UNIVERSITY OF CALIFORNIA
Los Angeles

Optimization-based Planning and Control for Robust and Dexterous Locomotion and Manipulation through Contact

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Mechanical Engineering

by

Yuki Shirai

2024

© Copyright by
Yuki Shirai
2024

ABSTRACT OF THE DISSERTATION

Optimization-based Planning and Control for Robust and Dexterous Locomotion and Manipulation through Contact

by

Yuki Shirai
Doctor of Philosophy in Mechanical Engineering
University of California, Los Angeles, 2024
Professor Dennis W. Hong, Chair

Although robotic locomotion and manipulation have shown some remarkable progress in the real world, the current locomotion and manipulation algorithms are inefficient in performance. They often only work for relatively simple tasks such as walking and running for locomotion and pick-and-place in structured environments (e.g., factory) for manipulation. In contrast, humans can perform quite dexterous tasks through *contact* as contacts provide additional dexterity to interact with environments. Hence, understanding the underlying contact mechanics plays a key role in designing contact-aware planners, controllers, and estimators for locomotion and manipulation.

However, design for planners, controllers, and estimators is extremely challenging. First, the number of contact states such as making and breaking contact with environments increases dramatically as the number of contacts increases. Thus, the underlying contact dynamics become large-scale non-smooth dynamics. As a result, optimization solvers have difficulties converging due to the non-convexity of the optimization problem.

Second, it is desirable that a robot should be able to interact in unknown environments during operation, leading to generalizable locomotion and manipulation. However, robust planning with frictional interaction with uncertain physical properties is very tough as the robot might cause undesired unexpected contact events. As a result, a robot might not be able to complete its desired task.

Third, once uncertainty is quite large, it is indispensable for closed-loop controllers to stabilize locomotion and manipulation. However, the design of manipulation is quite challenging as most manipulation systems are underactuated and unobservable with potential changes in contact states and modes.

In this dissertation, we present a methodology for contact-rich locomotion and planning using trajectory optimization. We first show that the planner using graph-search planners with trajectory optimization can be beneficial for decreasing the computation complexity. Second, we describe our contact-implicit trajectory optimization for planning of multi-limbed systems for running and climbing. We use decomposition-based optimization techniques to efficiently design a trajectory for a robot subject to various complicated contact constraints such as mixed-integer constraints. Then, we present our robust and stochastic trajectory optimization algorithms for multi-contact systems. We show that our chance-constrained optimization is applicable for planning multi-limbed robots. We also propose covariance steering algorithm for contact-rich systems using a particle filter to approximate a distribution of underlying contact dynamics. Our covariance steering is able to regulate robots' states and contact states simultaneously with probabilistic guarantees. Furthermore, utilizing the underlying structure of contact-rich manipulation, we present robust bilevel trajectory optimization for pivoting manipulation under uncertain physical parameters such as friction coefficients. Our proposed framework is able to design optimal control sequences while improving the worst-case stability margin along the manipulation. Finally, we present our closed-loop controller framework for tool manipulation using visuo-tactile feedback. Our approach enables the robot to achieve tool manipulation under unexpected contact events in closed-loop control fashion with no visual feedback for partially unknown objects.

The perspectives gained from this dissertation provide better insight into developing a contact-rich planning, estimation, and control framework for dexterous locomotion and manipulation in highly unstructured environments.

The dissertation of Yuki Shirai is approved.

Brett Thomas Lopez

Jacob Rosen

Lieven Vandenberghe

Dennis W. Hong, Committee Chair

University of California, Los Angeles

2024

Think Like an Amateur, Do As an Expert

Takeo Kanade

TABLE OF CONTENTS

List of Figures . . . . **xiv**

List of Tables . . . . **xxviii**

Acknowledgments . . . . **xxxii**

Vita . . . . **xxxvi**

1 Introduction **1**

1.1 Motivation 1

1.2 Contributions 2

1.2.1 Contribution 1: Planning through Contact 3

1.2.2 Contribution 2: Robust Planning through Contact 4

1.2.3 Contribution 3: Closed-Loop Control through Contact 6

1.3 Outline of Dissertation 7

2 Research Challenges and Related Works **8**

2.1 Optimization-based Planning through Contact 8

2.2 Optimization-based Robust Planning through Contact 10

2.2.1 Robust Optimization and Stochastic Optimization 10

2.2.2 Stochastic Optimization for Contact-Rich Systems 11

2.2.3 Robust Optimization for Contact-Rich Systems 13

2.3 Closed-Loop Control through Contact 14

2.3.1 Covariance Steering: Control Policy with Probabilistic Guarantees for Contact-Rich Systems 15

2.3.2 Contact Estimation and Control with Visuotactile Sensors for Reactive Manipulation . . . 16

3 Lazy Trajectory Optimization . . . 18

3.1 Overview . . . 18

3.2 Related Works . . . 20

3.3 Problem Formulation . . . 22

3.3.1 Notation . . . 22

3.3.2 Graph Structure . . . 22

3.3.3 Mixed-Integer Convex Programs . . . 25

3.3.4 Warm-Start Strategy . . . 25

3.4 Lazy Trajectory Optimization Formulation . . . 26

3.4.1 Trajectory Optimization-Aware Cost . . . 27

3.4.2 Main Loop (Algorithm 1) . . . 28

3.4.3 Expansion (Algorithm 2) . . . 28

3.4.4 Edge generation (Algorithm 3) . . . 29

3.4.5 Vertex Validation (Algorithm 4) . . . 30

3.5 Formal Analysis . . . 31

3.5.1 Complexity . . . 31

3.5.2 Efficiency . . . 33

3.5.3 Completeness and Optimality . . . 33

3.6 Numerical Experiments . . . 34

3.6.1 Free-Flying Robots . . . 35

3.6.2 Legged Robots . . . 38

3.7 Conclusion . . . 39

4 Planning through Contact using Decomposition-based Optimization for Multi-Limbed Robots . . . 42

4.1 Overview . . . 42

4.2 Patch Contact Model with Micro-Spines . . . 45

4.2.1 Frictional Limit Surface . . . 45

4.2.2 Limit Surface of Micro-Spines . . . 45

4.2.3 Limit Surface for Two-Finger Micro-Spine Grippers . . . 46

4.3 Simultaneous Contact-Rich Grasping and Locomotion Trajectory Optimization 46

4.3.1 Preliminary . . . 47

4.3.2 Optimal Control Problem for Grasping and Locomotion . . . 48

4.3.3 Alternating Direction Method of Multipliers (ADMM) . . . 53

4.3.4 Decomposition-based Optimal Control Problem . . . 54

4.3.5 Multi-Block Decomposition-based Optimal Control Problem . . . 56

4.4 Experimental Results . . . 56

4.4.1 Implementation Details . . . 56

4.4.2 Computation Results . . . 57

4.4.3 Results of Our Generated Trajectories . . . 60

4.4.4 Contact Modeling Results . . . 62

4.5 Discussion . . . 64

5 Risk-Aware Motion Planning for Multi-Limbed Robots . . . 66

5.1 Overview . . . 66

5.2 Problem Formulation . . . 68

5.2.1 Friction Cone with Stochastic Gripping Forces . . . 68

5.2.2 Model of Reaction Force Using Limb Compliance . . . 70

5.2.3 Model of Gripping Force Using Gaussian Process Regression 70
5.2.4 Spine Gripper for Wall Climbing 72
5.2.5 Data Collection 73
5.3 Chance-Constrained Nonlinear Programming for Locomotion 74
5.3.1 Deterministic Constraints 77
5.3.2 Chance Constraints 77
5.3.3 Cost Function 79
5.3.4 Two Step Optimization for a Position-Controlled Robot 80
5.4 Results 82
5.4.1 Energy Efficient Planning 83
5.4.2 Climbing on Non-Uniform Walls 84
5.4.3 Climbing with Less Stable Gait: Tripod Gait 86
5.5 Conclusion 87

6 Chance-Constrained Optimization for Uncertain Contact-Rich Systems 89

6.1 Overview 90
6.2 Problem Preliminary 92
6.2.1 Discrete-time Linear Complementarity System (DLCS) 93
6.2.2 Contact-Implicit Trajectory Optimization 94
6.2.3 Stochastic Discrete-time Linear Complementarity Systems (SDLCS) . 94
6.3 Robust Trajectory Optimization for SDLCS 95
6.3.1 Joint Linear Chance Constraints 96
6.3.2 Chance Complementarity Constraints (CCC) for SDLCS 98
6.3.3 Mixed-Integer Quadratic Programming with Chance Constraints . . . 101
6.3.4 Stochastic Model Predictive Control with Complementarity Constraints 102

6.4 Numerical Simulations 104
6.4.1 Implementation Details 104
6.4.2 Example Details 105
6.4.3 Robustness of Open-Loop Trajectories 108
6.4.4 Monte Carlo Simulation Results 110
6.4.5 Computation Results 112
6.4.6 Discussion of Assumptions 113
6.5 SMPC for Planar Pushing 116
6.5.1 Planar Pushing 116
6.5.2 Results 118
6.6 Discussion and Conclusion 119

7 Robust Pivoting Manipulation using Contact Implicit Bilevel Optimization 121
7.1 Overview 122
7.2 Mechanics of Pivoting 124
7.2.1 Mechanics of Pivoting with External Contacts 126
7.3 Robust Pivoting Formulation 128
7.3.1 Frictional Stability Margin 129
7.3.2 Stability Margin for Uncertain Mass 130
7.3.3 Stability Margin for Uncertain CoM Location 131
7.3.4 Stability Margin for Stochastic Friction 132
7.3.5 Stability Margin for Finger Contact Location 134
7.3.6 Stability Margin for Uncertain Mass on a Slope 135
7.3.7 Pivoting with Patch Contact between the object and the manipulator 136

7.4 Robust Trajectory Optimization 138

7.4.1 Contact-Implicit Trajectory Optimization for Pivoting 139

7.4.2 Robust Bilevel Contact-Implicit Trajectory Optimization 139

7.4.3 Robust Bilevel Contact-Implicit Trajectory Optimization under Frictional Uncertainty 143

7.4.4 Robust Bilevel Optimization over Mode Sequences for Non-Convex Objects 144

7.4.5 Robust Bilevel Contact-Implicit Trajectory Optimization with Patch Contact 146

7.5 Experimental Results 147

7.5.1 Experiment Setup 149

7.5.2 Results of Bilevel Optimization for Uncertain Mass and CoM Parameters 150

7.5.3 Results of Bilevel Optimization with Different Manipulator Initial State 152

7.5.4 Results of Bilevel Optimization for Uncertain CoM parameters with Different Mass and Friction of Object 153

7.5.5 Results of Bilevel Optimization for Uncertain Friction Parameters . . 156

7.5.6 Results of Bilevel Optimization for Uncertain Finger Contact Location 156

7.5.7 Results of Bilevel Optimization over Mode Sequences for Non-Convex Objects 157

7.5.8 Results of Bilevel Optimization for Uncertain Mass on a Slope 158

7.5.9 Results of Bilevel Optimization for Patch Contact 159

7.5.10 Computation Results 160

7.5.11 Hardware Experiments 161

7.6 Discussion and Future Work 163

8 Covariance Steering for Uncertain Contact-Rich Systems 165

8.1 Overview 166
8.2 Problem Formulation 167
    8.2.1 Stochastic Discrete-time Linear Complementarity Systems 167
    8.2.2 Stochastic Control for Contact-Rich Systems 168
8.3 Covariance Steering for Contact-Rich Systems 168
    8.3.1 Particle-based Control for Contact-Rich Systems 169
    8.3.2 Bilevel Optimization for Particle-based Control 171
    8.3.3 Important-particle Method for Particle-based Control 172
8.4 Results 174
    8.4.1 Uncertainty Propagation for SDLCS 174
    8.4.2 Cartpole with Softwalls 175
    8.4.3 Acrobot with Soft Joints 178
8.5 Discussion 180

9 Closed-Loop Tactile Control for Tool Manipulation 182
9.1 Overview 182
9.2 Mechanics of Tool Manipulation 184
    9.2.1 Quasi-Static Mechanics of Tool Manipulation 185
    9.2.2 Contact Model 186
    9.2.3 Contact between Tool and Object 187
    9.2.4 Trajectory Optimization for Planning 188
9.3 Tactile Tool Manipulation 189
    9.3.1 Tactile Stiffness Regression 189
    9.3.2 Tactile Estimator 189
    9.3.3 Tactile Controller 191

9.4 Results . . . 191

9.4.1 Experiment Setup . . . 193

9.4.2 Open-Loop Controller . . . 193

9.4.3 Tactile Estimator and Controller . . . 194

9.5 Conclusion . . . 198

10 Conclusion . . . 199

10.1 Summary of Contributions . . . 199

10.2 Limitations and Future Works . . . 201

10.2.1 Computational Complexity of ADMM . . . 201

10.2.2 Propagation of Uncertainty for SDLCS Analytically . . . 201

10.2.3 Contact-Rich CIBO . . . 201

10.2.4 Accurate Contact Mechanics . . . 202

10.2.5 Analysis of Controllability and Observability for Dexterous Manipulation 202

10.2.6 Hysteresis of Visuotactile Sensors . . . 202

10.2.7 Learning Manipulation Skills with Model-based Optimization . . . 203

Bibliography . . . 204

List of Figures

1.1 We categorize our contributions using two axes. The horizontal axis shows if the proposed framework considers the change of contact modes (e.g., braking-making contact. See Fig. 2.1) in the proposed framework. The vertical axis shows if the framework uses sensor measurements (i.e., open-loop (planner) or closed-loop control (controller). See Fig. 1.2). 2

1.2 We study contact-rich planners and controllers for different contact-rich systems from multi-limbed robots for climbing to tool manipulation for pivoting. x and u represent states and control inputs, respectively. The subscripts r, c, m, e represent reference, command, measurements, and estimates of x and u, respectively. Planners (e.g., trajectory optimization algorithms in Chapter 3, Chapter 4, Chapter 5, Chapter 6, Chapter 7)) do not use sensor measurements or estimates of the states. On the other hand, controllers (e.g., covariance steering in Chapter 8 and MPC in Chapter 9) use sensor measurements or estimates of the states. 3

2.1 Conceptual drawing of different contact modes. In (a), the robot makes non-zero contact forces when making contact. Otherwise, it makes zero contact forces. In (b), the contact does not slip when the contact force is not on the edge of the friction cones. Otherwise, it slips depending on the direction of the contact force. 9

3.1 Overview of LTO. Given an initial graph where the true configuration of the robot and the trajectory are unknown, LTO iteratively solves TO locally to obtain the true vertex (configuration) and edges (trajectories) in $\mathcal{G}$. Based on the updated $\mathcal{G}$ by TO, GSP performs either expansion of the vertex, gets the true vertex using TO, or gets the true edge using TO. Assume the vertex P is the current vertex LTO chooses from an open list. In (a), LTO gets the true configuration of the vertex C using TO. In (b), it generates the true trajectory from the vertex P to C using TO. In (c), it expands the vertex C and inserts the vertices A and B into the open list. 23

