0.1$ to obtain the system matrices: $\tilde{A} = \begin{bmatrix} 1 & 0 & 0.1 & 0 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0.981 & 1 & 0 \\ 0 & 1.962 & 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 0 \\ 0.1 \\ 0.1 \end{bmatrix}$, $\tilde{D} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0.1 & -0.1 \end{bmatrix}$,

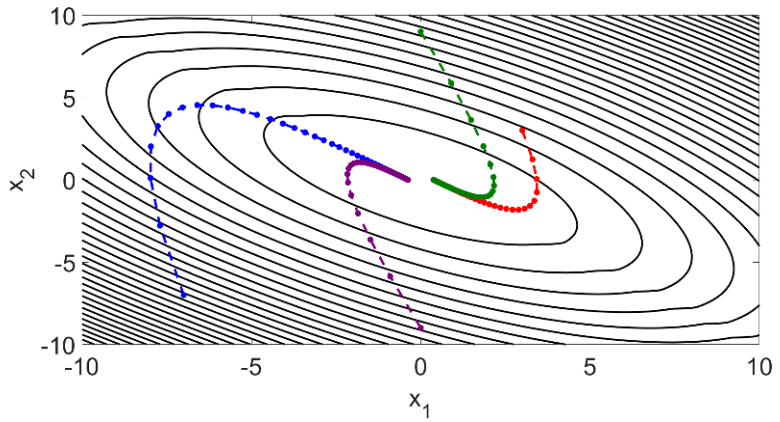

Figure 6.7: Sublevel sets of the piece-wise quadratic Lyapunov function $V(x_k, \lambda_k)$ with four different trajectories for the box with friction example.

$\tilde{E} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix}$, $\tilde{F} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $\tilde{c} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, $A = \tilde{A} + BK_{LQR}$ and, K_{LQR} is the gain of the linear-quadratic regulator that stabilizes the linear system $(\tilde{A}, B)$. However, the equilibrium $x_e = 0$ is not globally stable due to the soft walls.

We solve the optimal control problem (6.11) with $N = 10$, $Q^{\text{OPT}} = 10I$, $R^{\text{OPT}} = 1$, and Q_N^{OPT} as the solution of the discrete algebraic Riccati equation to generate samples of the form $(x, \pi_{\text{OPT}}(x))$. For this particular problem, we generate 4000 samples and we train a neural network $\phi(x)$ with two layers, each with 10 neurons, to approximate the optimal controller π_{OPT} and we used the ADAM optimizer to do the training. Then, we analyze the linear complementarity system with the neural network controller $u_k = \phi(x_k)$. Following the procedure in Section 6.2, we first express the neural network as a linear complementarity problem using Lemma 22 and then transform the LCS with the NN controller into the form (6.7). We compute a Lyapunov function of the form (6.8) in 1.61 seconds that verifies that the closed-loop system with the neural network controller $\phi(x)$ is globally exponentially stable. For this example, a common Lyapunov function is enough to verify stability. In Figure 6.6, we present the envelopes for 1000 trajectories.

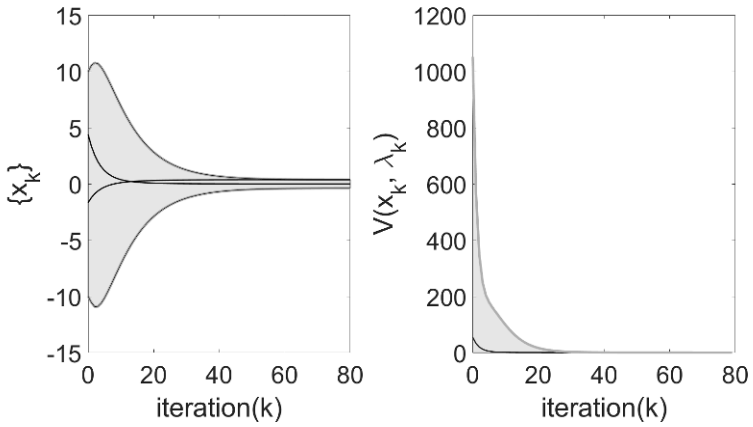

Figure 6.8: Envelopes for 1000 trajectories and the corresponding Lyapunov functions (in gray) with a sample trajectory (in black) for the box with friction example.

6.4.3. Box with Friction

In this example, we consider the regulation task of a box on a surface as in Figure 5.11. This simple model serves as an example where the contact forces λ_k are not unique due to Coulomb friction between the surface and the box. Here, x_1 is the position of the cart, x_2 is the velocity of the cart, u is the input applied to the cart, $g = 9.81$ is the gravitational acceleration, $m = 1$ is the mass of the cart, and $\mu = 0.1$ is the coefficient of friction between the cart and the surface. The system can be modeled by:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + \tilde{D}\tilde{\lambda}_k, \\ 0 \leq \tilde{\lambda}_k &\perp \tilde{E}x_k + \tilde{F}\tilde{\lambda}_k + Hu_k + \tilde{c} \geq 0, \end{aligned} \tag{6.13}$$

where $\tilde{A} = \begin{bmatrix} 1 & 0.1 \\ 0 & 1 \end{bmatrix}$, $B = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix}$, $\tilde{D} = \begin{bmatrix} 0 & 0 & 0 \\ 0.1 & -0.1 & 0 \end{bmatrix}$, $\bar{E} = \begin{bmatrix} 0 & 1 \\ 0 & -1 \\ 0 & 0 \end{bmatrix}$, $\tilde{F} = \begin{bmatrix} 1 & -1 & 1 \\ -1 & 1 & 1 \\ -1 & -1 & 0 \end{bmatrix}$, $\tilde{c} = \begin{bmatrix} 0 \\ 0 \\ 0.981 \end{bmatrix}$, $H = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$, $\tilde{E} = \bar{E} + HK_{LQR}$, $A = \tilde{A} + BK_{LQR}$ and, K_{LQR} is the gain that (the linear-quadratic regulator controller) stabilizes the linear system $(\tilde{A}, B)$. Observe that the matrix $\tilde{F}$ is not a P-matrix, hence for a given x_k, the contact forces λ_k are not unique. Similar to

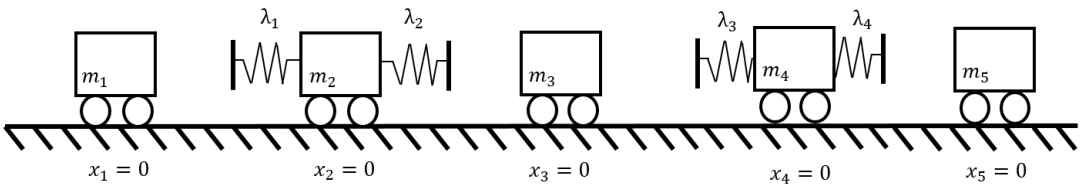

Figure 6.9: Regulation task of five carts to their respective origins.

the previous example, we generate 2000 samples $(x, \pi_{\text{OPT}}(x))$ for the LCS in (6.13) with $N = 5$, $Q^{\text{OPT}} = Q_N^{\text{OPT}} = 0.1$, $R^{\text{OPT}} = 1$ and train a neural network $\phi(x)$ that approximates the optimal controller. Then we convert the system in (6.13) with the neural network controller $\phi(x)$ into the form (6.7). Next, we compute the piece-wise quadratic Lyapunov function (with sublevel sets shown in Figure 6.7) of the form (6.8) in 1.05 seconds such that the exponential stability condition is verified outside a ball around the origin, $\mathcal{D} = \{x : ||x||_2^2 > 0.6\}$. More precisely, we prove convergence to a set (smallest sublevel set of V that contains $\mathcal{D}$) which contains the equilibrium. This is expected because the trajectories do not reach the origin due to stiction. We demonstrate the envelopes for 1000 trajectories and their respective Lyapunov functions in Figure 6.8.