3.2 The planning procedure by LTO in $[0, 1]^2$ where $K = 4, i = 1, r = 1/4$. The left figure shows a graph at the start of planning where each vertex is inside the orange voxel and each edge is represented as a black dashed line. When LTO intends to expand the vertex, TO is solved within this voxel. When LTO investigates the edge, it solves TO within the associated voxels. For instance, when it investigates the edge from the vertex (a) to (d), it solves TO in the voxels (a), (b), (c), (d), taking into accout the constraints (e.g., obstacles) in the voxels. The right figure shows the current graph structure after iterations. For simplicity, it does not show the black dashed lines. The true vertex configurations and the true edges found by TO are shown as the black circles and blue lines, respectively. The infeasible vertex and edge judged by TO are shown as the red circle and red lines, respectively. For example, the vertex (e) is removed since TO cannot find the feasible vertex configuration in the voxel (e). 24

3.3 Generated trajectories in $\mathbb{R}^2$. The short-horizon TO gets stuck at the local optimum and cannot find the trajectory from start to goal. The long-horizon TO finds the optimal trajectory but takes an extended amount of time. LTO prioritizing planning time avoids the area where the time for solving TO is long due to many integer variables (i.e., obstacles). LTO prioritizing optimality of the trajectory finds the resolution-optimal solution with less planning time compared with the long-horizon TO. 36

3.4 The results of Section 3.6.1. Error bars represent a 95 % confidence interval for a Gaussian distribution. Note that for some algorithms, the confidence intervals are very small and are not visible. Compared with TO, LTO prioritizing optimality (i.e., $\omega = 0$) finds the optimal solution about nine times faster without sacrificing the solution cost much (1.2 % worse). 37

3.5 Consumed time with 2000 voxels in $\mathbb{R}^2$ for LTO. The larger the inflation factor ω is, the less time LTO spends by avoiding expensive edge generation. 37

3.6 The results of Section 3.6.2 in $\mathbb{R}^{21}$. The red lines in the top figure indicate the body trajectory. Error bars represent a 95 % confidence interval for a Gaussian distribution. Note that for LTO, the confidence intervals are very small and are not visible. LTO with 6000 voxels with the warm-start finds the optimal solution about 11.3, 14.0, 1.4 times faster than TO without degrading the solution cost so much (about 0.33, 0.35, 0.01 % worse), from the left to the right environment, respectively. By increasing the inflation factor ω, LTO can generate globally suboptimal trajectories faster than TO feasible option. 40

4.1 We consider motion planning of multi-limbed robots for free-climbing. Our proposed framework efficiently generates trajectories for multi-limbed robots equipped with multi-finger grippers while considering locomotion, grasping, and contacts. The left figure shows trajectories of one of the fingers of each gripper while the robot avoids obstacles. The trajectories around A-E are discussed in Sec 4.4.3. The right figure shows that our real four-limbed robot executes our planned trajectory. 43

4.2 Mathematical model of a multi-limbed robot. In this example, $n_f = 4, n_l = 2, C = 4, V = 2$. We also visualize two examples of frames $\Sigma_W, \Sigma_{c=3}$ where z-axis of Σ_W is perpendicular to the ground and z-axis of the local frame $\Sigma_{c=3}$ is perpendicular to the face of the climbing hold and along this axis we have non-zero z element of m. The spine makes contact with the contact angle (a): $\phi = \frac{\pi}{3}$ and (b): $\phi = \frac{\pi}{2}$. 48

4.3 We propose two decomposition-based optimization based on ADMM specific for motion planning of limbed robots. (top): Two-block ADMM where MIQP considers discrete constraints and NLP considers nonlinear constraints so that the planner effectively solves the MINLP once these two optimization problems achieve consensus. (bottom): Multi-block ADMM where the planner consists of n_l MIQP problems and one NLP. 54

4.4 Evolution of residuals for walking using two- and five-block ADMM. (a): Residual of body and finger positions, (b): residual of finger forces. 58

4.5 Evolution of residuals for free-climbing using two- and five-block ADMM. We show residual of (a): body and finger positions, (b): finger forces, (c): body rotation, (d): finger moments. 58

4.6 Change of modes as our ADMM proceeds with snapshots of hardware experiments. (top): after 1 iteration of our ADMM, the planner generates a trot gait, which is physically infeasible since MIQP is not fully influenced by nonlinear constraints yet. (bottom): after 6 iterations, the planner finds a physically feasible one leg gait sequence ($1 \rightarrow 4 \rightarrow 2 \rightarrow 3$). 61

4.7 Our planned trajectories on holds with varying coefficients of frictions. The robot grasps the faces whose coefficients of friction are high. 62

4.8 Time history of reaction forces for one finger. Here, we enforce z element of reaction force in Σ_{C_c} is always zero with $f^i_{\max} = 4$ N in (4.3). 63

4.9 Time history of wrench trajectories between the trajectory (a): considering patch constraints and (b): not considering patch constraints. 64

5.1 A planned trajectory for wall climbing that considers risk arising from slippery terrain. The black area shows a high friction area, the green area shows a low friction area, and the red area shows a zero friction area. Blue and red dots show the planned foot positions, and the hexagons show the body of the robot. 68

5.2 Deflection of a multi-limbed robot bracing between walls 69

5.3 Friction cone considering stochastic gripping forces. 70

5.4 Mechanical design of the spine gripper 71

5.5 Experiment setup to evaluate maximum gripping forces on sandpaper 73

5.6 The predicted maximum gripping force PDF from GP, the training data PDF, and the testing dataset 74

5.7 Time history of the consumed power under the different violation probabilities. The shaded regions are when the robot lifts a specific limb and puts it on the next position, and white regions are when the robot pushes its body up. The figure shows that the consumed power of a particular limb decreases when the limb is in the air, while it increases when the limb is on the wall to generate the normal force on the wall. 84

5.8 Consumed energy with different Δ during $t = 0 - 15$ s 85

5.9 Side view of planned footsteps on non-uniform walls under: left $\Delta = 0.1$, right $\Delta = 0.001$. In the left panel, the robot puts its feet on low and high friction terrain by taking a high risk bound. In the right panel, the robot puts its feet only on high friction terrain. 86

5.10 Snapshots of climbing experiments on non-uniform walls under the different violation probabilities 87

5.11 Climbing with tripod gait. Left: A planned trajectory of the tripod gait under $\Delta = 0.4$. Red arrows indicate the reaction forces from the walls. Right: A snapshot of climbing experiments with the tripod gait under $\Delta = 0.4$. 88

6.1 This chapter presents chance-constrained optimization for SDLCS. The figure shows the case of stochastic planar pushing with uncertain dynamics. Note that w and v are additive uncertainty terms. We show a MIP-based stochastic MPC formulation for control of stochastic planar pushing system. 91

6.2 Deterministic and stochastic complementarity constraints. We have the complementarity constraints $0 \leq \lambda_{k+1,i} \perp y_{k+1,i} \geq 0$ where $y_{k+1,i}$ has uncertainty and accepts the violation of ϵ. 99

6.3 Problems described in Section 6.4. 106

6.4 Results with different Δ for the cartpole with softwalls system. First, the cart moves in the negative direction to utilize the contact force λ_2 because the control input is bounded. Once the cart obtains enough λ_2, the cart is accelerated in the positive direction. We can observe the effect of our proposed chance constraints in particular around $t \in [0, 0.1]$ and $t \in [0.4, 0.5]$. When $t \in [0, 0.1]$, the mode changes from the "contact on the wall 2" to the "no contact" and the cart tries to be far from wall 2 to satisfy the CCC. When $t \in [0.4, 0.5]$, the trajectories are farther away from $x_1 = 0.05$ and $x_2 = 0.15$ as Δ decreases. 109

6.5 Results with different Δ for the sliding box with friction example. First, the box is accelerated in the positive direction. Then, the control decreases with time to regulate the box around the origin by employing friction forces. We can observe the effect of our proposed chance constraints in particular around $t \in [0.2, 0.3]$ where the trajectories are farther away from $x = 0.88$ as Δ decreases. 109

6.6 Results with dual manipulation. First, the box is pushed by the right arm in the negative direction. Next, the left arm regulates the box to the origin. In particular, around $t \in [0.2, 0.3]$ s, the trajectories are farther away from $x_1 = -0.17$ m as Δ decreases. 110

6.7 Simulated trajectories of x_2 of the cartpole example over 1000 samples with $\Delta = 0.5$ for the left column and with $\Delta = 0.02$ for the right column. The bottom row enlarges the top row figures around the area where the chance constraints effect is observed. The red line shows the 99.9 % confidence interval. 113

6.8 Simulated trajectories of x of sliding box with friction example over 1000 samples with $\Delta = 0.5$ for the left column and with $\Delta = 0.002$ for the right column. The bottom row enlarges the top row figures around the area where the chance constraints effect is observed. The red line shows the 99 % confidence interval. . 114

6.9 Simulated trajectories of x_1 over 1000 samples with $\Delta = 0.2$ for dual manipulation. The right figure shows the enlarged figure of the left figure. The red line shows the 99 % confidence interval. 115

6.10 Trajectories for the stochastic cartpole system obtained by using 1000 samples for the uncertain parameters while the control input is set to that computed using our proposed optimization (6.15) where the uncertainty values correspond to those used in Section 6.4.2.1. (a): simulated trajectories of x_2 with uncertain $\frac{1}{k_1}, \frac{1}{k_2}$ for which standard deviations are 10^{-6}. (b): the simulated force trajectories of λ_2 corresponding to (a). (c): simulated trajectories of x_2 with uncertain $\frac{1}{k_1}, \frac{1}{k_2}$ which standard deviations are 10^{-4}. (d): the simulated force trajectories of λ_2 corresponding to (c). (e): simulated trajectories of x_2 with uncertain $\frac{1}{k_1}, \frac{1}{k_2}$ which standard deviations are $5 * 10^{-4}$. (f): the simulated force trajectories of λ_2 corresponding to (e). We rollout dynamics with the control input u which was computed (6.15), the blue lines show the optimal x_2, λ_2 and the rollouts are shown in grey. 115

6.11 A schematic of a planar pusher-slider system. State of the system is $[x, y, \theta, p_y]^\top$ assuming that the pusher only comes in contact with the left edge as shown in the figure. 118

6.12 Results of SNMPC. Top left: no mpc (open loop), top right: DMPC, bottom left: $\Delta = 0.5$, bottom right: $\Delta = 0.01$. The blue curve shows the reference trajectory of the center of the box and the green lines show the simulated trajectories. The red curves show the bounds. 119

7.1 We consider the problem of reorienting parts for assembly using pivoting manipulation primitive. Such reorientation could possibly be required when the parts being assembled are too big to grasp in the initial pose (such as the gears) or the parts to be inserted during assembly are not in the desired pose (such as the pegs). The figure shows some instances during the implementation of our controller to reorient a gear and a peg. 122

7.2 A schematic showing the free-body diagram of a rigid body during pivoting manipulation when the relative angle between F_W and F_S is zero. Point P is the contact point with a manipulator. The black circle represents the origin of each frame. The object experiences four forces corresponding to two friction forces from external contact points A and B, one control input f_P from the manipulator at point P, and gravity at point C. 125

7.3 A schematic showing the frame definition of a rigid body during pivoting manipulation. F_W, F_S, F_O, and F_B are the world frame, slope frame, object frame, and frame at contact location B, respectively. Gravity is defined in F_W where the gravity is parallel to y-axis of F_W. Pivoting manipulation happens with extrinsic contact A and B defined in F_S. F_O is fixed with CoM of an object. F_B is in parallel to F_S with offset B_x^S along x-axis of F_S. We also show an example of i_x^Σ and i_x^Σ in Table 7.1. In this example, C_x^B and C_y^B are illustrated. 125

7.4 A schematic showing the free-body diagram of a rigid body during pivoting manipulation. We consider the stability margin of finger location due to imperfect control of stiffness controller in a robotic manipulator. 135

7.5 A schematic showing the free-body diagram of a rigid body during pivoting manipulation with patch contact. We approximate patch contact as two point contacts P_1 and P_2 with the same force distribution. We assume that P_1 always lies on the vertex of the object for this simplistic patch contact model. s is the distance between point contact P_1 and P_2 along y-axis of F_O. 137

7.6 Conceptual schematic of our proposed frictional stability and robust trajectory optimization for pivoting. Due to slipping contact, friction forces at points A, B lie on the edge of friction cone. Given the nominal trajectory of state and control inputs, friction forces can account for uncertain physical parameters to satisfy quasi-static equilibrium. We define the range of disturbances that can be compensated by contacts as frictional stability. The above figure shows the case of uncertain mass and CoM location. 138

7.7 This figure illustrates the idea of the proposed contact implicit bilevel optimization, CIBO. Given the trajectory of x, u, f, the stability margin over the trajectory can be computed as shown in lower-level optimization problem. Then, given the computed stability margin over the trajectory ϵ, the upper-level optimization problem maximizes the worst-case stability margin over the trajectory by optimizing the trajectory of x, u, f. Our CIBO simultaneously optimizes the lower-level optimization problem and the upper-level optimization problem. In the right plot, red and blue arrows represent the stability margin along positive and negative directions, respectively. Our CIBO optimizes the stability margin for each direction. 140

7.8 A schematic of pivoting for a non-convex shape object where contact set changes over time. During mode 1, the peg rotates with contact at A and B_2. During mode 2, the peg rotates with contact at A and B_1. γ represents one of the kinematic features of peg, which is used to discuss the result in Sec 7.9. 144

7.9 Trajectory of frictional stability margin. ϵ_A, ϵ_B are bounds of ϵ from (7.10), (7.11). r_A, r_B are bounds of r from (7.13). $\epsilon_+, \epsilon_-, r_+, r_i$ are solutions obtained from CIBO. (a), (b): Trajectory of frictional stability of gear 1 based on uncertain mass obtained from baseline optimization, our CIBO, respectively. (c), (d): Trajectory of frictional stability of gear 1 based on uncertain CoM location obtained from baseline optimization, CIBO, respectively. (e): Snapshots of pivoting motion for gear 1 obtained from CIBO considering uncertain CoM location. 148

7.10 (a), (b): Trajectory of frictional stability margin of peg 1 based on uncertain mass obtained from CIBO, baseline optimization, respectively. Note that here we solve CIBO sequentially for each mode (i.e., hierarchical planning), instead of using the proposed mode-sequence-based optimization. (c): Snapshots of pivoting motion for peg 1, obtained from CIBO considering uncertain mass. 148

7.11 We show the time history of object angle, finger position, and contact forces from a manipulator during pivoting of gear 1. The top row shows the result using CIBO (7.29) considering CoM uncertainty and the bottom one shows the result using (7.25) (i.e., it does not consider robustness criteria in the formulation explicitly.). The top row results and the bottom row results are used in visualizing the stability margin in Fig. 7.9 (d), (c), respectively. 149

7.12 Time history of frictional stability margin considering CoM location with different initial manipulator position $P_y^O(t=0)$. 152

7.13 (a): Time history of stability margin considering CoM location with different mass. The trajectory with the same color means that the same mass is used in the CIBO. The trajectories where $r > 0$ are the trajectories of r_+ and the the trajectories where $r < 0$ are the trajectories of r_-. (b): Time history of f_{nP}^O. (c): Time history of f_{tP}^O. 154