6.4.4. Five Carts

We consider the regulation task of five carts as in Figure 6.9. Here x_i describes the state of the i-th cart, the interaction between the carts is modeled by soft springs represented by λ_i, and all carts can be controlled via the applied force u_i. We approximate Newtons's second law with a force balance equation and obtain the following quasi-static model:

$$
\begin{aligned}
x_{k+1}^{(1)} &= x_k^{(1)} + u_k^{(1)} - \lambda_k^{(1)}, \\
x_{k+1}^{(i)} &= x_k^{(i)} + u_k^{(i)} - \lambda_k^{(i-1)} - \lambda_k^{(i)}, \text{ for } i = 2, 3, 4, \\
x_{k+1}^{(5)} &= x_k^{(5)} + u_k^{(5)} + \lambda_k^{(4)}, \\
0 \leq \lambda_k^{(i)} &\perp x_k^{(i+1)} - x_k^{(i)} + \lambda_k^{(i)} \geq 0.
\end{aligned}
$$

We designed an LQR controller with with state penalty matrix $Q^{\text{LQR}} = I$ and input penalty matrix $R^{\text{LQR}} = I$. Then, we solve the optimal control problem (6.11) with $N = 10$, $Q^{\text{OPT}} = Q_N^{\text{OPT}} = 10I$,

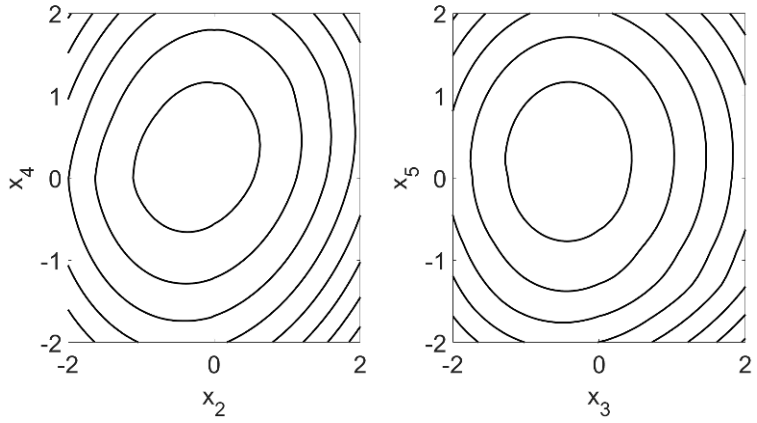

Figure 6.10: Sublevel sets of the piece-wise quadratic Lyapunov function for the five carts example on the planes $\mathcal{P}_1 = \{x : x_1 = x_3 = x_5 = 0\}$ and $\mathcal{P}_2 = \{x : x_1 = x_2 = x_4 = 0\}$ respectively.

and $R^{\text{OPT}} = 1$ to generate 2000 samples of the form $(x, \pi_{\text{OPT}}(x))$. Using these samples, we train $\phi(x)$ with two layers of size 10 and express the neural network as a linear complementarity problem using Lemma 22.

We compute a piece-wise quadratic Lyapunov function of the form (6.8) in 1.63 seconds (sub-level sets as in Figure 6.10) that verifies that the closed-loop system with the neural network controller $\phi(x)$ is globally exponentially stable outside a ball $\mathcal{D} = \{x : ||x||_2^2 > 0.1\}$. We also verified that there is not a common Lyapunov function that satisfies the LMI's in (6.10). We note that a common Lyapunov function that satisfies Theorem 23 might exist, but no such function satisfies our relaxation in (6.10). On the other hand, this demonstrates the importance of searching over a wider class of functions. In Figure 6.11, we present the envelopes for 1000 trajectories and the corresponding Lyapunov functions. We note that *memory* is the limiting factor in terms of scalability of our method and present scalability tests in Table 6.1.

6.5. Conclusion and Discussion

In this work, we have shown that neural networks with ReLU activation functions have an equivalent linear complementarity problem representation. Furthermore, we have shown that a linear complementarity system with a ReLU neural network controller can be transformed into an LCS with a higher dimensional complementarity variable. This allows one to use the existing literature

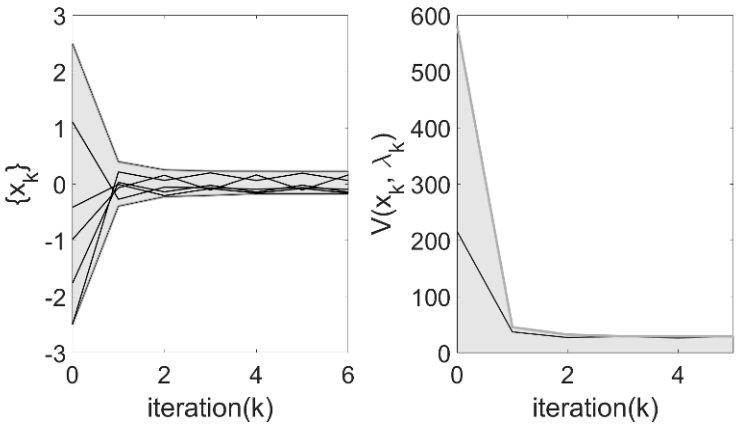

Figure 6.11: Envelopes for 1000 trajectories and the corresponding Lyapunov functions (in gray) with a sample trajectory (in black) for the five carts example.

RAM	**Number of neurons**	**Solve time**
8GB RAM	20	2.1 seconds
8GB RAM	60	194.72 seconds
8GB RAM	100	OOM
16GB RAM	100	1364.78 seconds
16GB RAM	140	OOM

Table 6.1: Scalability tests.

on linear complementarity systems when analyzing an LCS with NN controller.

Towards this direction, we have derived the discrete-time version of the stability results in (Camlibel et al., 2007) and shown that searching for a Lyapunov function for an LCS with ReLU NN controller is equivalent to finding a feasible solution to a set of linear matrix inequalities. The proposed method exploits the complementarity structure of both the system and the NN controller and avoids enumerating the exponential number of potential modes. We have also demonstrated the effectiveness of our method on numerical examples, including a difference inclusion model.

As future work, we are planning to explore tools from algebraic geometry that use samples instead of the S-procedure terms which result in a stronger relaxation (Cifuentes and Parrilo, 2017). Also, we consider using passivity results (Miranda-Villatoro et al., 2018) in order to develop algorithms that can verify the stability for larger neural networks. At last, it is of interest to learn stabilizing neural network controllers utilizing the complementarity viewpoint.

CHAPTER 7

CONCLUSION

In this thesis, we focused on local hybrid models called linear complementarity systems as simple but powerful approximations of multi-contact systems. Employing these local hybrid models, this thesis presents scalable and fast algorithmic solutions for challenging multi-contact problems.

In Section 4, we presented an algorithm, C3, for model predictive control of multi-contact systems. The framework tackles the hybrid MPC problem by shifting the complexity to the projection sub-problems via a consensus formulation. This enables parallelization of the contact scheduling problem and results in a fast MPC approach for robots that make and break contact with their environment. Additionally, we have integrated C3 with an online residual learner and proposed an adaptive MPC framework.