7.14 (a): Time history of stability margin considering CoM location with different friction at P. The trajectory with the same color means that the same mass is used in the CIBO. The trajectories where $r > 0$ are the trajectories of r_+ and the the trajectories where $r < 0$ are the trajectories of r_-. (b): Time history of P_y^O. (c): Time history of f_{nP}^O. 155

7.15 Trajectory of frictional stability margin of (a) gear 1 and (b) gear 3, based on uncertain friction obtained from CIBO (7.30), respectively. 156

7.16 We consider CIBO with uncertain finger contact location. (a): Time history of frictional stability margin. (b) Time history of normal force at the finger. . . . 157

7.17 We show the time history of object angle, finger position, contact forces from a manipulator, and frictional stability margins. The top row shows the result with peg 2 and the bottom one shows the result with peg 3. The pink and blue shade regions represent that the system follows mode 1 and mode 2, respectively. . . . 158

7.18 We consider CIBO with uncertain mass on varying angles of slope. (a): Time history of stability margin, ϵ_+. (b) Time history of stability margin, ϵ_-. The case where the object is on the slope whose angle of slope is 20° is illustrated in Fig. 7.18b. 159

7.19 Trajectory of frictional stability margin of gear 2 based on uncertain CoM obtained from CIBO using point contact model and patch contact model, respectively. The vertical blue line represents the moment when the projection of CoM lies on the contact B. 162

7.20 Snapshots of hardware experiments. We show snapshots of the white peg and gear (instead of overlaid images) for clarity. 163

7.21 The different objects used in hardware evaluation of the proposed method. Please check the hardware experiments results in the video at this link `https://www.youtube.com/watch?v=ojlZDaGytSY`. 164

8.1 (a): cartpole with softwalls. (b): acrobot with soft joints. 175

8.2 Uncertainty propagation for cartpole system. Here only uncertainty arises from stiffness parameters k_1, k_2. 175

8.3 Simulated trajectories for cartpole system using ERM-based controller. $\Delta = 0.2$ and $\Delta_{\text{test}} = 0.083$. Red lines show boundaries specified in chance constraints. . . 176

8.4 Simulated trajectories for cartpole system using our open-loop controller. $\Delta = 0.2$ and $\Delta_{\text{test}} = 0.190$ where Δ is input of optimization and Δ_{test} is the empirically obtained success rate from MC simulation. Red lines show boundaries specified in chance constraints. 176

8.5 Simulated trajectories for cartpole system using our closed-loop controller. Top: $\Delta = 0.6$ and $\Delta_{\text{test}} = 0.510$, bottom: $\Delta = 0.2$ and $\Delta_{\text{test}} = 0.188$, where Δ is input of optimization and Δ_{test} is the empirically obtained success rate from MC simulation. Red lines show boundaries specified in chance constraints. 177

8.6 Simulated trajectories for acrobot using our open- and closed-loop controllers. Top: closed-loop controller with $\Delta = 0.8$ and $\Delta_{\text{test}} = 0.771$, bottom: open-loop controller with $\Delta = 0.4$ and $\Delta_{\text{test}} = 0.366$. Red lines show boundaries specified in chance constraints. The reader should note that open-loop controller solution was infeasible for $\Delta = 0.8$, and thus we show results for $\Delta = 0.4$. 180

9.1 We present tactile tool manipulation where a robot uses an external tool to manipulate an external object. Usage of an external tool results in multiple contact formations which leads to a large number of constraints that need to be satisfied during manipulation. Using the underlying frictional mechanics, we present design of open-loop and closed-loop controllers which can successfully maintain all contact formations during manipulation. We present the design and use of a tactile estimator which makes use of tactile sensing to estimate pose of the system. The tactile estimator is used to perform closed-loop control in an MPC fashion. All hardware experiment videos could be found at https://youtu.be/VsClK04qDhk. 183

9.2 **Mechanics of tool manipulation.** (a): A simplified 3D contact model for tool manipulation highlighting the three main contact interactions during the task. (b): Free-body diagram of a rigid body and a tool during tool manipulation in 2D. (c): Force from a tool to an object has to lie on a cone defined by the shape of the object. 186

9.3 **Tactile estimator.** (a): Given measurements of robot proprioception and tactile sensors, our method estimates the state of the object and the tool. (b): Schematic showing tool manipulation experiencing rotational slipping by θ_S. 190

9.4 **Open-loop tool manipulation.** Our controller could successfully perform tool manipulation with different object-tool-environment pairs. The bottom right picture shows the objects and the tools we use in this chapter. 192

9.5 **Evaluation of the tactile estimator.** We show the time history of error of θ_O for 5 trials (a) with the open-loop controller under no disturbance, (b) with the open-loop controller under disturbance, and (c): with the closed-loop controller under disturbance. The red line shows the mean of and the blue region shows the 95% confidence interval. We added disturbance around $t = 40$ s for (b) and (c) (see the blue box). For (c), we added another disturbance around $t = 150$ s (see the orange box). The contact is lost around $t = 110$ s for (b) (see the green box). Note that for the open-loop controller, the trajectory runs until $t = 160$ s because open-loop controller is pre-defined. 195

9.6 **Evaluation of the closed tactile controller.** We show time history of (a) θ_O and (b) θ_G, with the closed-loop controller under disturbances at $t = 40, 150$ s. The blue line is the reference trajectory computed offline and the red trajectory is the mean of the 5 trajectories computed online. 197

List of Tables

4.1 Notation of variables. C or B indicates the variable is continuous or binary variables, respectively. In Σ column, we indicate the frame of variables. Subscript t indicates time-step. 47

4.2 Comparison of computation time and success rate for walking and climbing problems between benchmark optimization based on P1 using [1], our two-block ADMM, and our multi-block ADMM. For ADMMs, the computation time is calculated as the total computation time until the norm of residual for body and foot positions converge to 0.03 m and the norm of residual for reaction forces converge to within 0.5 N. For computation time, we show the mean time and 99 % confidence interval. 60

4.3 Comparison of the number of slipping between a generated trajectory with patch constraints (Traj A) and without patch constraints (Traj B) over 5 samples for each case. 63

5.1 Varied orientations for collecting datasets of GP 73

5.2 NLP specifications for climbing on non-uniform walls 82

6.1 Comparison of the constraint violation probability specified in the different optimization problems against the observed constraint probabililty obtained from simulation of the "cartpole with softwalls" over 1000 samples. In the table, Δ represents the constraint violation probability for our approach, β for the ERM-based approach in [2], and the Δ_z is for the CCC method in [3]. 111

6.2 Comparison of the constraint violation probability specified in the different optimization problems against the observed constraint probabililty obtained from simulation of the “a sliding box with friction” over 1000 samples. In the table, Δ represents the constraint violation probability for our approach, β for the ERM-based approach in [2], and the Δ_z is for the CCC method in [3]. 111

6.3 Comparison of the constraint violation probability specified in the different optimization problems against the observed constraint probabililty obtained from simulation of the “dual manipulation” over 1000 samples. In the table, Δ represents the constraint violation probability for our approach, β for the ERM-based approach in [2], and the Δ_z is for the CCC method in [3]. 112

6.4 Computation Time of our method. $n_C, n_I, n_{\text{constraints}}$ show the number of continuous variables, the number of integer variables, and the number of total constraints, respectively, for each problem. 116

6.5 Computation Time of other methods. $n_{\text{variables}}$ and $n_{\text{constraints}}$ show the total number of variables and the number of total constraints, respectively, for each problem. 116

6.6 Comparison of obtained Δ and MSE with different Δ from the simulation of "pushing with slipping" over 100 samples. 118

7.1 Notation of variables for analysis of frictional stability margin. In Σ column, we indicate the frame of variables. We use the following indices for defining variables in this table: $j \in \{A, B, C, P\}$ for representing the location of frames, $i \in \{A, B, P\}$ for representing contact location, and $\Sigma \in \{W, S, O, B\}$ for representing a frame. 127

7.2 Parameters of objects. m, l, w represent the mass, length, and the width of the object, respectively. For pegs, the first element in l, w are l_1, w_1 and the second element in l, w are l_2, w_2, respectively, shown in Fig. 7.20. For pegs, since they are made of the same material and they make contact on the same environment, we can assume $\mu_B = \mu_{B_1} = \mu_{B_2}$. 147

7.3 Worst-case stability margin over the control horizon obtained from optimization for gear 1. Note that the stability margin for the solution of the benchmark optimization is analytically calculated. 147

7.4 Obtained worst stability margins over the time horizons from optimization for peg 1. Note that the stability margin for the solution of the benchmark optimization is analytically calculated. 149

7.5 Computed worst-case stability margin considering uncertain CoM location with different P_y^O at $t = 0$ over the control horizon obtained from optimization for gear 1. 153

7.6 Average Solving Time (AST) comparison between benchmark optimization (7.25) and CIBO under mass uncertainty using (7.29) with gear 2. 160

7.7 NLP specification for CIBO under frictional uncertainty using (7.30) with gear 1. 161

7.8 NLP specification for CIBO over mode sequences considering uncertain CoM location using (7.32) with peg 3. 161

7.9 Computed worst-case stability margin considering uncertain CoM location over the control horizon obtained from optimization for gear 2. 162

8.1 Comparison of feasibility for cartpole system among open-, non-contact-aware closed, and contact-aware-closed controllers with different Δ. ○ and × show if optimization finds a feasible solution or not, respectively. 178

8.2 Comparison of safe probability and runtime for cartpole system between important-particle method (top) with $\gamma = 10, \eta = 10$ and naive method (bottom) with $\Delta = 0.6$ for designing the closed-loop controller. T represents runtime for each iteration and n_p is the number of particles. 179

8.3 Comparison of safe probability and runtime for acrobot system between important-particle method (top) with $\gamma = 4, \eta = 4$ and naive method (bottom) with $\Delta = 0.8$. T represents runtime for each iteration and n_p is the number of particles. 181

9.1 **Notation of variables**. Σ column indicates the frame of variables. See Fig. 9.2 and Fig. 9.3 for graphical definition. 185

9.2 **Evaluation of the closed tactile controller with disturbances**. The number of successful pivoting attempts of the box over 5 trials for different disturbances are summarized. 198

9.3 **Evaluation of the closed tactile controller with inaccurate parameters**. The number of successful pivoting attempts of the box over 5 trials for different mass of the object are summarized. The true value of mass of the object is $m_O = 100$ g. 198

Acknowledgments

First and foremost, I would like to extend my sincerest gratitude to my PhD advisor, Prof. Dennis Hong, for his unconditional support and guidance. He has been generous and I am extremely appreciative of the creative freedom he gave me while I worked in the lab. I learned a number of things from him to become a good researcher and engineer. One thing I have to pick from what I learned from him is "break robots" - without understanding the limitations of robots, no innovation can happen. I would like to keep remembering this philosophy for the rest of my robotics life. Thanks to him, I have a role model for life that I can always look up to and reference as I now live the rest of my life.

I also would like to thank my committee members, Prof. Lieven Vandenberghe, Prof. Jacob Rosen, and Prof. Brett Lopez, for their valuable feedback on both research and presentation. I also would like to thank my former committee members, Prof. Jason Speyer and Prof. Paulo Tabuada at UCLA for their feedback on my PhD qualifying exam.

I also would like to express my gratitude to Prof. Kazuya Yoshida at Tohoku University and Prof. Kenji Nagaoka at Kyushu Institute of Technology for their support of my undergraduate research project. I enjoy my research in the Space Robotics Lab at Tohoku University, which encourages me to move forward to PhD study at UCLA. Prof. Yoshida provided me with a creative environment with funding support and Prof. Nagaoka gave me opportunities to discuss my research project. What I learned from them is fundamental and that has been useful even during my PhD study.

I also would like to thank Dr. Devesh Jha, Dr. Arvind Raghunathan, and Dr. Diego Romeres at Mitsubishi Electric Research Laboratories (MERL) for providing me with the opportunity to work as a research intern at MERL. In particular, I would like to say thank Devesh for his unlimited passion and knowledge, and skills for robotic manipulation research. With his support and guidance, I could dramatically enhance my research capabilities. I also would like to express my gratitude to Arvind, for his deep experience and knowledge for optimization. Thanks to him, I have a better understanding of optimization and re-

recognize how interesting optimization research is. Diego has helped me work on stochastic optimization projects and I appreciate his kind support for my stay at MERL. By working with him, I could learn more about stochastic optimization.

I am grateful to Dr. Tao Pang, Dr. Jiuguang Wang, Dr. Max Simchowitz, Dr. Simon LeCleach, Dr. Hongkai Dai, Dr. Andy Park, Farzad Niroui, Hyung Ju Terry Suh, Huaijiang Zhu, and Xinpei Ni for their kind guidance and support for my internship opportunity at Boston Dynamics AI Institute (BDAI). In particular, I would like to extend my gratitude to Pang, for his deep knowledge of contact simulation and engineering skills, especially coding skills and some software knowledge. Thus, I have a better understanding of these skills. JW is a great manager - he has been very generous and has helped my internship projects. Thanks to him, I could experience how R&D goes in the industry. I also would like to really appreciate Max's kind and great support for my internship project. Thanks to his unlimited passion and knowledge in a wide range of robotics research fields, I could enjoy my internship project. I also would like to say thank to Simon for his kind support and his deep experience in contact-rich optimization. His generous efforts make me want to keep working on contact optimization after my PhD.

I would like to thank members of the Robotics and Mechanisms Laboratory (RoMeLa). I have received significant help over the years for discussion, debugging, and having fun. In particular, I would thank members of the climbing robot team. I am very fortunate and grateful to work with Dr. Xuan Lin. When I joined RoMeLa, Xuan was the student who helped me work on my planning project for SiLVIA. He has been generous and always gives me the correct feedback for research ideas, optimization knowledge, coding, engineering skills, writing, and discussion. He has been my role model as a PhD student. For my PhD study, I am so grateful and fortunate to work with Yusuke Tanaka, who has exceptional experiences in both software and hardware. Thanks to his great efforts, I could work on great robots, SiLVIA, GOAT grippers, and SCALER. He helped me a lot in not only research projects in RoMeLa but also activities outside the campus. I also really enjoyed some random discussions with him during my PhD. My work could not be done without his kind and great support.

I also would like to thank Alexander Schperberg for his kind support during my PhD. In particular, Alex and I worked as research interns at MERL at the same time for two summers. That was a great moment as I could have many deep research and non-research discussions with him so frequently. I also want to say thank you to Kyle Gillespie, Hayato Kato, Varit Vichathorn, and Feng Xu for their kind support and help in climbing robot projects. I am so honored to work with them since they are experts in robotics. What's more, they are very passionate about the projects and thus I have a very productive and fun time with them. I also would like to say thank you to Dr. Jeffrey Yu, Dr. Hosik Chae, Dr. Gabriel Fernandez, Dr. Aaron Zhang, Dr. Junjie Shen, Colin Togashi, Nick Liu, Donghun Noh, Harry Nam for taking the time with me to answer my random research and non-research questions.