In Section 5, we have introduced a controller that can utilize both state and force feedback and also proposed an algorithm for synthesizing contact-aware control policies for linear complementarity systems with possibly non-unique solutions. It was shown that pure local, linear analysis was entirely insufficient and utilizing contact in the control design is critical to achieve high performance. Furthermore, the proposed algorithm exploits the complementarity structure of the system and avoids enumerating the exponential number of potential modes, enabling efficient design of multi-contact control policies.

In Section 6, we have shown that neural networks with ReLU activation functions have an equivalent linear complementarity problem representation. Furthermore, we have demonstrated that a linear complementarity system with a ReLU neural network controller can be transformed into an LCS with a higher dimensional complementarity variable. This allows one to use the existing literature on linear complementarity systems when analyzing an LCS with NN controller. We then proposed a method for stability analysis that exploits the complementarity structure of both the system and the NN controller and avoids enumerating the exponential number of potential modes.

7.1. Challenges and Open Questions

Computational Challenges. As the thesis proposes numerical algorithms, computational scaling is critical. While the approaches presented in this thesis scale well compared to state-of-the art approaches, there is a room for improving memory usage (Chapter 5, 6) or runtime (Chapter 4). The algorithm in Chapter 5 requires solving feasibility problems that include bilinear matrix inequalities. For the examples presented in this chapter (except Section 5.5.7), the runtime of the algorithm was short and we found solutions to the problems relatively quickly. On the other hand, it is important to note that for some parameter choices and initializations, the solver was unable to produce feasible solutions. For the approach in Section 6, memory is the limiting factor in terms of scalability (as shown in Table 6.1). To improve scalability at the algorithmic level, it is possible to use first-order method solvers to solve the LMI's. For example, the SCS solver (O'donoghue et al., 2016) has lower memory limitations at the expense of slower solve time. For the algorithm proposed in Chapter 4, projections can be difficult to solve, especially as we rely on the MIQP-based projection method for problems with frictional contact. Exploring alternate heuristics for the projection step could be of future interest. Similarly, it would be interested to leverage learning-based approaches (Cauligi et al., 2020) to speed up the process.

Dependence on Models. Except Section 4.6, all the methods presented in this thesis are model-based and rely on accurate models. It is an interesting future research direction to extend our methods (as in Section 4.6) to work well even with relatively large modeling errors.

Role of Locality. The models considered in this thesis are hybrid models that enable decisions such as making and breaking contact. Regardless, such models are still local (as we linearize the smooth components hence the geometry) and are limited (e.g. if the end-effector is close to one side of a cube, it cannot reason about interacting with other sides of the cube and will only make local decisions about that particular side). It is an interesting future direction to use local hybrid planners (e.g. C3) to make global decisions, perhaps using techniques such as sampling (e.g. running local planners from different initial conditions (one per each side of a cube) in parallel to find the best location (which side of the cube is the best to interact with for a given task)).

APPENDIX A

CHAPTER 5

A.1. Matrix Inequalities

We present the matrix inequalities in (5.9) and (5.10) explicitly. We can represent (5.9) as:

$$T_1 - S_1^T W_1 S_1 - \frac{1}{2}(S_{2,1} + S_{2,1}^T) \succeq 0,$$
$$T_2 + S_1^T W_2 S_1 + \frac{1}{2}(S_{2,2} + S_{2,2}^T) \preceq 0,$$

where W_i are decision variables with non-negative entries, $J_i = \text{diag}(\tau_i)$ where τ_i are free decision variables, and

$$T_1 = \begin{bmatrix} P - \gamma_1 I & Q & p/2 \\ * & R & r/2 \\ * & * & z \end{bmatrix}, T_2 = \begin{bmatrix} P - \gamma_2 I & Q & p/2 \\ * & R & r/2 \\ * & * & z \end{bmatrix},$$
$$S_1 = \begin{bmatrix} E & F & c \\ 0 & I & 0 \\ 0 & 0 & 1 \end{bmatrix}, S_{2,i} = \begin{bmatrix} 0 & 0 & 0 \\ J_i E & J_i F & J_i c \\ 0 & 0 & 0 \end{bmatrix}.$$

We can represent (5.9) as:

$$T_3 + S_3^T W_3 S_3 + \frac{1}{2}(S_4 + S_4^T) + \sum_{i=1}^{m} \frac{1}{2}(S_{5,i} + S_{5,i}^T) \preceq 0,$$

where W_i are decision variables with non-negative entries, $J_i = \text{diag}(\tau_i)$ where τ_i are free decision variables and $\zeta_{5,i} = \text{diag}(\theta_i)$ where θ_i are zero everywhere expect the ith entry which is a free

variable and (T_3 is symmetric)

$$T_3 = \begin{bmatrix} PA + A^T P & PD + A^T Q & Q & A^T p/2 + Pa & 0 & 0 \\ * & D^T Q + Q^T D & R & Q^T a + D^T p/2 & 0 & 0 \\ * & * & 0 & r/2 & 0 & 0 \\ * & * & * & p^T a & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$S_3 = \begin{bmatrix} E & F & 0 & c & 0 & 0 \\ 0 & I & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0, \end{bmatrix},$$

$$S_4 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ J_3 E & J_3 F & 0 & J_3 c & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ J_4 E A & J_4 E D & J_4 F + J_5 & J_4 E a & J_4 & J_5 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$S_{5,i} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \zeta_{7,i} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \zeta_{8,i} \\ \zeta_{9,i} E & \zeta_{9,i} F & 0 & \zeta_{9,i} c & 0 & 0 \end{bmatrix}.$$

A.2. Polynomial Optimization

Next, we present the two polynomial optimization problems. The first one is regarding Proposition 17:

$$\begin{aligned} \text{find} \quad & w, \eta, p_i^k, p_{i,j}^k, s_i^k \\ \text{subject to} \quad & \phi_1(\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \boldsymbol{q}) \geq 0, \\ & \phi_2(\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \boldsymbol{q}) \geq 0, \end{aligned} \tag{A.1}$$

and the second one considers the optimization problem in (5.21):

$$\begin{aligned} \min_{w,\eta,p_i^k,p_{i,j}^k,s_i^k} \quad & r^T N^T w \\ \text{subject to} \quad & \phi_1(\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \boldsymbol{q}) \geq 0, \\ & \phi_2(\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \boldsymbol{q}) \geq 0, \\ & |w_i| \leq 1, \ \forall i, \ \eta \geq 0, \end{aligned} \tag{A.2}$$

where the functions ϕ_1 and ϕ_2 are defined as

$$\begin{aligned} \phi_1(\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \boldsymbol{q}) = {} & (\eta + w^T(\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_2))(\boldsymbol{\lambda}_1^T \boldsymbol{\lambda}_1 + \boldsymbol{\lambda}_2^T \boldsymbol{\lambda}_2) \\ & + \sum_i p_i^1 \boldsymbol{\lambda}_{1,i} + \sum_i p_i^2 \boldsymbol{\lambda}_{2,i} + \sum_i \sum_j p_{i,j}^3 \boldsymbol{\lambda}_{1,i} \boldsymbol{\lambda}_{1,j} \\ & + \sum_i p_i^4 (\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_1) + \sum_i p_i^5 (\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_2) \\ & + \sum_i \sum_j p_{i,j}^6 (\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_1)(\boldsymbol{q}_j + F_j^T \boldsymbol{\lambda}_2) \\ & + \sum_i \sum_j p_{i,j}^7 (\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_2) \boldsymbol{\lambda}_{1,j} \\ & + \sum_i \sum_j p_{i,j}^8 (\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_1) \boldsymbol{\lambda}_{2,j} \\ & + \sum_i s_i^1 \boldsymbol{\lambda}_{1,i} (\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_1) \end{aligned}$$