I would like to thank the members of Space Robotics Laboratory (SRL), Prof. Kentaro Uno, Dr. Warley Francisco Rocha Ribeiro, Dr. Yuto Suebe, Masafumi Endo, Hayato Minote, Kyohei Maruya, Akemi Sato, and everyone in SRL for their deep expertise on research, engineering, presentation skills, and great kindness and support. In particular, I really appreciate Minote-san and Maruya-san for their kind and deep support for my undergraduate research. I was not skilled and I did not know anything in robotics, but they were people who helped me get started on my robotics research journey and my first role models as graduate students. I really would like to say thank you to Prof. Uno. He and I have worked on similar research topics over the years and I appreciate his deep expertise in climbing robotics. Thanks to the discussion with him, I could come up with many different research ideas. Also, he organized our visit to Tohoku University where we had a great time for presentation and discussion of climbing robotics, which was very fun and appreciated. I am very honored to keep collaborating with him over the years. Thanks to him, I have another role model in academia. I would like to say thank you to Warley-san and Endo-san as well since I really had a great time during my undergraduate research and even after I graduated from Tohoku University. We had many research and non-research discussions and I appreciate their kind support.

I received scholarships from the Funai Foundation for Information Technology and the

Ezoe Memorial Recruit Foundation, which were essential to my PhD study. I sincerely appreciate their kind support. I could focus on my research projects without worrying about the funding. The foundation also provided me with many opportunities to interact with other researchers and PhD students in other top schools, which enabled me to expand my network during my PhD study. I also would like to thank Prof. Takeo Kanade at CMU, who also guided me in my journey toward my PhD in the US. Prof. Kanade said to me, "Enjoy stochasticity in research because that is essential to research.". This was quite challenging to me as I had been a very nervous man who ensured all the steps toward the goal. Through my PhD, I believe I have become a researcher who enjoys stochasticity in research much more.

Finally, I would like to express my most sincere gratitude to my parents, Hiroki and Akemi, and to my brother, Koki, for their unconditional love and support during the incredible but also challenging journey of my PhD.

Vita

2018–2024	Ph.D. (expected), Mechanical Engineering, University of California, Los Angeles.
2023	Research Intern, Boston Dynamics AI Institute.
2022	Research Intern, Mitsubishi Electric Research Laboratories.
2021	Research Intern, Mitsubishi Electric Research Laboratories.
2014–2018	B.S., Mechanical and Aerospace Engineering, Tohoku University.

Publications

1. **Y. Shirai**, D. Jha, and A. Raghunathan, "Robust Pivoting Manipulation using Contact Implicit Bilevel Optimization", (under review for IEEE Transactions on Robotics).

2. **Y. Shirai**, D. Jha, A. Raghunathan, and D. Romeres, "Chance-Constrained Optimization for Contact-rich Systems using Mixed Integer Programming", Nonlinear Analysis: Hybrid Systems, 2024.

3. A. Schperberg, **Y. Shirai**, A. Schperberg, X. Lin, Y. Tanaka, and D. Hong, "Adaptive Force Controller for Contact-Rich Robotic Systems Using an Unscented Kalman Filter", in *Proc. 2023 IEEE-RAS Int. Conf. Humanoid Robot.*, pp. 1-8, 2023.

4. **Y. Shirai**, D. Jha, A. Raghunathan, and D. Romeres, "Chance-Constrained Optimization in Contact-rich Systems", in *Proc. 2023 American Cont. Conf.*, pp. 14-21, 2023.

5. **Y. Shirai**, D. Jha, and A. Raghunathan, "Covariance steering for uncertain contact-rich systems", in *Proc. 2023 IEEE Int. Conf. Robot. Auto.*, pp. 7923-7929, 2023.

6. **Y. Shirai**, D. Jha, A. Raghunathan, and D. Hong, "Tactile Tool Manipulation", in *Proc. 2023 IEEE Int. Conf. Robot. Auto.*, pp. 12597-12603, 2023.

7. **Y. Shirai**, X. Lin, A. Schperberg, Y. Tanaka, H. Kato, V. Vichathorn, and D. Hong, "Simultaneous Contact-Rich Grasping and Locomotion via Distributed Optimization Enabling Free-Climbing for Multi-Limbed Robots", in *Proc. 2022 IEEE/RSJ Int. Conf. Intel. Robot. Syst.*, pp. 13563-13570, 2022.

8. Y. Tanaka, **Y. Shirai**, X. Lin, A. Schperberg, H. Kato, A. Swerdlow, N. Kumagai, and D. Hong, "SCALER: A Tough Versatile Quadruped Free-Climber Robot", in *Proc. 2022 IEEE/RSJ Int. Conf. Intel. Robot. Syst.*, pp. 5632-5639, 2022.

9. **Y. Shirai**, D. Jha, A. Raghunathan, and D. Romeres, "Robust Pivoting: Exploiting Frictional Stability Using Bilevel Optimization", in *Proc. 2022 IEEE Int. Conf. Robot. Auto.*, pp. 14-21, 2022.

10. Y. Tanaka, **Y. Shirai**, Z. Lacey, X. Lin, J. Liu, and D. Hong, "An Under-Actuated Whippletree Mechanism Gripper based on Multi-Objective Design Optimization with Auto-Tuned Weights", in *Proc. 2021 IEEE/RSJ Int. Conf. Intel. Robot. Syst.*, pp. 6139-6146, 2021.

11. **Y. Shirai**, X. Lin, A. Mehta, and D. Hong, "LTO: Lazy Trajectory Optimization with Graph-Search Planning for High DOF Robots in Cluttered Environments", in *Proc. 2021 IEEE Int. Conf. Robot. Auto.*, pp. 7533-7539, 2021.

12. **Y. Shirai**, X. Lin, Y. Tanaka, A. Mehta, and D. Hong, "Risk-Aware Motion Planning for a Limbed Robot with Stochastic Gripping Forces Using Nonlinear Programming", *IEEE Robot. Auto. Lett.*, vol. 5, no. 4, pp. 4994-5001, 2020.

CHAPTER 1

Introduction

1.1 Motivation

Robotic manipulation and locomotion are a critical aspect of automation that emulates the human ability to handle and interact with environments in the physical world. Robotic manipulation and locomotion have shown impressive progress for the past few decades in a variety of fields such as manufacturing, logistics, healthcare, and inspection [4, 5, 6]. In particular, for robotic manipulation, it has shown successful capabilities in pick-and-place tasks where a robot grasps an object often with a two-fingered parallel jaw gripper in highly structured environments such as factories. Robotic locomotion also has shown remarkable capabilities in walking and running tasks where a robot can walk and run in various terrains even including slippy deformable terrains. These capabilities are accomplished thanks to the rapid advancement of robotic planning, control, estimation, and sophisticated hardware.

However, despite significant progress in robotic manipulation and locomotion, the current robotic manipulation and locomotion cannot achieve as dexterous and robust tasks as we humans can do. For example, humans can cut an onion, which is much more complex than the traditional pick-and-place task since it involves multiple contacts between the robot and the knife and the knife and the onion. Similarly, humans can do rock climbing by understanding how to grasp the climbing holds and when to move the legs. There are still many challenges in robotic manipulation and locomotion to achieve these capabilities. The reason why robotic manipulation and locomotion have many unique challenges is **contact**. A robotic manipulator uses contact between the robot and an object to interact with the object, and a legged robot uses contact between the robot and environments to traverse in environments.

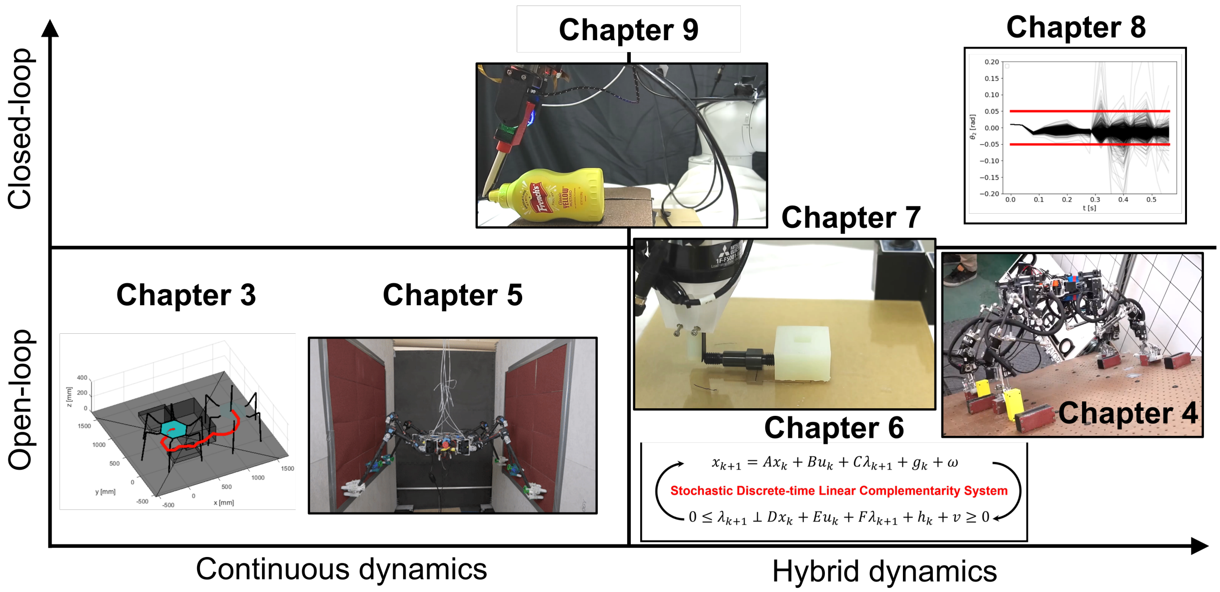

Chapter 9
Chapter 8
Chapter 7
Chapter 3
Chapter 5
Chapter 6
Chapter 4
Closed-loop
Open-loop
Continuous dynamics
Hybrid dynamics
$x_{k+1} = Ax_k + Bu_k + C\lambda_{k+1} + g_k + \omega$
Stochastic Discrete-time Linear Complementarity System
$0 \leq \lambda_{k+1} \perp Dx_k + Eu_k + F\lambda_{k+1} + h_k + v \geq 0$

Figure 1.1: We categorize our contributions using two axes. The horizontal axis shows if the proposed framework considers the change of contact modes (e.g., braking-making contact. See Fig. 2.1) in the proposed framework. The vertical axis shows if the framework uses sensor measurements (i.e., open-loop (planner) or closed-loop control (controller). See Fig. 1.2).

Therefore, understanding contact is the key to introducing additional dexterity and robustness for achieving human-like dexterous and robust manipulation and locomotion skills. In this dissertation, we focus on discussing how contact can be useful by understanding the underlying contact mechanics. In particular, we study **planning through contact**, **robust planning through contact**, and **closed-loop control through contact**.

1.2 Contributions

Toward human-like dexterous and robust manipulation and locomotion, we have the following contributions as shown in Fig. 1.1. Note that in this dissertation we define a planner as a feedforward controller which is not updated during operation. In contrast, we define a controller as a feedback controller which commands control inputs accordingly based on the sensor measurements. Fig. 1.2 illustrates the pipeline we present in this dissertation.

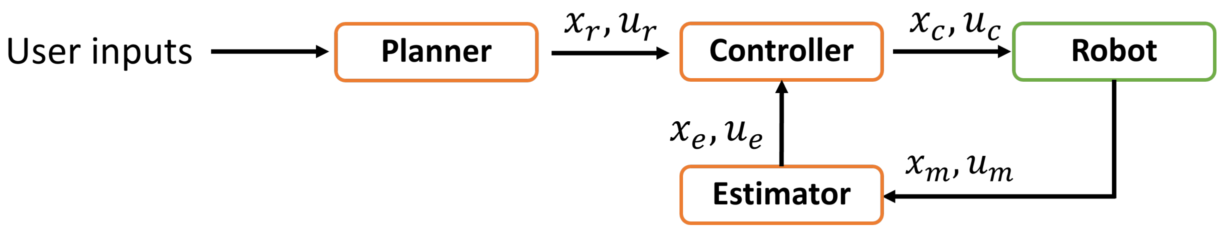


Figure 1.2: We study contact-rich planners and controllers for different contact-rich systems from multi-limbed robots for climbing to tool manipulation for pivoting. x and u represent states and control inputs, respectively. The subscripts r, c, m, e represent reference, command, measurements, and estimates of x and u, respectively. Planners (e.g., trajectory optimization algorithms in Chapter 3, Chapter 4, Chapter 5, Chapter 6, Chapter 7)) do not use sensor measurements or estimates of the states. On the other hand, controllers (e.g., covariance steering in Chapter 8 and MPC in Chapter 9) use sensor measurements or estimates of the states.

1.2.1 Contribution 1: Planning through Contact

1.2.1.1 Contribution 1.1: Lazy Trajectory Optimization with Graph-Search Planning in Cluttered Environments

In this dissertation, we first show our initial work considering trajectory optimization for multi-limbed robots without considering the change of contact modes although we still consider complex collision avoidance constraints. We present Lazy Trajectory Optimization (LTO) that unifies local short-horizon TO and global Graph-Search Planning (GSP) to generate a long-horizon global optimal trajectory. We demonstrate LTO's performance on motion planning problems for a 2 DOF free-flying robot and a 21 DOF legged robot, showing that LTO outperforms existing algorithms in terms of its runtime and reliability.

1.2.1.2 Contribution 1.2: Planning through Contact using Decomposition-based Optimization for Multi-Limbed Robots

While we do not consider the change of contact modes in Section 1.2.1.1, we here consider the change of contact modes. We consider motion planning for contact-rich robotic systems, where the robot makes *many* contacts. We present an efficient motion planning framework for simultaneously solving kinematics, dynamics, friction, and contact problems. To accelerate the planning process, we propose decomposition-based optimization frameworks based on Alternating Direction Methods of Multipliers (ADMM) to solve the original large-scale Mixed-Integer NonLinear Programming (MINLP). The resulting frameworks use Mixed-Integer Quadratic Programming (MIQP) to solve contact and NonLinear Programming (NLP) to solve nonlinear dynamics, which are more computationally tractable and less sensitive to parameters. Also, we explicitly enforce patch contact constraints from limit surfaces with micro-spine grippers. We demonstrate our proposed framework in the hardware experiments of robotic climbing, showing that the multi-limbed robot is able to realize various motions including free-climbing at a slope angle of 45° with a much shorter planning time.

1.2.2 Contribution 2: Robust Planning through Contact

1.2.2.1 Contribution 2.1: Risk-Aware Motion Planning for Multi-Limbed Robots

We present a motion planning algorithm with probabilistic guarantees for multi-limbed robots with stochastic gripping forces. Our proposed planner enables the robot to simultaneously plan its pose and contact force trajectories while considering the risk associated with the gripping forces. Our planner is formulated as a nonlinear programming problem with chance constraints, which allows the robot to generate a variety of motions based on different risk bounds. To model the gripping forces as random variables, we employ Gaussian Process regression. We validate our proposed motion planning algorithm on an 11.5 kg six-limbed robot for two-wall climbing. Our results show that our proposed planner generates various trajectories (e.g., avoiding low friction terrain under the low risk bound, choosing an unstable but faster gait under the high risk bound) by changing the probability of risk based on various specifications.