$$+ \sum_i s_i^2 \boldsymbol{\lambda}_{2,i}(\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_2),$$

where $p_i^k, p_{i,j}^k$ are non-negative variables (sum-of-squares polynomials), s_i^k are free variables (polynomials with no restriction) and

$$\begin{aligned}
\phi_2(\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \boldsymbol{q}) = (\eta - w^T(\boldsymbol{\lambda}_1 - \boldsymbol{\lambda}_2))(\boldsymbol{\lambda}_1^T \boldsymbol{\lambda}_1 + \boldsymbol{\lambda}_2^T \boldsymbol{\lambda}_2) \\
+ \sum_i p_i^9 \boldsymbol{\lambda}_{1,i} + \sum_i p_i^{10} \boldsymbol{\lambda}_{2,i} + \sum_i \sum_j p_{i,j}^{11} \boldsymbol{\lambda}_{1,i} \boldsymbol{\lambda}_{1,j} \\
+ \sum_i p_i^{12}(\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_1) + \sum_i p_i^{13}(\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_2) \\
+ \sum_i \sum_j p_{i,j}^{14}(\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_1)(\boldsymbol{q}_j + F_j^T \boldsymbol{\lambda}_2) \\
+ \sum_i \sum_j p_{i,j}^{15}(\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_2) \boldsymbol{\lambda}_{1,j} \\
+ \sum_i \sum_j p_{i,j}^{16}(\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_1) \boldsymbol{\lambda}_{2,j} \\
+ \sum_i s_i^3 \boldsymbol{\lambda}_{1,i}(\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_1) \\
+ \sum_i s_i^4 \boldsymbol{\lambda}_{2,i}(\boldsymbol{q}_i + F_i^T \boldsymbol{\lambda}_2),
\end{aligned}$$

where $p_i^k, p_{i,j}^k$ are non-negative variables (sum-of-squares polynomials), s_i^k are free variables (polynomials with no restriction).

APPENDIX B

CHAPTER 4

B.1. ADMM

We will clarify the derivation of general augmented Lagrangian (4.6) and ADMM iterations (4.7), (4.8), (4.9). Consider the equivalent optimization problem to (4.5):

$$\begin{aligned} \min_{z} \quad & c(z) + \mathcal{I}_{\mathcal{D}}(z) + \mathcal{I}_{\mathcal{C}}(z) + \sum_{k=0}^{N-1} \mathcal{I}_{\mathcal{H}_k}(\delta_k) \\ \text{s.t.} \quad & T(z-\delta) = 0 \end{aligned} \tag{B.1}$$

where $T = \mathbf{blkdiag}(T_0, \ldots, T_{N-1})$ and $T_k^T T_k = G_k$. The augmented Lagrangian is:

$$\begin{aligned} \mathcal{L}_y(z, \delta, y) &= c(z) + \mathcal{I}_{\mathcal{D}}(z) + \mathcal{I}_{\mathcal{C}}(z) \\ &+ \sum_{k=0}^{N-1} \big(\mathcal{I}_{\mathcal{H}_k}(\delta_k) + y_k^T T_k(z_k - \delta_k) + \rho ||T_k(z_k - \delta_k)||_2^2 \big), \end{aligned}$$

where $y^T = [y_0^T, y_1^T, \ldots, y_{N-1}^T]$, y_k are the dual variables. In order to solve (B.1), ADMM iterations are:

$$z^{i+1} = \text{argmin}_z \mathcal{L}_y(z, \delta^i, y^i), \tag{B.2}$$

$$\delta^{i+1} = \text{argmin}_\delta \mathcal{L}_y(z^{i+1}, \delta, y^i), \tag{B.3}$$

$$y^{i+1} = y^i + 2\rho T(z^{i+1} - \delta^{i+1}). \tag{B.4}$$

Notice that individual vectors δ_k^{i+1} in (B.3) can be computed as

$$\delta_k^{i+1} = \text{argmin}_{\delta_k} \mathcal{L}_y^k(z_k^{i+1}, \delta_k, y_k^i)$$

due to separability of $\mathcal{L}_y$, where $\mathcal{L}_y^k(z_k, \delta_k, y_k) = \big(\mathcal{I}_{\mathcal{H}_k}(\delta_k) + y_k^T T_k(z_k - \delta_k) + \rho||T_k(z_k - \delta_k)||_2^2\big)$. Also note that δ_k^{i+1} only depends on z_k^{i+1} and y_k^i. Similarly (B.4) can be written as

$$y_k^{i+1} = y_k^i + 2\rho T_k(z_k^{i+1} - \delta_k^{i+1}).$$

After that, define the scaled dual variables w_k such that $y_k = 2\rho T_k w_k$. With w_k, the augmented Lagrangian is of the following form:

$$\begin{aligned}\mathcal{L}_\rho(z, \delta, w) &= c(z) + \mathcal{I}_{\mathcal{D}}(z) + \mathcal{I}_{\mathcal{C}}(z) \\ &+ \sum_{k=0}^{N-1} \big(\mathcal{I}_{\mathcal{H}_k}(\delta_k) + 2\rho w_k^T G_k(z_k - \delta_k) \\ &+ \rho(z_k - \delta_k)^T G_k(z_k - \delta_k)\big)\end{aligned}$$

following the fact that $G_k = T_k^T T_k$. It is equivalent to:

$$\begin{aligned}\mathcal{L}_\rho(z, \delta, w) &= c(z) + \mathcal{I}_{\mathcal{D}}(z) + \mathcal{I}_{\mathcal{C}}(z) \\ &+ \sum_{k=0}^{N-1} \big(\mathcal{I}_{\mathcal{H}_k}(\delta_k) + \rho(r_k^T G_k r_k - w_k^T G_k w_k)\big).\end{aligned}$$

The corresponding ADMM iterations are:

$$\begin{aligned}z^{i+1} &= \text{argmin}_z \mathcal{L}_\rho(z, \delta^i, w^i), \\ \delta_k^{i+1} &= \text{argmin}_{\delta_k} \mathcal{L}_\rho^k(z_k^{i+1}, \delta_k, w_k^i),\ \forall k, \\ w_k^{i+1} &= w_k^i + z_k^{i+1} - \delta_k^{i+1},\ \forall k\end{aligned}$$

as $y_k = 2\rho T_k w_k$ and T_k is a positive definite matrix with $\mathcal{L}_\rho^k(z_k, \delta_k, w_k) = \mathcal{I}_{\mathcal{H}_k}(\delta_k) + \rho(r_k^T G_k r_k - w_k^T G_k w_k)$.