1.2.2.2 Contribution 2.2: Chance-Constrained Optimization for Contact-Rich Systems

In Section 1.2.2.1, stochastic contact dynamics is not discussed, which leads to the failure of the mission. We present a chance-constrained formulation for robust trajectory optimization during general contact-rich systems, not limited to quasi-static locomotion as presented in Section 1.2.2.1. In particular, we present chance-constrained optimization of Stochastic Discrete-time Linear Complementarity Systems (SDLCS). The optimization problem is formulated as a Mixed-Integer Quadratic Program with Chance Constraints (MIQPCC). In our formulation, we explicitly consider joint chance constraints for complementarity variables and states to capture the stochastic evolution of dynamics. Additionally, we demonstrate the use of our proposed approach for designing a Stochastic Model Predictive Controller (SMPC) with complementarity constraints for a planar pushing system.

1.2.2.3 Contribution 2.3: Robust Pivoting Manipulation using Contact Implicit Bilevel Optimization

We study robust optimization for planning of pivoting manipulation in the presence of uncertainties. We present insights about how friction can be exploited to compensate for inaccuracies in the estimates of the physical properties during manipulation. Under certain assumptions, we derive analytical expressions for stability margin provided by friction during pivoting manipulation. This margin is then used in a Contact Implicit Bilevel Optimization (CIBO) framework to optimize a trajectory that maximizes this stability margin to provide robustness against uncertainty in several physical parameters of the object. We present analysis of the stability margin with respect to several parameters involved in the underlying bilevel optimization problem. We demonstrate our proposed method using a 6 DoF manipulator for manipulating several different objects.

1.2.3 Contribution 3: Closed-Loop Control through Contact

1.2.3.1 Contribution 3.1: Covariance Steering for Uncertain Contact-Rich Systems

Planning and control for uncertain contact systems is challenging as it is not clear how to propagate uncertainty for planning. Contact-rich tasks can be modeled efficiently using complementarity constraints among other techniques. In this paper, we present a stochastic optimization technique with chance constraints for systems with stochastic complementarity constraints. We use a particle filter-based approach to propagate moments for stochastic complementarity system. To circumvent the issues of open-loop chance constrained planning, we propose a contact-aware controller for covariance steering of the complementarity system. Our optimization problem is formulated as Non-Linear Programming (NLP) using bilevel optimization. We present an important-particle algorithm for numerical efficiency for the underlying control problem. We verify that our contact-aware closed-loop controller is able to steer the covariance of the states under stochastic contact-rich tasks.

1.2.3.2 Contribution 3.2: Closed-Loop Control for Dexterous Tool Manipulation using Tactile Feedback

We present closed-loop control of a complex manipulation task where a robot uses a tool to interact with objects. Manipulation using a tool leads to complex kinematics and contact constraints that need to be satisfied for generating feasible manipulation trajectories. We first present an open-loop controller design using NLP that satisfies these constraints. In order to design a closed-loop controller, we present a pose estimator of objects and tools using tactile sensors. Using our tactile estimator, we design a closed-loop controller based on Model Predictive Control (MPC). The proposed algorithm is verified using a 6 DoF manipulator on tasks using a variety of objects and tools. We verify that our closed-loop controller can successfully perform tool manipulation under several unexpected contacts.

1.3 Outline of Dissertation

The structure of chapters are arranged as follows:

- Chapter 2 summarizes closely related works in this dissertation.
- Chapter 3 presents Contribution 1.1.
- Chapter 4 presents Contribution 1.2.
- Chapter 5 presents Contribution 2.1.
- Chapter 6 presents Contribution 2.2.
- Chapter 7 presents Contribution 2.3.
- Chapter 8 presents Contribution 3.1.
- Chapter 9 presents Contribution 3.2.
- Chapter 10 summarizes the works presented in this dissertation, discusses the limitation and the potential future works, and concludes this dissertation by describing our final thoughts toward robust manipulation and locomotion in the real world.

CHAPTER 2

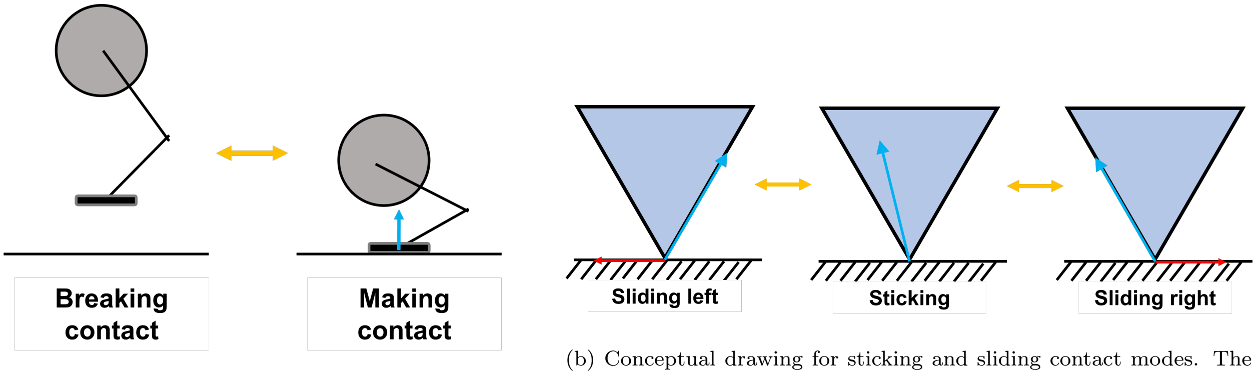
Research Challenges and Related Works

In this chapter, we present the technical challenges we try to solve in this dissertation. In particular, we show three challenges in planning and control for robotic manipulation and locomotion. We then describe related works for each research challenge.

2.1 Optimization-based Planning through Contact

Planning through contact (aka contact-rich planning, contact-aware planning, or open-loop control through contact) computes the offline control input trajectory considering the change of contact such as making and breaking contact relationships and slipping and sticking contact relationships [7], as illustrated in Fig. 2.1. Contact-rich planner enables the robot to design various non-intuitive trajectories with the changes in contact modes.

Planning through contact is already quite challenging. First, the computational complexity can increase exponentially easily. Given N contacts, we have 2^N contact modes for breaking-making contact and 3^N contact modes for slipping-sticking contact. In planning through contact, complementarity constraints [8] or mixed-integer constraints [9] are often used to model these hybrid contact dynamics in trajectory optimization, which leads to non-convex NonLinear Programming (NLP) or Mixed-Integer Programming (MIP). As N increases, the computational complexity of NLP and MIP also increases, and thus eventually the computational complexity becomes intractable. Second, the underlying dynamics for multi-contact systems are nonlinear and non-smooth dynamics, which makes the design of an optimization problem extremely difficult because of non-convexity of the optimization problem. In particular, the non-smooth dynamics make the gradient-based optimization



(a) Conceptual drawing for making and breaking contact modes. The blue arrow shows contact force.

(b) Conceptual drawing for sticking and sliding contact modes. The blue arrows show contact forces and the red arrows show the slip direction. The blue triangles show friction cones.

Figure 2.1: Conceptual drawing of different contact modes. In (a), the robot makes non-zero contact forces when making contact. Otherwise, it makes zero contact forces. In (b), the contact does not slip when the contact force is not on the edge of the friction cones. Otherwise, it slips depending on the direction of the contact force.

solver (e.g., IPOPT [10], SNOPT [11]) face challenges since gradients in one of the contact modes can be almost zero. Although IPOPT considers the relaxation of complementarity constraints and considers more strict complementarity constraints over the iteration of the optimization, the numerical stability decreases with complementarity constraints.

To deal with these challenges, there have been many works in Contact-Implicit Trajectory Optimization (CITO) [12, 9, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23], where the optimizer designs the trajectory of control inputs, states of the system, and contact sequences simultaneously, without any user-specified contact sequences. Some of those works use complementarity constraints and other works use integer constraints to model contact. For both approaches, however, the computational complexity increases and it can be challenging to find feasible solutions as the number of discrete modes increases. Thus, many works solve the approximated Mixed-Integer NonLinear Programming (MINLP). In [14, 24], the authors use NLP with phase-based formulations. One drawback is that the order of phases cannot be changed, which is undesirable for motion planning for limbed robots since the number of phases limbed robot planners consider is large and it can lead to infeasible solutions. The authors in [25, 26] use continuous formulation in NLP to represent discrete terrain. In this dissertation, we consider fully discrete environments for free-climbing tasks so we cannot use these techniques. In [27, 28], the authors decouple the MINLP problem as *sequential* sub-problems (i.e., hierarchical planning). However, such a formulation cannot guarantee that the entire planning process is feasible since it does not in general consider all coupling constraints among sub-problems.

Our work is inspired by distributed optimization such as ADMM [29], which has gathered attention for large-scale optimization problems [30, 31, 32, 33]. In [30, 31], the authors introduce an ADMM-based framework to reason centroidal and whole-body dynamics. This work does not consider whole-body dynamics but considers contact dynamics from grippers and discrete constraints. ADMM is also employed in Model Predictive Control (MPC) for linear complementarity problem [32]. The work in [33] proposed an ADMM-based framework for CITO. We instead consider nonlinear centroidal dynamics and propose a specific splitting scheme for motion planning of limbed robots.

2.2 Optimization-based Robust Planning through Contact

Robust planning through contact computes the offline control input trajectory considering the change of contact under uncertainty. In reality, uncertainty always exists such as process noises, observation noises, and noises in physical parameters such as friction constants. Considering these uncertainties is very important so that the robot is able to complete its desired task under uncertainty. In this section, we first describe the comparison between robust and stochastic optimization. Then, we show the unique challenge of stochastic optimization for contact-rich systems. Finally, we describe robust optimization techniques for contact-rich systems.

2.2.1 Robust Optimization and Stochastic Optimization

In order to design a robust open-loop controller, robust planning problems using optimization can be categorized into two approaches, stochastic optimization and robust optimization

[34, 35, 36, 37]. In robust optimization, the planner designs trajectories that guarantee the feasibility of the motion given the uncertainty bounds. On the other hand, stochastic optimization designs trajectories that guarantee the feasibility of the motion given the Probability Density Function (PDF): it prevents the probability of violating state constraints (violation probability) from being higher than a pre-specified probability.

As for risky tasks (e.g., walking on irregular terrain, climbing on slippy terrain), the stochastic optimization approach has advantages over the robust optimization approach. Because the robust approach can be very conservative, the planner would be likely to find infeasible trajectories. In contrast, the risk-bounded method is more aggressive than the robust approach, leading to a higher probability of finding feasible trajectories through risk-taking. What is more, the violation probability provides a tuning knob to generate various motions in diverse environments that suit the need of the task, which is not taken into account in the current deterministic planning algorithm discussed in Section 2.1.

The key challenge here is it is not clear how to formulate stochastic optimization for multi-contact systems. Furthermore, it is not trivial to obtain the distribution of stochastic parameters. In this dissertation, we present our proposed algorithm which can achieve probabilistic guarantees for motion planning problem of multi-limbed robots for climbing under uncertain friction parameters with some assumptions.

2.2.2 Stochastic Optimization for Contact-Rich Systems

Although there are many works for planning under uncertainty for continuous dynamical systems such as autonomous driving and UAV [35, 38, 39, 40], planning with uncertain contact-rich systems is relatively unexplored. This is primarily because it is not clear how to propagate uncertainty through the complementarity system for planning. Very recently, there has been some work done in this area and we describe them next.

Recent work on robust trajectory optimization in contact-rich systems can be found in [2, 3, 41]. In [2], the authors have utilized the formulation of expected residual minimization (ERM) [42] for robust TO. ERM, first introduced in [42] for Stochastic Linear

Complementarity Problem (SLCP), aims at minimizing the expected error in satisfying the SLCP. In [2], authors use ERM as an additional penalty term in their TO problem. However, such a formulation does not consider the stochastic state evolution of the system during optimization. A chance-constrained formulation for the stochastic nonlinear complementarity system is presented in [3]. This method augments the ERM-augmented objective in [2] with additional chance constraints on satisfying the complementarity constraints. The formulation ignores the stochastic evolution of system state during optimization, and thus borrows the limitations of [2]. Furthermore, this formulation is incapable of enforcing a constraint violation probability smaller than 0.5 for any degree of uncertainty. Consequently, this method is very fragile for trajectories with horizon lengths longer than one ($T > 1$, T is the time horizon), as the chance of violating the constraints for such trajectories is $0.5T \geq 1$ [43]. In this dissertation, our formulation addresses these weaknesses under certain simplifying assumptions for SDLCS. More recently, there has been another work that makes use of particles to perform uncertainty propagation in SDLCS [44]. However, the resulting optimization could become computationally challenging. In contrast, our proposed method formulates a computationally efficient method at the cost of some simplifying assumptions.

Another line of work that is relevant to our proposed work is related to Chance-Constrained Optimization (CCO). This has been extensively studied in robotics as well as in the optimization literature [35, 38, 45]. In [35], authors have proposed stochastic optimization formulation for open-loop collision avoidance problems using chance constraints under Gaussian noise. The authors in [45] use statistical moments of the distribution to handle non-Gaussian chance constraints. An important point to note here is that in all CC formulation for dynamic optimization, one needs to consider the Cumulative Density Function (CDF) function for the joint probability distribution of all variables. However, such distribution is extremely challenging to compute. Thus, in general, the joint chance constraint is decomposed into individual chance constraints using Boole's inequality (see [35, 45]), which results in very conservative approximation of the individual constraints. We also utilize Boole's inequality to convert the original computationally intractable joint chance constraints into conservative but tractable independent chance constraints.

2.2.3 Robust Optimization for Contact-Rich Systems

In this section, we present research challenges and some related works for robust optimization for contact-rich systems, in particular dexterous manipulation tasks. Robust optimization for contact-rich systems often considers stability margin and quasi-static stability margin with multiple point contacts has been widely used in legged locomotion [46, 47, 48, 49]. These works consider the problem of mechanical stability of the legged robot under multiple contacts by considering the stability polygon defined by the frictional contacts. Then, they consider min-max trajectory optimization where the optimizer designs the optimal control sequence while improving the worst-case stability margin along locomotion. The planning framework for optimizing contact wrench cone margin during locomotion is able to achieve robust locomotion results [50, 49, 51]. Similar to the concept of these works, we present the idea of frictional stability which defines the extent to which multiple points of contact can compensate for unknown forces and moments in the presence of uncertainty in the mass, Center of Mass (CoM) location, contact location, and frictional parameters. This idea exploits contact forces to ensure the stability of the object during the two-point pivoting manipulation.

There are many manipulation planning works showing remarkable results. Here, we show some of them and highlight the differences between their works and our work presented in this dissertation. Our work is related to manipulation by shared grasping [52] which discusses the mechanics of shared grasping and shows impressive demonstrations. In contrast to the work presented in [52], we present a robust contact-implicit bilevel optimization (CIBO) framework that can be used to find feasible solutions in the presence of uncertainty during the pivoting manipulation and avoids consideration of different modes during planning. In [53], authors consider stabilization of a table-top manipulation task during online control. They consider a decomposition of the control task in object state control and contact state control. The contact state was detected using vision-based tactile sensors [54, 55]. As the task mostly required sticking contact for stability, the tactile feedback was designed to make corrections to push the system away from the boundary of the friction cone at the different contact locations. However, the authors did not consider the problem of designing trajectories which can provide robustness to uncertainty. Furthermore, the authors only considered controlled sticking in [53] which is, in general, easier than controlled slipping. Similarly, in [56], authors design and validate their sliding controller for in-hand tool pivoting. In [57], the authors extend their sliding controller in [56] such that the sliding controller is able to achieve adaptive control for friction coefficients using visual and force measurements, showing impressive demonstrations. Also, authors in [58] consider pivoting manipulation with a parallel gripper without relying on fast and precise robotic systems. In contrast to their work in [56, 57, 58], we present the pivoting manipulation with extrinsic contacts, which introduces additional complexity of the manipulation, and other uncertain parameters such as mass, CoM location, and robot contact location. The work in [59] discusses dexterous in-hand manipulation including extrinsic contact. However, the work in [59] does not consider uncertainty in physical parameters. Other previous works that study stable pivoting also consider sticking contact during pivoting using multiple points of contact [60]. The problem in [60] is inherently stable as the object is always in stable grasp. Furthermore, the authors do not consider any uncertainty during planning. Similarly, authors in [61] present a MIP formulation to generate contact trajectory given a desired reference trajectory for the object for several manipulation primitives. In contrast, this work proposes a bilevel optimization technique which maximizes the minimum margin from instability that the object experiences during an entire trajectory.

2.3 Closed-Loop Control through Contact

Although robust planning is useful for robots to complete their desired tasks, the robust planner might not be useful and the robots might not be able to complete their tasks if uncertainty is too large. Thus, closed-loop control is indispensable. However, obviously, designing closed-loop control for contact-rich systems is much more challenging than planning through contact and there are many challenges. First, it is not clear how to design a control policy for contact-rich systems because Lyapunov control theory cannot be used for contact-rich systems. Another challenge is the design of a contact estimator. In order to run closed-loop control, it is necessary to estimate the contact states, which can be quite challenging due to the partial observability of the system. In this section, we describe some of the key research challenges and related works for achieving closed-loop control for contact-rich systems.

2.3.1 Covariance Steering: Control Policy with Probabilistic Guarantees for Contact-Rich Systems

Recently, contact-aware feedback controllers for contact-rich systems have been proposed [19] for linear complementarity systems. The authors designs a piecewise affine linear controller using state and contact force feedback using an optimization problem with a bilinear matrix inequality. However, it cannot be extended to consider stochastic complementarity constraints to provide stochastic guarantees. Thus, once the model has uncertainty, the controller might not work. In this dissertation, we present a stochastic linear feedback controller for stochastic contact-rich systems. Thus, our controller is able to achieve robust control under uncertain physical parameters such as friction constants.

Using stochastic complementarity constraints for planning robust manipulation is not so well understood in the literature. Some of the recent work can be found in [2, 26]. However, the problem with these approaches is that the uncertainty needs to be very small otherwise the optimization might be infeasible. Consequently, these approaches could fail to provide robust plans for uncertain contact systems. Furthermore, uncertainty propagation for stochastic complementarity systems is not properly modeled in these approaches. One of the reasons is the implicit relationship between contact and state variables in complementarity constraints. As a consequence, most of the known approaches (e.g., extended Kalman filter [62], unscented Kalman filter [63], moment-based [64, 40]) for uncertainty propagation can not be used for stochastic complementarity systems.

Since open-loop CCO would lead to quite conservative solutions to satisfy chance constraints, covariance steering methods have gained attention to deal with long-horizon planning for uncertain systems [65, 38]. Covariance steering methods are able to design feedforward and feedback gains simultaneously to satisfy chance constraints. However, these cannot be directly applied to contact-rich systems since they assume (in general) linear dynamics with Gaussian additive noises.

2.3.2 Contact Estimation and Control with Visuotactile Sensors for Reactive Manipulation

Our work is inspired by seminal work on manipulation by shared grasping [52] which discusses mechanics of shared grasping and shows impressive demonstrations. The task that we present in this paper is a complex version of shared grasping where the robot uses a tool instead of a rigid end-effector to manipulate objects. This variation leads to additional contact formations. These additional constraints make the problem more complicated to plan, control, and estimate compared to those works.

Model-based planning for tool manipulation was earlier presented in [66]. Learning-based algorithm of grasping for tool manipulation is presented in [67]. In our work, we consider a closed-loop controller and estimator in addition to planning for tool manipulation to robustify the system.

Our work is also closely related to the remarkable previous work on tactile estimation and reactive manipulation presented in [68, 69, 70, 71, 53, 72, 73, 74]. For estimators, [68] show a pose estimator for tools, and [69] present tactile localization. Learning-based estimator for tool manipulation using vision is presented in [70]. In this work, in addition to a tool through tactile sensors, we try to estimate and control a pose of an object, which introduces additional extrinsic contact. For reactive manipulation, our work is closely related to the seminal work presented in [53] where slip detection is used to recompute a new controller that can stabilize the manipulation task. [72] shows the impressive closed-loop controller by simultaneous design of controller and estimator. However, the task in [53] is inherently stable as the object is always grasped by the robot. Also, the tactile sensors can directly estimate the pose of the object, which cannot be done for tool manipulation because tactile sensors are not attached between the object and the end-effector. Compared to [72], which focuses on regulation of an object using force / torque sensors, we focus on tracking of tool manipulation using tactile sensors. Furthermore, the current paper considers multiple contact formations which leads to more complex constraints.

CHAPTER 3

Lazy Trajectory Optimization

In this chapter, we present our framework that unifies local short-horizon Trajectory Optimization (TO) and global Graph-Search Planning (GSP) to generate a long-horizon global optimal trajectory. We first motivate why the basic TO cannot work for complex motion planning problems. Then, we show some closely related works in this chapter. Then, we present our proposed algorithm with some proof of the computational complexity and sub-optimality accounting for TO and GSP. Finally, we demonstrate our algorithm for various robotic systems. Although we do not consider the change of contact modes in this chapter, this chapter shows that planner just considering nonlinear kinematics, dynamics, and collision avoidance constraints is already complex and indicates that TO can be more complicated with the contact constraints which enables the robot to change the contact modes.

This chapter has been partially adapted from one conference paper:

- **Y. Shirai**, X. Lin, A. Mehta, and D. Hong, "LTO: Lazy Trajectory Optimization with Graph-Search Planning for High DOF Robots in Cluttered Environments", in *Proc. 2021 IEEE Int. Conf. Robot. Auto.*, pp. 7533-7539, 2021.

3.1 Overview

Trajectory Optimization (TO), such as the ones based on Mixed-Integer Convex Programming (MICP), solves a motion planning problem to generate an optimal trajectory while satisfying constraints. Since TO can be formulated as long as users acquire constraints and objective functions, it is widely used in various robotic systems [75], [9]. In particular, one of the unique advantages TO has compared with other planning algorithms, such as Sampling-Based Planning (SBP) (e.g., Rapidly-exploring Random Tree (RRT), Probabilistic RoadMap (PRM)) and Graph-Search Planning (GSP) (e.g., A*), is to easily formulate a wide variety of constraints, including equality constraints. Conversely, GSP and SBP take considerable time in a narrow passage because it is difficult to place a sufficient number of grids or samples to represent the states present [76]. Reinforcement learning also has difficulty in satisfying hard constraints in the continuous domain [77].

However, TO has two main drawbacks: expensive computational complexity with a long horizon and convergence to local optima [26], [78]. Long-horizon TO is indispensable for generating feasible global trajectories in cluttered environments, but the computation time grows exponentially as the number of horizons increases. Model Predictive Control (MPC), TO in receding-horizon fashion, can spend less planning time than the long-horizon TO. The limitation of MPC is that since it considers relatively short-horizon TO, it has a greater probability of getting stuck at local optima.

To this end, we address Lazy Trajectory Optimization (LTO) unifying the local short-horizon TO and the global long-horizon GSP. LTO effectively reasons the same constraints as the original large-horizon but with the improved time complexity. We also propose a cost function that considers the computation time of TO to balance the optimality of the trajectory and the planning time. In particular, we focus on the difficulty of the edge evaluation instead of the number of edge evaluations. Next, based on Lazy Weighted A* (LWA*) [79], we improve LWA* by making the vertex generation "lazy". In this work, "lazy" means that LTO runs TO only when it intends to evaluate the configuration and trajectories. In other words, LTO does not run TO to generate all configurations and edges in the graph. Because LTO solves many similar TOs, it employs a warm-start to solve TO, resulting in less planning time. By employing MICP as a short-horizon TO, we are able to analyze the computational complexity of LTO in addition to the bounded solution cost. Note that other TOs can also be incorporated into this framework. Note that our formulation is not restricted to MICP. Other TOs can also be incorporated into this framework.

The contributions can be summarized as follows:

1. We propose LTO, a framework incorporating GSP as a high-layer planner and TO as a low-layer planner, that efficiently generates long-horizon global trajectories.
2. We present the cost function balancing the planning time of TO and the optimality of the trajectory.
3. We present proofs of the complexity, efficiency, completeness, and optimality of LTO.
4. We demonstrate LTO's efficiency on motion planning problems of a 2 DOF free-flying robot and a 21 DOF legged robot.

3.2 Related Works

TO finds a trajectory from dynamic systems that satisfies constraints while minimizing a cost function. For instance, a long-horizon TO based on nonlinear programming is used to generate a globally feasible trajectory [26] while a short-horizon TO based on mixed-integer linear programming is used to generate a locally optimal trajectory [80]. One key characteristic of TO is that it can directly generate kinodynamic trajectories, which is difficult for GSP and SBP.

We focus on using MICP as a short-horizon TO for the following reasons. First, MICP can deal with nonlinear constraints with approximations. For instance, Valenzuela used piecewise McCormick envelopes to deal with bilinear terms [75]. In addition, MICP can consider constraints involving discrete decision variables. However, such formulation can result in extended solving time if the problem has many discrete decision variables and/or the planner computes the long-horizon problem. We show that LTO can find the global resolution-optimal trajectory with the decreased planning time by restricting the MICP to the short-horizon TO.

Another reason is that the worst-case solving time for MICP is bounded theoretically. MICPs are generally solved by Branch and Bound (B&B) [81], [82], which iteratively solves a convex program in a binary search tree while tightening bounds on the best possible solutions.

Graph-search planning algorithms, such as A* [83], first construct a graph where each vertex represents a configuration of a robot. Edges in the graph are connected between vertices if the edge is collision-free. Although A* works well in low-dimensional space, it takes quite an amount of time in high-dimensional space (the curse of dimensionality).

Several GSPs have been studied to overcome the expensive computation time in high-dimensional space. LWA* [79] evaluates edges only when the planner uses them. We improve LWA* to make vertex generations lazy. Typically, GSPs use a pre-computed roadmap, but it takes an extended amount of time to pre-compute the roadmap for planning problems with dynamics. Inspired by LWA*, we notice that it is not necessary to pre-compute vertices in the roadmap, and the planner evaluates the feasible vertex if the planner intends to expand it. In other words, at the start of planning, LTO does not need to know a priori knowledge about the roadmap. LTO updates the roadmap by running TO as time passes.

Optimization algorithms can find the optimal solution with decreased computation time by providing good initial guesses for the problem, which technique is known as a warm-start. Several works have been proposed to have good warm-starts [84], [85]. For example, SBP and kinematics-aware trajectory are used as an initial guess [84], [86]. However, because they do not consider dynamics constrained-trajectories in their initial guesses, they may not fully achieve a good warm-start for motion planning with dynamics. Bergman uses dynamics-aware GSP based on motion primitives to generate initial guesses [85]. In this work, we perform a similar approach. We can regard a roadmap as a set of motion primitives. Compared with [85], we also use the dynamics trajectory (e.g., force variable) for the warm-start. Additionally, as our framework collects similar trajectories over time, this information can enhance the quality of the warm-start for the current trajectory generation.

3.3 Problem Formulation

This section explains our notation, graph structure, MICP formulation, and warm-start strategy.

3.3.1 Notation

In this chapter, LTO solve TO with the help of GSP as shown in Fig. 3.1. Let $\mathcal{G}$ be a graph. $\mathcal{G} = (V, E)$ consists vertices and edges, where $V = (v_1, v_2, \ldots)$, $E = (e(v_i \leftrightarrow v_j)\forall i, \forall j)$. Each vertex represents the state of a robot, and each edge represents the trajectory of the robot between the vertices. In addition, we make voxels in the continuous domain, such that each vertex is in each voxel as shown in Fig. 3.2. At the start of planning, we first connect every two vertices if their ℓ_∞ norm is less than or equal to r. Let K be the number of intervals along each axis. Let i be the number of voxels along each axis to produce a hypercube region where LTO solves the edge (see Fig. 3.2).

We explain the benefits of using this graph structure in Section 3.3.2, how we solve TO within the voxels in Section 3.3.3, and how much we can have dense voxels in Corollary 1.

Since our target is motion planning with dynamics in cluttered environments, it is difficult for planners to have a feasible graph prior to conducting GSP. Hence, at the start of planning, LTO does not have any prior knowledge about $\mathcal{G}$, which means LTO does not know the true configuration associated with the vertex and the true trajectory associated with the edge. We use 0-step TO to solve V and N-steps TO to solve E, and update them in $\mathcal{G}$ while running GSP.

3.3.2 Graph Structure

We solve TO within associated voxels and only use the constraints within voxels as shown in Fig. 3.2. It means while we keep the same constraints of kinematics and dynamics and add additional constraints imposing on start and goal states, we remove several domain-specific constraints, such as obstacle-avoidance constraints, if they are outside the voxels. Because

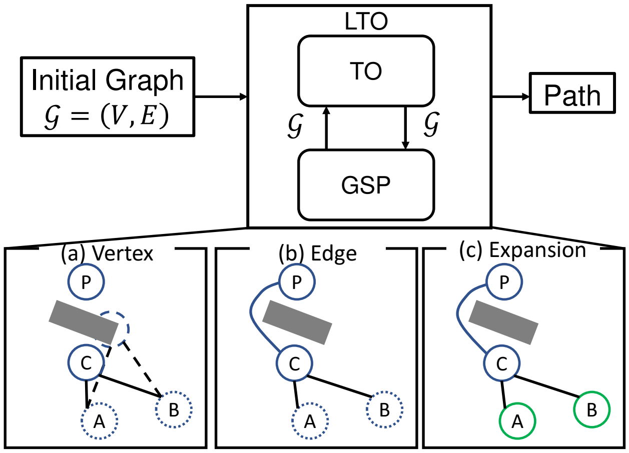


Figure 3.1: Overview of LTO. Given an initial graph where the true configuration of the robot and the trajectory are unknown, LTO iteratively solves TO locally to obtain the true vertex (configuration) and edges (trajectories) in $\mathcal{G}$. Based on the updated $\mathcal{G}$ by TO, GSP performs either expansion of the vertex, gets the true vertex using TO, or gets the true edge using TO. Assume the vertex P is the current vertex LTO chooses from an open list. In (a), LTO gets the true configuration of the vertex C using TO. In (b), it generates the true trajectory from the vertex P to C using TO. In (c), it expands the vertex C and inserts the vertices A and B into the open list.

we keep each vertex and edge within the voxels, the constraints outside the voxels do not have an influence on the generated vertex and edge in voxels. In other words, LTO cuts off unnecessary constraints from the original TO and only keeps the necessary constraints to generate vertices and edges in the voxels, resulting in the decreased solving time in TO. If we have a dense enough roadmap, we effectively solve the same constraints as the original long-horizon TO.