B.2. Algorithm 3 and Simplified Model

The algorithm requires a model $\mathcal{M}$ (Section 3.4) C3 parameters θ (Section 4.3.2), discretization step length Δt, and a nominal input $\hat{u}$. First, an LCS approxmation ($\mathcal{L}_{\Delta t}$) of model $\mathcal{M}$ around

the state estimate $\hat{x}$ and nominal input $\hat{u}$ is obtained. Then, (following Algorithm 1) one can use this LCS model along with the state estimate and pre-specified C3 parameters (θ) and this process returns u_{C3}. Next, we compute the time spent (Δt_c) during the previous steps and calculate an LCS model $\mathcal{L}_{\Delta t_c}$ using Δt_c as the discretization time-step. Lastly, we compute the desired state x_d and the desired contact force λ_d using $\mathcal{L}_{\Delta t_c}$, $\hat{x}$ and u_{C3}.

Often times, C3 generates the desired end-effector trajectories and contact forces using a simplified model of the robot. Using such models for planning (Chen and Posa, 2020) is extremely common and improves the solve time of the planning algorithms, e.g. C3. This specific simple model, $\mathcal{M}_s$, assumes one can control the translational accelerations of the spherical end-effector (Figure 4.8) in all directions ($\ddot{x}^e_x, \ddot{x}^e_y, \ddot{x}^e_z$) and that the end-effector does not rotate.

B.3. Impedance Control

We use impedance control (Hogan, 1985) to track the end-effector trajectories and contact forces that C3 produces which relies on model of Franka Emika Panda Arm, $\mathcal{M}_f$. Concretely, the controller we implemented is:

$$\begin{aligned} u =& J_f^T \Lambda \begin{bmatrix} K & 0 \\ 0 & B \end{bmatrix} \tilde{x}^e + C_f + \tau_{ff} \\ &+ N_f \big(K_n (q_d^f - q^f) + B_n (\dot{q}_d^f - \dot{q}^f) \big), \end{aligned} \tag{B.5}$$

where x^e is the current state of the end-effector, x^e_d is the desired end-effector state given by C3 and $\tilde{x}^e = x^e_d - x^e$ represents the error. The K and B are gain matrices for position error and velocity error respectively. Given $\mathcal{M}_f$, J_f is the manipulator Jacobian for the end-effector and $\Lambda = \left(J_f M_f^{-1} J_f^T\right)^{-1}$ is the end-effector inertia matrix, where M_f is the manipulator's mass matrix. The nullspace projection matrix N_f directly follows from (Hermus et al., 2021), q_d^f and $\dot{q}_d^f$ are the desired null-space position and velocity for the manipulator, and K_n, B_n are the corresponding gain matrices. The feedforward torque term is computed as $\tau_{ff} = J_c^T \lambda_d$ where λ_d is the desired contact force obtained from C3 and J_c is the contact Jacobian computed using the full-order model.

BIBLIOGRAPHY

Yeuhi Abe, Marco Da Silva, and Jovan Popović. Multiobjective control with frictional contacts. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 249–258, 2007.

Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. *arXiv preprint arXiv:1703.00443*, 2017.

Mihai Anitescu. Optimization-based simulation of nonsmooth rigid multibody dynamics. *Mathematical Programming*, 105:113–143, 2006.

Anuradha M Annaswamy and Alexander L Fradkov. A historical perspective of adaptive control and learning. *Annual Reviews in Control*, 52:18–41, 2021.

Alp Aydinoglu and Michael Posa. Real-time multi-contact model predictive control via admm. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 3414–3421. IEEE, 2022.

Alp Aydinoglu, Victor M Preciado, and Michael Posa. Contact-aware controller design for complementarity systems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1525–1531. IEEE, 2020.

Alp Aydinoglu, Mahyar Fazlyab, Manfred Morari, and Michael Posa. Stability analysis of complementarity systems with neural network controllers. In *Proceedings of the 24th International Conference on Hybrid Systems: Computation and Control*, pages 1–10, 2021a.

Alp Aydinoglu, Philip Sieg, Victor M Preciado, and Michael Posa. Stabilization of complementarity systems via contact-aware controllers. *IEEE Transactions on Robotics*, 38(3):1735–1754, 2021b.

Alp Aydinoglu, Adam Wei, and Michael Posa. Consensus complementarity control for multi-contact mpc. *arXiv preprint arXiv:2304.11259*, 2023.

Arthur Becker, P Kumar, and Ching-Zong Wei. Adaptive control with the stochastic approximation algorithm: Geometry and convergence. *IEEE Transactions on Automatic Control*, 30(4):330–338, 1985.

Alberto Bemporad, Francesco Borrelli, and Manfred Morari. Optimal controllers for hybrid systems: Stability and piecewise linear explicit form. In *Proceedings of the 39th IEEE Conference on Decision and Control (Cat. No. 00CH37187)*, volume 2, pages 1810–1815. IEEE, 2000.

Alberto Bemporad, Manfred Morari, Vivek Dua, and Efstratios N Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

Gerardo Bledt. *Regularized predictive control framework for robust dynamic legged locomotion*. PhD thesis, Massachusetts Institute of Technology, 2020.

Gerardo Bledt, Matthew J Powell, Benjamin Katz, Jared Di Carlo, Patrick M Wensing, and Sangbae Kim. MIT Cheetah 3: Design and Control of A Robust, Dynamic Quadruped Robot. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2245–2252. IEEE, 2018.

Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

Stephen Boyd, Laurent El Ghaoui, Eric Feron, and Venkataramanan Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*, volume 15. SIAM, 1994.

Stephen Boyd, Neal Parikh, and Eric Chu. *Distributed optimization and statistical learning via the alternating direction method of multipliers*. Now Publishers Inc, 2011.

G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

Bernard Brogliato. Some Perspectives on the Analysis and Control of Complementarity Systems. *IEEE Transactions on Automatic Control*, 48(6):918–935, 2003.

Bernard Brogliato. *Nonsmooth Mechanics: Models, Dynamics and Control*. Springer, 2016.

Bernard Brogliato and Aneel Tanwani. Dynamical Systems Coupled with Monotone Set-valued Operators: Formalisms, Applications, Well-posedness, and Stability. *Siam Review*, 62(1):3–129, 2020.

Bernard Brogliato and Lionel Thibault. Existence and Uniqueness of Solutions for Non-autonomous Complementarity Dynamical Systems. *Journal of Convex Analysis*, 17(3&4):961–990, 2010.

Bernard Brogliato, Rogelio Lozano, Bernhard Maschke, and Olav Egeland. Dissipative Systems Analysis and Control. *Theory and Applications*, 2, 2007.

M Kanat Çamlıbel, WPMH Heemels, and JM Schumacher. On Linear Passive Complementarity Systems. *European Journal of Control*, 8(3):220–237, 2002.

M Kanat Camlibel, Jong-Shi Pang, and Jinglai Shen. Lyapunov Stability of Complementarity and Extended Systems. *SIAM Journal on Optimization*, 17(4):1056–1101, 2006.

M Kanat Camlibel, Jong-Shi Pang, and Jinglai Shen. Lyapunov stability of complementarity and extended systems. *SIAM Journal on Optimization*, 17(4):1056–1101, 2007.

MK Çamlibel, WPMH Heemels, AJ van der Schaft, and JM Schumacher. Solution Concepts for Hybrid Dynamical Systems. In *IFAC World Congress*, 2002.

MK Camlibel et al. Complementarity Methods in the Analysis of Piecewise Linear Dynamical Systems. Technical report, Tilburg University, School of Economics and Management, 2001.

Abhishek Cauligi, Preston Culbertson, Bartolomeo Stellato, Dimitris Bertsimas, Mac Schwager, and Marco Pavone. Learning mixed-integer convex optimization strategies for robot planning and control. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 1698–1705. IEEE, 2020.

Abhishek Cauligi, Preston Culbertson, Edward Schmerling, Mac Schwager, Bartolomeo Stellato, and Marco Pavone. Coco: Online mixed-integer control via supervised learning. *IEEE Robotics and Automation Letters*, 7(2):1447–1454, 2021.

Kong Yao Chee, Thales C Silva, M Ani Hsieh, and George J Pappas. Enhancing sample efficiency and uncertainty compensation in learning-based model predictive control for aerial robots. *arXiv preprint arXiv:2308.00570*, 2023.

Chi-Tsong Chen. *Analog and digital control system design: transfer-function, state-space, and algebraic methods*. Oxford University Press, Inc., 1995.

Shaoru Chen, Mahyar Fazlyab, Manfred Morari, George J Pappas, and Victor M Preciado. Learning lyapunov functions for piecewise affine systems with neural network controllers. *arXiv preprint arXiv:2008.06546*, 2020.

Yu-Ming Chen and Michael Posa. Optimal reduced-order modeling of bipedal locomotion. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8753–8760. IEEE, 2020.

Xianyi Cheng, Eric Huang, Yifan Hou, and Matthew T Mason. Contact mode guided motion planning for quasidynamic dexterous manipulation in 3d. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2730–2736. IEEE, 2022.

Diego Cifuentes and Pablo A Parrilo. Sampling algebraic varieties for sum of squares programs. *SIAM Journal on Optimization*, 27(4):2381–2404, 2017.

Simon Le Cleac'h, Taylor Howell, Shuo Yang, Chi-Yen Lee, John Zhang, Arun Bishop, Mac Schwager, and Zachary Manchester. Fast contact-implicit model-predictive control, 2023.

Richard W Cottle, Jong-Shi Pang, and Richard E Stone. *The Linear Complementarity Problem*. SIAM, 2009.

Robin Deits, Twan Koolen, and Russ Tedrake. LVIS: Learning From Value Function Intervals for Contact-aware Robot Controllers. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7762–7768. IEEE, 2019.

Vishnu R Desaraju, Alexander Spitzer, and Nathan Michael. Experience-driven predictive control

with robust constraint satisfaction under time-varying state uncertainty. In *Robotics: Science and Systems*, 2017.

Steven Diamond, Reza Takapoui, and Stephen Boyd. A general system for heuristic minimization of convex functions over non-convex sets. *Optimization Methods and Software*, 33(1):165–193, 2018.

Yanran Ding, Abhishek Pandala, and Hae-Won Park. Real-time model predictive control for versatile dynamic motions in quadrupedal robots. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8484–8490. IEEE, 2019.

Steven P Dirkse and Michael C Ferris. The PATH Solver: A Non-monotone Stabilization Scheme for Mixed Complementarity Problems. *Optimization Methods and Software*, 5(2):123–156, 1995.

Elliott Donlon, Siyuan Dong, Melody Liu, Jianhua Li, Edward Adelson, and Alberto Rodriguez. GelSlim: A High-Resolution, Compact, Robust, and Calibrated Tactile-sensing Finger. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1927–1934. IEEE, 2018.

Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972a.

Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, jan 1972b. ISSN 0001-0782. doi: 10.1145/361237.361242. URL https://doi.org/10.1145/361237.361242.

Souradeep Dutta, Xin Chen, Susmit Jha, Sriram Sankaranarayanan, and Ashish Tiwari. Sherlock-a tool for verification of neural network feedback systems: demo abstract. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*, pages 262–263, 2019.

Jiameng Fan, Chao Huang, Xin Chen, Wenchao Li, and Qi Zhu. Reachnn*: A tool for reachability analysis of neural-network controlled systems. In *International Symposium on Automated Technology for Verification and Analysis*, pages 537–542. Springer, 2020.

Mahyar Fazlyab, Manfred Morari, and George J Pappas. Probabilistic verification and reachability analysis of neural networks via semidefinite programming. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 2726–2731. IEEE, 2019a.

Mahyar Fazlyab, Manfred Morari, and George J Pappas. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *arXiv preprint arXiv:1903.01287*, 2019b.

Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. In *Advances in Neural Information Processing Systems*, pages 11427–11438, 2019c.

Giancarlo Ferrari-Trecate, Francesco Alessandro Cuzzola, Domenico Mignone, and Manfred Morari. Analysis of discrete-time piecewise affine and hybrid systems. *Automatica*, 38(12):2139–2146, 2002.

Hiroaki Fukushima, Tae-Hyoung Kim, and Toshiharu Sugie. Adaptive model predictive control for a class of constrained linear systems based on the comparison model. *Automatica*, 43(2):301–308, 2007.

Carlos E Garcia, David M Prett, and Manfred Morari. Model predictive control: Theory and practice—a survey. *Automatica*, 25(3):335–348, 1989.

Ruben Grandia, Fabian Jenelten, Shaohui Yang, Farbod Farshidian, and Marco Hutter. Perceptive locomotion through nonlinear model predictive control. *arXiv preprint arXiv:2208.08373*, 2022.

Jacob W Guggenheim, Leif P Jentoft, Yaroslav Tenzer, and Robert D Howe. Robust and Inexpensive Six-axis Force–torque Sensors Using MEMS Barometers. *IEEE/ASME Transactions on Mechatronics*, 22(2):838–844, 2017.

Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2021. URL https://www.gurobi.com.

Rel Guzman, Rafael Oliveira, and Fabio Ramos. Adaptive model predictive control by learning classifiers. In *Learning for Dynamics and Control Conference*, pages 480–491. PMLR, 2022.

Mathew Halm. *ADDRESSING STIFFNESS-INDUCED CHALLENGES IN MODELING AND IDENTIFICATION FOR RIGID-BODY SYSTEMS WITH FRICTION AND IMPACTS*. PhD thesis, University of Pennsylvania, 2023.

Mathew Halm and Michael Posa. A quasi-static model and simulation approach for pushing, grasping, and jamming. In *International Workshop on the Algorithmic Foundations of Robotics*, pages 491–507. Springer, 2018.

Mathew Halm and Michael Posa. A Quasi-static Model and Simulation Approach for Pushing, Grasping, and Jamming. *arXiv preprint arXiv:1902.03487*, 2019.

Mathew Halm and Michael Posa. Set-valued rigid-body dynamics for simultaneous, inelastic, frictional impacts. *Under Review*, 2023.

WPMH Heemels, Johannes M Schumacher, and S Weiland. Linear Complementarity Systems. *SIAM Journal on Applied Mathematics*, 60(4):1234–1269, 2000.

James Hermus, Johannes Lachner, David Verdi, and Neville Hogan. Exploiting redundancy to facilitate physical interaction. *IEEE Transactions on Robotics*, 38(1):599–615, 2021.

Francois R Hogan and Alberto Rodriguez. Reactive planar non-prehensile manipulation with hybrid

model predictive control. *The International Journal of Robotics Research*, 39(7):755–773, 2020.

Francois R Hogan, Jose Ballester, Siyuan Dong, and Alberto Rodriguez. Tactile dexterity: Manipulation primitives with tactile feedback. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 8863–8869. IEEE, 2020.

Neville Hogan. Impedance control: An approach to manipulation: Part ii—implementation. 1985.

Robert D Howe. Tactile Sensing and Control of Robotic Manipulation. *Advanced Robotics*, 8(3): 245–261, 1993.