Here, we describe how we build our graph. At the start of planning, we make voxels in the continuous domain and place a vertex in each voxel. In this work, we target a complex motion planning problem. Thus, if we place a vertex to represent the robot's state without considering the feasibility of the state, the probability of the state associated with the vertex being infeasible would be high. In contrast, if we use TO to place a vertex, TO considers constraints so that LTO can place the vertex in a feasible region. Therefore, during the planning process, LTO uses TO to get the true configuration associated with the vertex and update $\mathcal{G}$. To execute TO, LTO associates each grid voxel with a continuous state of the

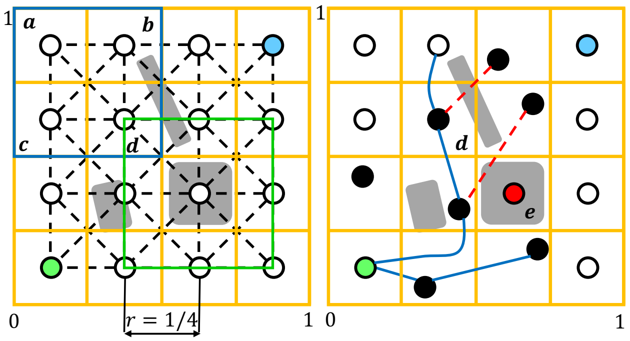


Figure 3.2: The planning procedure by LTO in $[0, 1]^2$ where $K = 4, i = 1, r = 1/4$. The left figure shows a graph at the start of planning where each vertex is inside the orange voxel and each edge is represented as a black dashed line. When LTO intends to expand the vertex, TO is solved within this voxel. When LTO investigates the edge, it solves TO within the associated voxels. For instance, when it investigates the edge from the vertex (a) to (d), it solves TO in the voxels (a), (b), (c), (d), taking into accout the constraints (e.g., obstacles) in the voxels. The right figure shows the current graph structure after iterations. For simplicity, it does not show the black dashed lines. The true vertex configurations and the true edges found by TO are shown as the black circles and blue lines, respectively. The infeasible vertex and edge judged by TO are shown as the red circle and red lines, respectively. For example, the vertex (e) is removed since TO cannot find the feasible vertex configuration in the voxel (e).

robot as illustrated in Fig. 3.2.

As for edge generation, LTO does not assume that the edge is straight-line since it does not assume a holonomic robot. Hence, during the planning process, it uses TO to get the true trajectory associated with the edge and update $\mathcal{G}$. When generating the edge, it solves TO in the hypercube, consisting of corner points of the voxels where the target vertices are located.

3.3.3 Mixed-Integer Convex Programs

The MICP to generate edges in $\mathcal{G}$ is given by:

$$\begin{aligned}
\text{minimize} \quad & c_T(x_N, z) + \textstyle\sum_{t=0}^{N-1} c_i(x_t, z) \\
\text{subject to} \quad & f_i(x_t, z) \leq 0, \quad t = 0, \ldots, N-1 \\
& x_{\min} \leq x_t \leq x_{\max}, \quad t = 0, \ldots, N \\
& x_0 = x_s, \quad x_N = x_g \\
& x_t \in \mathcal{X}, \quad t = 0, \ldots, N \\
& z \in \{0, 1\}^{n_z}
\end{aligned} \tag{3.1}$$

where x_t are the states of the robot at time t, z are binary decision variables, c_T, c_i, f_i are convex functions, and $\mathcal{X}$ shows the convex set. When finding the edge in $\mathcal{G}$, we solve (3.1), where x_s, x_g are the state of the start vertex and the state of the goal vertex, respectively. When finding the vertex in $\mathcal{G}$, we solve (3.1) with $N = 0$ without $\sum_{t=0}^{N-1} c_i\left(x_t, z\right)$.

3.3.4 Warm-Start Strategy

We use a warm-start to accelerate the planning process. Let v_p and v_c be the start and goal state in the trajectory LTO tries to generate in $\mathcal{G}$. LTO searches the most similar trajectory in $\mathcal{G}$ based on the deviation cost as follows:

$$d_{\text{cost}} = \|v_p - v_i\| + \|v_c - v_j\| \tag{3.2}$$

where v_i and v_j are the start and goal of other trajectories in $\mathcal{G}$. We assume that a trajectory in $\mathcal{G}$ with a close start and goal designs a similar trajectory, enabling the robot to be aware of complex constraints (e.g., dynamics, environment-oriented constraints). Therefore, when LTO tries to generate an edge that is not investigated by TO yet, it uses the edges that are already generated by TO as initial guesses for generating the current edge if d_{cost} is lower than the threshold.

Our strategy of a warm-start has several advantages. First, LTO solves many similar (almost identical) TOs in $\mathcal{G}$ and generates similar edges during the planning process. As time passes, LTO solves more edges and this information can enhance the quality of the warm-start for the current trajectory generation using TO. This feature cannot be provided if we use a regular TO without running the same problem previously. We can provide an initial guess, such as straight-line trajectory, to a solver for the regular TO, but it only works for the relatively straightforward problem. Additionally, we solve TO with dynamic constraints so that we can provide the value of dynamic variables to the next execution of TO, which is difficult if we use the trajectory from other planning methods such as SBP, which typically only considers kinematics.

3.4 Lazy Trajectory Optimization Formulation

We propose LTO that unifies TO and GSP. LTO does not have any prior knowledge about the vertices and edges in $\mathcal{G}$. We generate the vertex and the edge by running TO. We employ LWA* as GSP of LTO because it delays an edge generation until the planner intends to use the edge, resulting in the decreased planning time. However, LWA* has several limitations. First, it does not consider the difficulty of the edge generation based on TO. To solve this problem, we propose a new cost function that considers the time complexity (difficulty) of TO with the guaranteed suboptimality bound. Another limitation is that it assumes that the robot's configuration is already known before the planner runs. Since we focus on the complex planning problem, we cannot figure out the true configuration until TO runs. Because executing TO for all the voxels to validate the vertices is demanding, we also delay vertex validation using TO until the planner intends to expand the vertex.

The high-level process of LTO is shown in Fig. 3.1. Roughly speaking, LTO works as follows. Given a uniform grid graph where LTO does not know the true vertex and edge, LTO performs an action from three possible actions: an expansion, a vertex validation, and an edge generation. When the expansion is chosen, LTO expands the state, which means LTO generates all successors and puts them in an open list. If the vertex validation is chosen, TO solves the true configuration of the vertex within the voxel. When edge generation is chosen, TO solves to generate the trajectory between the vertices and update the true edge cost.

We use the notation $X \overset{+}{\leftarrow} \{\mathbf{x}\}$ and $X \overset{-}{\leftarrow} \{\mathbf{x}\}$ to show the compounding operations $X \leftarrow X \cup \mathbf{x}$ and $X \leftarrow X \setminus \mathbf{x}$, respectively. Q_o, Q_c are priority queues to maintain the states discovered but not expanded and the expanded states, respectively. $\hat{g}(v)$, $\hat{h}(v)$, and $\hat{f}(v)$ are estimates of cost-to-come, cost-to-go, and cost from the start to goal through v, respectively. We use $\hat{h}(v)$ as follows: $\hat{h}(v) = \|v_i - v_{goal}\|$. TrueVertex and TrueEdge show if a state v has the true configuration and the true edge cost, respectively. Conf(v) represents the true configuration of the vertex.

3.4.1 Trajectory Optimization-Aware Cost

We propose the TO-aware cost as follows:

$$\begin{aligned} c\left(v_1, v_2\right) &= \|v_1 - v_2\| \\ c_{TO}\left(v_1, v_2\right) &= (1 + \omega^{n_i}) c\left(v_1, v_2\right) \end{aligned} \tag{3.3}$$

where $c\left(v_1, v_2\right)$ is the cost of the edge using the Euclidean distance and $c_{TO}\left(v_1, v_2\right)$ is the inflated cost (i.e., overestimating cost) of the edge considering the time complexity of TO. n_i is the number of discrete decision variables associated with edge generation and we count n_i within the associated voxels. ω is a user-defined inflation factor.

With ω^{n_i}, c_{TO} can be very large so that LTO may not enthusiastically investigate the edge in the voxels with many discrete variables. In other words, ω is a tuning knob that balances the optimality of a trajectory and the planning time. We use c_{TO} as the cost of the edge if it is not investigated by TO and use c if it is investigated by TO.

While many papers have tried to minimize the number of edge evaluations [87], only a few papers discuss the "difficulty" of the edge evaluation. By recognizing that the running time of the edge generation by TO grows exponentially as the number of discrete variables increases [75], [88], we propose this cost function. In addition, by defining c_{TO} as the inflated c, we can bound the suboptimality in Theorem 3.5.3.

Algorithm 1 LTO($\mathcal{G}$, v_{start}, v_{goal})

```
 1: $Q_o \leftarrow v_{\text{start}},\ Q_c \leftarrow \emptyset,\ \hat{g}\,(v_{\text{start}}) = 0,\ \hat{g}(v) \leftarrow \infty$
 2: $\text{TrueVertex}(v) \leftarrow \text{False}, \text{TrueEdge}(v) \leftarrow \text{False}$
 3: $\text{TrueVertex}(v_{\text{start,goal}}) \leftarrow \text{True}, \text{TrueEdge}(v_{\text{start}}) \leftarrow \text{True}$
 4: while $\hat{f}\,(v_{\text{goal}}) > \min_{v \in Q_o}(\hat{f}(v))$ do
 5:     $v = \text{argmin}_{v \in Q_o}(\hat{f}(v)), Q_o \overset{-}{\leftarrow} \{v\}$
 6:     if $v == v_{\text{goal}}$ then
 7:         return ReconstructPath($v_{\text{start}}$, $v_{\text{goal}}$)
 8:     if $v \in Q_c$ then
 9:         CONTINUE
10:     else if TrueVertex($v$) then
11:         if TrueEdge($v$) then
12:             $Q_o, Q_c = \text{Expansion}(\mathcal{G}, Q_o, Q_c, v)$
13:         else
14:             $\mathcal{G}, Q_o, Q_c = \text{UpdateEdge}(\mathcal{G}, Q_o, Q_c, v)$
15:     else
16:         $\mathcal{G}, Q_o, Q_c = \text{UpdateVertex}(\mathcal{G}, Q_o, Q_c, v)$
17: return No Path Exists
```

3.4.2 Main Loop (Algorithm 1)

Lines 1-7 are typical of A*. We iteratively remove the cheapest state in Q_o until the goal is chosen. Lines 8-9 are from LWA*, showing that a state is not expanded again if it is already expanded and continues to the next iteration of the while loop. Lines 10-16 are new. TrueVertex(v) and TrueEdge(v) check if the expanded state v has the true configuration and the true edge cost, respectively.

3.4.3 Expansion (Algorithm 2)

In Algorithm 2, the expanded state has both the true vertex and the true edge cost so that LTO puts all successors of v in Q_o. GetSuccessors generates a copy of each neighboring

Algorithm 2 Expansion(Q_o, Q_c, v)

```
1: $Q_c \overset{+}{\leftarrow} \{v\}$, $S = \text{GetSuccessors}(v)$
2: for all $v' \in S$ do
3:     $\text{parent}(v') = v$
4:     if $\exists v'' \in (Q_o \textbf{or} Q_c)$s.t. $\text{TrueVertex}(v'')$ and$\text{Conf}(v') = \text{Conf}(v'')$ then
5:         $\hat{g}(v') = \hat{g}(\text{parent}(v')) + c_{TO}(\text{parent}(v'), v')$
6:         $\text{TrueVertex}(v') = \text{true}$
7:     else
8:         $\hat{g}(v') = \hat{g}(\text{parent}(v')) + c_{x,v}(\text{parent}(v'), v')$
9:     if $\nexists v'' \in Q_o$ s.t. $\text{Conf}(v'') = \text{Conf}(v')$ and $\text{TrueEdge}(v'')$ and$\hat{g}(v'') \leq \hat{g}(v')$ and$v' \notin Q_c$ then
10:        $\hat{f}(v') = \hat{g}(s') + \hat{h}(v')$, $Q_o \overset{+}{\leftarrow} \{v'\}$
11: return $Q_o, Q_c$
```

state to maintain the states from different parents (line 1). The same vertex that originated from other parent states might have already figured out the true configuration of the vertex by already running TO. Hence, LTO checks if other versions of the successor state v' have the true configuration in Q_o, Q_c (line 4). If true, we update $\hat{g}(v')$ with c_{TO}. Thus, $\mathcal{G}$ has an expensive cost for edges with many integer variables. We also set TrueVertex(v') to true (line 6). If another version of the successor state v' does not have the true configuration, we use a distance from v to the voxel's edge where v' belongs as a cost of the edge to guarantee the bounded suboptimality (line 8). LTO checks if this version of v' should be considered for maintaining in Q_o (line 9). If there exists the state v'' that represents the same configuration of v' with the true edge cost and the lower $\hat{g}$ value, we do not maintain v'.

3.4.4 Edge generation (Algorithm 3)

In Algorithm 3, the state has the true vertex but does not obtain the true edge cost yet. Let v_y^x and v_y^z be the vertices representing the same configuration but originated from different parents x, z. Here, $e(v_b^a \leftrightarrow v_d^c) = e(v_b^e \leftrightarrow v_d^f)$. Hence, we do not want to run expensive TO

Algorithm 3 UpdateEdge($\mathcal{G}, Q_o, Q_c, v$)

```
1: if CheckSamePair($\mathcal{G}$, parent($v$), $v$) is False then
2:     $w_{\text{opt}}$ = GetWarmStart($\mathcal{G}$, parent($v$), $v$)
3:     $c, \mathcal{G}$, Edge = RunTO($\mathcal{G}$, parent($v$), $v$, $w_{\text{opt}}$)
4: else
5:     $c, \mathcal{G}$, Edge = GetSamePair($\mathcal{G}$, parent($v$), $v$)
6: if Edge is feasible then
7:     TrueEdge ($v$) = true, $c$ = CostSamePair(parent($v$), $v$))
8:     $\hat{g}(v) = \hat{g}(\text{parent}(v)) + c$
9:     if $\nexists v'' \in Q_o$ s.t. Conf ($v''$) = Conf ($v$) and TrueEdge($v''$) and $\hat{g}(v'') \leq \hat{g}(v)$ then
10:        $\hat{f}(v) = \hat{g}(v) + \hat{h}(v)$, $Q_o \xleftarrow{+} \{v\}$
11: return $\mathcal{G}, Q_o, Q_c$
```

again to get $e(v_b^a \leftrightarrow v_d^c)$ if we already obtain $e(v_b^e \leftrightarrow v_d^f)$. On line 1, CheckSamePair checks if we already obtain the same configuration pair from different parents. If true, we get the same configuration pair (line 5). Line 6 checks if the obtained edge is feasible. If true, we set TrueEdge(v) to true, get the cost (line 7) and use it to update the $\hat{g}(v)$ (line 8). We use c instead of c_{TO} because LTO already figures out the true edge cost, and it does not make sense for the edge cost to be expensive due to ω^{n_i}. On lines 9-10, like line 9 in Algorithm 2, we insert v into Q_o if no states exist satisfying the if condition. If false on line 1, LTO runs TO (line 3) and perform the same action between lines 7-10. On line 2, GetWarmStart computes the initial guesses w_{opt} of each decision variable.