Wanxin Jin, Alp Aydinoglu, Mathew Halm, and Michael Posa. Learning linear complementarity systems. In *Learning for Dynamics and Control Conference*, pages 1137–1149. PMLR, 2022.

Mikael Johansson and Anders Rantzer. Computation of Piecewise Quadratic Lyapunov Functions for Hybrid Systems. In *1997 European Control Conference (ECC)*, pages 2005–2010. IEEE, 1997.

Mikael K-J Johansson. *Piecewise linear control systems: a computational approach*, volume 284. Springer, 2003.

Benjamin Karg and Sergio Lucia. Stability and feasibility of neural network-based controllers via output range analysis. *arXiv preprint arXiv:2004.00521*, 2020.

Hassan K Khalil. Nonlinear Systems. *Upper Saddle River*, 2002.

Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

Sangwoon Kim, Devesh K Jha, Diego Romeres, Parag Patre, and Alberto Rodriguez. Simultaneous tactile estimation and control of extrinsic contact. *arXiv preprint arXiv:2303.03385*, 2023.

Michal Kočvara and Michael Stingl. PENNON: A Code for Convex Nonlinear and Semidefinite Programming. *Optimization Methods and Software*, 18(3):317–333, 2003.

Jonas Koenemann, Andrea Del Prete, Yuval Tassa, Emanuel Todorov, Olivier Stasse, Maren Bennewitz, and Nicolas Mansard. Whole-body model-predictive control applied to the hrp-2 humanoid. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3346–3351. IEEE, 2015.

Peng Kou, Deliang Liang, and Lin Gao. Stochastic model predictive control for wind turbines with doubly fed induction generators. In *2016 IEEE Power and Energy Society General Meeting (PESGM)*, pages 1–5. IEEE, 2016.

Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning,

estimation, and control design for the atlas humanoid robot. *Autonomous robots*, 40(3):429–455, 2016.

Vikash Kumar, Yuval Tassa, Tom Erez, and Emanuel Todorov. Real-time behaviour synthesis for dynamic hand-manipulation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6808–6815. IEEE, 2014.

Vikash Kumar, Emanuel Todorov, and Sergey Levine. Optimal Control with Learned Local Models: Application to Dexterous Manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 378–383. IEEE, 2016.

Vince Kurtz and Hai Lin. Contact-implicit trajectory optimization with hydroelastic contact and ilqr. *arXiv preprint arXiv:2202.13986*, 2022.

Yuri A Kuznetsov, Iu A Kuznetsov, and Y Kuznetsov. *Elements of applied bifurcation theory*, volume 112. Springer, 1998.

Fabien Lauer. On the complexity of piecewise affine system identification. *Automatica*, 62:148–153, 2015.

He Li, Robert J Frei, and Patrick M Wensing. Model hierarchy predictive control of robotic systems. *IEEE Robotics and Automation Letters*, 6(2):3373–3380, 2021.

Hai Lin and Panos J Antsaklis. Stability and Stabilizability of Switched Linear Systems: A Survey of Recent Results. *IEEE Transactions on Automatic Control*, 54(2):308–322, 2009.

J. Löfberg. YALMIP : A Toolbox for Modeling and Optimization in MATLAB. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

Johan Lofberg. Yalmip: A toolbox for modeling and optimization in matlab. In *2004 IEEE international conference on robotics and automation (IEEE Cat. No. 04CH37508)*, pages 284–289. IEEE, 2004.

Yudong Ma, Sergey Vichik, and Francesco Borrelli. Fast stochastic mpc with optimal risk allocation applied to building control systems. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 7559–7564. IEEE, 2012.

Zachary Manchester and Scott Kuindersma. Variational contact-implicit trajectory optimization. In *Robotics Research: The 18th International Symposium ISRR*, pages 985–1000. Springer, 2020.

Tobia Marcucci and Russ Tedrake. Warm Start of Mixed-Integer Programs for Model Predictive Control of Hybrid Systems. 2019.

Tobia Marcucci and Russ Tedrake. Warm start of mixed-integer programs for model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, 2020.

Tobia Marcucci, Robin Deits, Marco Gabiccini, Antonio Bicchi, and Russ Tedrake. Approximate Hybrid Model Predictive Control for Multi-contact Push Recovery in Complex Environments. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 31–38. IEEE, 2017.

Carlos Mastalli, Rohan Budhiraja, Wolfgang Merkt, Guilhem Saurel, Bilal Hammoud, Maximilien Naveau, Justin Carpentier, Ludovic Righetti, Sethu Vijayakumar, and Nicolas Mansard. Crocoddyl: An Efficient and Versatile Framework for Multi-Contact Optimal Control. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2536–2542. IEEE, 2020.

Hamza Merzic, Miroslav Bogdanovic, Daniel Kappler, Ludovic Righetti, and Jeannette Bohg. Leveraging Contact Forces for Learning to Grasp. *arXiv preprint arXiv:1809.07004*, 2018.

Felix A Miranda-Villatoro, Fulvio Forni, and Rodolphe Sepulchre. Dominance analysis of linear complementarity systems. *arXiv preprint arXiv:1802.00284*, 2018.

Irinel Constantin Morărescu and Bernard Brogliato. Trajectory Tracking Control of Multiconstraint Complementarity Lagrangian Systems. *IEEE Transactions on Automatic Control*, 55(6):1300–1313, 2010.

APS Mosek. The MOSEK Optimization Software. *Online at http://www.mosek.com*, 54(2-1):5, 2010.

Richard M Murray and John Edmond Hauser. *A Case Study in Approximate Linearization: The Acrobat Example*. 1991.

Philippe Nadeau, Michael Abbott, Dominic Melville, and Hannah S Stuart. Tactile sensing based on fingertip suction flow for submerged dexterous manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3701–3707. IEEE, 2020.

Miquel Oller, Mireia Planas i Lisbona, Dmitry Berenson, and Nima Fazeli. Manipulation via membranes: High-resolution and highly deformable tactile sensing and control. In *Conference on Robot Learning*, pages 1850–1859. PMLR, 2023.

Aykut Özgun Önol, Philip Long, and Taşkın Padır. Contact-implicit trajectory optimization based on a variable smooth contact model and successive convexification. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2447–2453. IEEE, 2019.

Brendan O'donoghue, Eric Chu, Neal Parikh, and Stephen Boyd. Conic optimization via operator splitting and homogeneous self-dual embedding. *Journal of Optimization Theory and Applications*, 169:1042–1068, 2016.

Jong-Shi Pang and David E Stewart. Differential variational inequalities. *Mathematical programming*, 113(2):345–424, 2008.

Tao Pang, H. J. Terry Suh, Lujie Yang, and Russ Tedrake. Global planning for contact-rich manipulation via local smoothing of quasi-dynamic contact models, 2023.

A Papachristodoulou and S Prajna. Robust Stability Analysis of Nonlinear Hybrid Systems. *IEEE Transactions on Automatic Control*, 54(5):1035–1041, May 2009.

Jaehyun Park and Stephen Boyd. General heuristics for nonconvex quadratically constrained quadratic programming. *arXiv preprint arXiv:1703.07870*, 2017.

Juyong Park, Jaeyoung Haan, and Frank C Park. Convex optimization algorithms for active balancing of humanoid robots. *IEEE Transactions on Robotics*, 23(4):817–822, 2007.

Pablo A Parrilo. Semidefinite Programming Relaxations for Semialgebraic Problems. *Mathematical programming*, 96(2):293–320, 2003.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Karime Pereida and Angela P Schoellig. Adaptive model predictive control for high-accuracy trajectory tracking in changing conditions. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7831–7837. IEEE, 2018.

Alberto Del Pia, Santanu S Dey, and Marco Molinaro. Mixed-integer quadratic programming is in np. *Mathematical Programming*, 162:225–240, 2017.

Michael Posa, Cecilia Cantu, and Russ Tedrake. A Direct Method for Trajectory Optimization of Rigid Bodies Through Contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014.

Michael Posa, Mark Tobenkin, and Russ Tedrake. Stability Analysis and Control of Rigid-body Systems with Impacts and Friction. *IEEE Transactions on Automatic Control*, 61(6):1423–1437, 2015.

Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018.

Arvind U Raghunathan, Devesh K Jha, and Diego Romeres. Pyrobocop: Python-based robotic control & optimization package for manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 985–991. IEEE, 2022.

Joseph M Romano, Kaijen Hsiao, Günter Niemeyer, Sachin Chitta, and Katherine J Kuchenbecker. Human-inspired Robotic Grasp Control with Tactile Sensing. *IEEE Transactions on Robotics*, 27(6):1067–1079, 2011.

Stefan Scholtes. *Introduction to piecewise differentiable equations*. Springer Science & Business Media, 2012.

Jinglai Shen and Jong-Shi Pang. Semicopositive Linear Complementarity Systems. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal*, 17(15):1367–1386, 2007.

Yuki Shirai, Xuan Lin, Alexander Schperberg, Yusuke Tanaka, Hayato Kato, Varit Vichathorn, and Dennis Hong. Simultaneous contact-rich grasping and locomotion via distributed optimization enabling free-climbing for multi-limbed robots. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 13563–13570. IEEE, 2022.

Chelsea Sidrane and Mykel J Kochenderfer. Overt: Verification of nonlinear dynamical systems with neural network controllers via overapproximation. In *Safe Machine Learning workshop at ICLR*, 2019.

Jean-Pierre Sleiman, Farbod Farshidian, Maria Vittoria Minniti, and Marco Hutter. A unified mpc framework for whole-body dynamic locomotion and manipulation. *IEEE Robotics and Automation Letters*, 6(3):4688–4695, 2021.

Georgi V Smirnov. *Introduction to The Theory of Differential Inclusions*, volume 41. American Mathematical Soc., 2002.

Charis Stamouli, Anastasios Tsiamis, Manfred Morari, and George J Pappas. Adaptive stochastic mpc under unknown noise distribution. In *Learning for Dynamics and Control Conference*, pages 596–607. PMLR, 2022.

B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020. doi: 10.1007/s12532-020-00179-2. URL https://doi.org/10.1007/s12532-020-00179-2.

Gilbert Stengle. A Nullstellensatz and A Positivstellensatz In Semialgebraic Geometry. *Mathematische Annalen*, 207(2):87–97, 1974.

David Stewart and Jeffrey C Trinkle. An Implicit Time-stepping Scheme for Rigid Body Dynamics with Coulomb Friction. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 162–169. IEEE, 2000.

David E Stewart. Rigid-body Dynamics with Friction and Impact. *SIAM review*, 42(1):3–39, 2000.

Jos F Sturm. Using SeDuMi 1.02, A MATLAB Toolbox for Optimization over Symmetric Cones. *Optimization Methods and Software*, 11(1-4):625–653, 1999.

Neha Sunil, Shaoxiong Wang, Yu She, Edward Adelson, and Alberto Rodriguez Garcia. Visuotactile affordances for cloth manipulation with local control. In *Conference on Robot Learning*, pages

1596–1606. PMLR, 2023.

Markku Suomalainen, Yiannis Karayiannidis, and Ville Kyrki. A survey of robot manipulation in contact. *Robotics and Autonomous Systems*, 156:104224, 2022.

Marko Tanaskovic, Lorenzo Fagiano, Roy Smith, and Manfred Morari. Adaptive receding horizon control for constrained mimo systems. *Automatica*, 50(12):3019–3029, 2014.

Aneel Tanwani, Bernard Brogliato, and Christophe Prieur. Well-posedness and output regulation for implicit time-varying evolution variational inequalities. *SIAM Journal on Control and Optimization*, 56(2):751–781, 2018.

Yuval Tassa, Tom Erez, and Emanuel Todorov. Synthesis and Stabilization of Complex Behaviors Through Online Trajectory Optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4906–4913. IEEE, 2012.

Ian H Taylor, Siyuan Dong, and Alberto Rodriguez. Gelslim 3.0: High-resolution measurement of shape, force and slip in a compact tactile-sensing finger. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10781–10787. IEEE, 2022.

Russ Tedrake and the Drake Development Team. Drake: Model-based design and verification for robotics, 2019. URL https://drake.mit.edu.

Stephen Tian, Frederik Ebert, Dinesh Jayaraman, Mayur Mudigonda, Chelsea Finn, Roberto Calandra, and Sergey Levine. Manipulation by Feel: Touch-based Control with Deep Predictive Models. *arXiv preprint arXiv:1903.04128*, 2019.

Hoang-Dung Tran, Xiaodong Yang, Diego Manzanas Lopez, Patrick Musau, Luan Viet Nguyen, Weiming Xiang, Stanley Bak, and Taylor T Johnson. Nnv: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In *International Conference on Computer Aided Verification*, pages 3–17. Springer, 2020.

Lieven Vandenberghe and Stephen Boyd. Semidefinite programming. *SIAM review*, 38(1):49–95, 1996.

Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Efficient formal safety analysis of neural networks. In *Advances in Neural Information Processing Systems*, pages 6367–6377, 2018.

Avishai Weiss and Stefano Di Cairano. Robust dual control mpc with guaranteed constraint satisfaction. In *53rd IEEE Conference on Decision and Control*, pages 6713–6718. IEEE, 2014.

Patrick M Wensing, Michael Posa, Yue Hu, Adrien Escande, Nicolas Mansard, and Andrea Del Prete. Optimization-based control for dynamic legged robots. *arXiv preprint arXiv:2211.11644*, 2022.

Alexander W Winkler, C Dario Bellicoso, Marco Hutter, and Jonas Buchli. Gait and Trajectory Optimization for Legged Systems Through Phase-based End-effector Parameterization. *IEEE Robotics and Automation Letters*, 3(3):1560–1567, 2018.

Akihiko Yamaguchi and Christopher G Atkeson. Combining Finger Vision and Optical Tactile Sensing: Reducing and Handling Errors While Cutting Vegetables. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 1045–1051. IEEE, 2016.

He Yin, Peter Seiler, and Murat Arcak. Stability analysis using quadratic constraints for systems with neural network controllers. *arXiv preprint arXiv:2006.07579*, 2020.

Wenzhen Yuan, Rui Li, Mandayam A Srinivasan, and Edward H Adelson. Measurement of Shear and Slip with a GelSight Tactile Sensor. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 304–311. IEEE, 2015.

Altay Zhakatayev, Bexultan Rakhim, Olzhas Adiyatov, Almaskhan Baimyshev, and Huseyin Atakan Varol. Successive linearization based model predictive control of variable stiffness actuated robots. In *2017 IEEE international conference on advanced intelligent mechatronics (AIM)*, pages 1774–1779. IEEE, 2017.

Huaijiang Zhu and Ludovic Righetti. Efficient object manipulation planning with monte carlo tree search. *arXiv preprint arXiv:2206.09023*, 2022.