3.4.5 Vertex Validation (Algorithm 4)

Since the state does not have the true vertex, LTO runs TO and gets the true configuration of v. The structure of Algorithm 4 and Algorithm 3 is essentially the same, but in Algorithm 4, we use c_{TO} as the cost to avoid the expensive edge generation (line 7).

Algorithm 4 UpdateVertex($\mathcal{G}, Q_o, Q_c, v$)

```
1: if CheckSameVertex($\mathcal{G}, v$) is False then
2:     $\mathcal{G}$, Configuration = RunTO($\mathcal{G}, v$)
3: else
4:     $\mathcal{G}$, Configuration = GetSameVertex($\mathcal{G}, v$)
5: if Configuration is feasible then
6:     TrueVertex ($v$) = true
7:     $g(v) = g(\text{parent}(v)) + c_{TO}(\text{parent}(v), v)$
8:     if $\nexists v'' \in Q_o$ s.t. $\text{Conf}(v'') = \text{Conf}(v)$ and TrueEdge($v''$) and $g(v'') \leq g(v)$ then
9:         $\hat{f}(v) = \hat{g}(v) + \hat{h}(v)$, $Q_o \xleftarrow{+} \{v\}$
10: return $\mathcal{G}, Q_o, Q_c$
```

3.5 Formal Analysis

We prove several essential properties in LTO.

3.5.1 Complexity

Theorem 3.5.1. *The running time of LTO is bounded by* $O(EV^2 \log V)$*, where* E*,* V *are the number of edges, vertices.*

Proof. Because LTO maintains the duplicate states, the maximum number of states Q_o contains is V^2. In Algorithm 1, the outer loop runs at most V^2, and line 5 is $O(\log V^2)$ with priority queues. Thus, the total running time is: $O\left(V^2\left(\log V^2 + \text{Alg.2} + \text{Alg.3} + \text{Alg.4}\right)\right)$. For Algorithm 2, the outer loop runs at most $|S|$, TrueVertex investigates at most V states, line 9 investigates at most V states, and line 10 takes $O(\log V)$. Thus, $\text{Alg.2} = O(|S| V \log V^2)$. For Algorithm 3, CheckSamePair investigates at most V^2, line 9 investigates at most V states, line 10 takes $O(\log V^2)$. We have the same discussion for Algorithm 4. □

While this time complexity is worse than that of other algorithms (e.g., A*: $O(E \log V)$), LTO finds a solution quickly in practice. For planning with expensive edge generation, it makes more sense to discuss the time complexity based on TOs. We identify line 3 in Algorithm 3 and line 2 in Algorithm 4 as the main sources of planning time. Recognizing this fact, we show the TO-aware time complexity.

Theorem 3.5.2. *Let $\mathcal{X}$ be the configuration space normalized to $[0,1]^d$, where $d \in \mathbb{N}$. Let K, r be the number of intervals along each axis and the normalized distance calculated as ℓ_∞ (see Fig. 3.2). Then, the TO-aware time complexity is $O((2i+1)^d K^d)$ where $i = 0, 1, \cdots, K$, $r = (1/K)i$.*

Proof. Fig. 3.2 shows the case where $d = 2, K = 4, r = 1/4, i = 1$. The total number of vertices is K^d so TO for finding a vertex is called at most K^d times. Regarding the edge generation, LTO connects the vertices if the ℓ_∞ between the center of the voxel to which one vertex belongs and the center of the voxel to which the other vertex belongs to is less than or equal to r. The total number of edges per vertex is $((2i+1)^d - 1)$ (the vertices inside the green rectangle in Fig. 3.2 except for the vertex (d)). Thus, the total number of TO to find the edge is $((2i+1)^d - 1)K^d$. □

We can even bound K when TO uses MICP.

Corollary 1. *K is bounded as: $T_o(2^B K^d + 2^{NB(i+1)^d}((2i+1)^d - 1)) \leq T_t$ where B is the maximum number of integer variables in a voxel, T_o is the average solving time of convex programming on a problem domain with no integer variables and $N = 0$, and T_t is the acceptable running time.*

Proof. When finding a true vertex, the MICP solver using B&B searches at most 2^B solutions and solves the regular convex programming for each solution with relaxed integer constraints. In the worst-case, it solves convex programs for all voxels and the solving time is $2^B K^d T_o$. Regarding the edge generation, the solver investigates $(i+1)^d$ voxels at most for every single edge. When finding edges, we consider N steps planning problem so that the total number of integer variables that TO has when finding an edge is $NB(i+1)^d$. Because the solver runs at most $((2i+1)^d - 1)$ per a voxel to find an edge, the worst-case solving time for edge generation is $2^{NB(i+1)^d}((2i+1)^d - 1)$. □

Given T_t, T_o, LTO can estimate the computational margin quantitatively by tuning K.

The space complexity in our algorithm is $O(V^2 + E)$ since LTO investigates at most V^2 vertices and E edges.

3.5.2 Efficiency

Line 1 in Algorithm 4 gets the vertices of the same configuration and checks if the vertex from other parents has the true configuration. If true and the configuration is feasible, line 6 marks the vertex as having true configuration. Once it is marked as true, there are no other lines to set the vertex marker to false. We have the same discussion for TrueEdge. Therefore, LTO solves TO at most once for the vertex and edges representing the same configuration.

3.5.3 Completeness and Optimality

LTO is complete. Since it evaluates all the edges in the worst-case, it eventually reduces to A*, which is complete.

We can bound the cost of the solution as follows:

Theorem 3.5.3. *Let* ξ^* *be an optimal path. LTO return a path* ξ *with cost* $c(\xi) \leq \alpha c(\xi^*)$ *with* $\alpha = (1 + \omega^M)$ *where* $M = NB(i+1)^d$.

Proof. If there are no integer variables in the domain, GSP in LTO reduces to A*, resulting in $\hat{g}(v) \leq g^*(v)$, where $g^*(v)$ is the optimal cost-to-come, as shown in Theorem 10 in [89]. Next, consider the domain with integer variables. We prove that we can bound the suboptimality with c_{TO}. To prove this, we need to show $\hat{g}(v) \leq \alpha g^*(v)$. We use induction. At the start of planning, $\hat{g}(v_{\text{start}}) = g^*(v_{\text{start}}) \leq \alpha g^*(v_{\text{start}})$ so the base case holds. Next, after some iteration of Algorithm 1 and assume that $\hat{g}(v) \leq \alpha g^*(v)$ holds for all $v \in \xi$ so far. Let $v_p \in \xi$ with $\hat{g}(v_p) > \alpha g^*(v_p)$, resulting in $\hat{f}(v_p) > \alpha(g^*(v_p)) + \hat{h}(v_p)$. It is obvious that $\hat{g}(v) \leq \alpha g^*(v)$ holds if no such v_p exists. Here, we show that LTO will not choose such v_p on line 5 in Algorithm 1 even if v_p exists.

Case 1: A vertex has been expanded beore v_p along ξ. In this case, We must have a $v_{a-1} \in \xi$ before v_p along ξ with successor v_a on Q_o. If TrueEdge(v_a) is true:

$$\hat{g}(v_a) \leq \hat{g}(v_{a-1}) + c(v_{a-1}, v_a)$$
$$\leq \alpha g^*(v_{a-1}) + c(v_{a-1}, v_a) \leq \alpha g^*(v_a)$$

If TrueEdge(v_a) is false:

$$\hat{g}(v_a) \leq \hat{g}(v_{a-1}) + c_{TO}(v_{a-1}, v_a)$$
$$\leq \omega^{n_i}(g^*(v_{a-1}) + c(v_{a-1}, v_a))$$
$$\leq \omega^M(g^*(v_{a-1}) + c(v_{a-1}, v_a)) = \alpha g^*(v_a)$$

Hence, the assumption $\hat{g}(v) \leq \alpha g^*(v)$ holds true for all iterations. Since for every vertex $\alpha g^*(v_i) + \hat{h}(v_i) \leq \alpha g^*(v_{i+1}) + \hat{h}(v_{i+1})$ is true due to the consistency of $\hat{h}$,

$$\hat{f}(v_a) = \hat{g}(v_a) + \hat{h}(v_a) \leq \alpha(g^*(v_a)) + \hat{h}(v_a)$$
$$\leq \alpha(g^*(v_p)) + \hat{h}(v_p) < \hat{f}(v_p)$$

which means that v_p will not be chosen.

Case 2: No expanded vertex before v_p along ξ. In this case, Q_o must contain the start vertex, where we can apply the same discussion above, resulting in $\hat{f}(v_{\text{start}}) < \hat{f}(v_p)$.

Finally, $c(\xi) = \hat{g}(v_{\text{goal}}) \leq \alpha g^*(v_{\text{goal}}) \leq \alpha c(\xi^*)$. $\square$

3.6 Numerical Experiments

We validate LTO on two motion planning problems: free-flying robots in $\mathbb{R}^2$ and legged robots in $\mathbb{R}^{21}$. For legged robot problems, we conduct the experiments for three different environments. We test algorithms with ten trials except for SBP algorithms, which we evaluate for five trials.

To set up experiments, we evaluate LTO of the different numbers of voxels with/without the warm-start option. We set r such that a vertex has 15 edges per vertex for the free-flying robot experiments and 20 edges for the legged robot experiments. We use the Euclidean distance in configuration space as c. With active warm-start options, LTO employs other already investigated trajectories by TO if d_{cost} in configuration space is less than 0.1. To compare with LTO, we also run the regular TO (i.e., no GSP is embedded), other GSP (i.e., weighted A*, LWA*), and SBP (i.e., PRM [90], lazyPRM [91], RRT [92], RG-RRT [93]). To have a fair comparison, we incorporate TO into GSPs and SBPs. It means that when the GSP and SBP find a node and connect nodes, it uses TO to figure out if the sampled configuration and the edge is feasible, just like LTO. In fact, given sufficient planning time, regular SBP and GSP algorithms (i.e., just sample nodes without TO) do not return any single feasible trajectory.

We use Gurobi [94] to solve MICP on Intel Core i7-8750H machine and implement all planning codes in Python.

3.6.1 Free-Flying Robots

We consider a free-flying robot in $\mathbb{R}^2$ with multi obstacles. We define $p_t \in \mathbb{R}^2$ as the position and $v_t \in \mathbf{R}^2$ as the velocity. The state $x_t = (p_t, v_t)$ is controlled by $u_t \in \mathbb{R}^2$. Thus, the robot solves the following MICP from x_{start} to x_{goal} while remaining in the safe region $\mathcal{X}_{\text{safe}}$ [95]:

$$\begin{array}{ll}
\text{minimize} & \sum_{\tau=0}^{N-1} \left(x_\tau - x_g\right)^\top Q \left(x_\tau - x_g\right) + u_\tau^\top u_\tau \\
\text{s.t.} & x_{t+1} = Ax_t + Bu_t, t = 0, \ldots, N-1 \\
 & \|u_t\|_2 \leq u_{\max}, \quad t = 0, \ldots, N-1 \\
 & x_{\min} \leq x_t \leq x_{\max}, \quad t = 0, \ldots, N \\
 & x_0 = x_{\text{start}}, \quad x_N = x_{\text{goal}} \\
 & x_t \in \mathcal{X}_{\text{safe}}, \quad t = 0, \ldots, N
\end{array} \tag{3.4}$$

The computationally demanding constraints $x_t \in \mathcal{X}_{\text{safe}}$ due to integer variables z are represented using a standard big-M formulation with binary variables [95]. We consider axis-aligned rectangular obstacles. We run LTO under 1000 and 2000 voxels with the inflation factor $\omega = 0, 100$ and with/without a warm-start option. We set $N = 7$ for TO inside LTO. For TO, we use $N = 70$, which is the minimum number of N for us to get a feasible trajectory. In this case, the number of continuous variables, binary variables, and constraints is 168, 3360, 4230, respectively. With $N < 70$, we only get trajectories that get stuck at local optima. For GSP and SBP, we run at most 5000 samples.

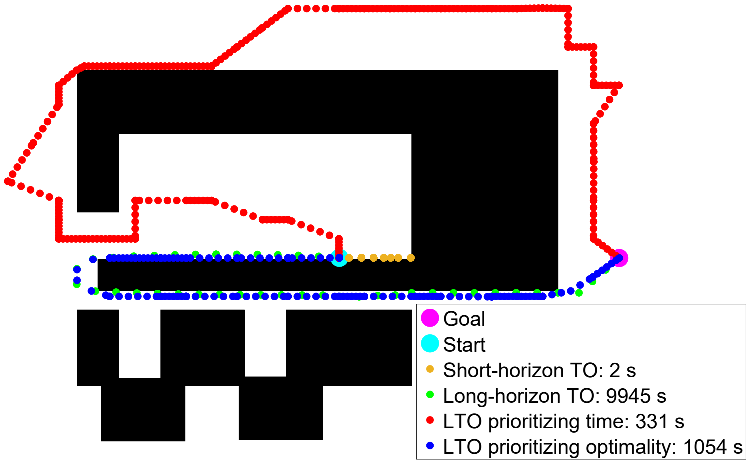

Goal
Start
Short-horizon TO: 2 s
Long-horizon TO: 9945 s
LTO prioritizing time: 331 s
LTO prioritizing optimality: 1054 s

Figure 3.3: Generated trajectories in $\mathbb{R}^2$. The short-horizon TO gets stuck at the local optimum and cannot find the trajectory from start to goal. The long-horizon TO finds the optimal trajectory but takes an extended amount of time. LTO prioritizing planning time avoids the area where the time for solving TO is long due to many integer variables (i.e., obstacles). LTO prioritizing optimality of the trajectory finds the resolution-optimal solution with less planning time compared with the long-horizon TO.

The solution cost versus planning time is plotted in Fig. 3.4. LTO finds as good solutions as TO finds with decreased planning time. The generated trajectories are shown in Fig. 3.3. By navigating the robot to a region with fewer integer variables (i.e., fewer obstacles), LTO can generate robot trajectories quickly. We also observe that the lower the inflation factor ω is, the more optimal trajectory LTO generates. It takes more time to design the trajectory since LTO does not guide the robot to avoid the computationally expensive regions, but it still quickly generates the trajectory without sacrificing the solution cost so much compared with the long-horizon TO.

We discuss Fig. 3.4 by comparing the results of LTO with TOs. While the best planner among LTOs (LTO with 2000 voxels with $\omega = 0$) designs the trajectory that cost is 1.2 % worse than TO with a cost function, it decreases the computation time by about 89 %. Furthermore, with the warm-start option, LTO decreases the computation time by about 93 %. In addition, TO with $N = 70$ is the simplest TO we get the feasible global trajectories. We manually increase N until we get the feasible trajectory. In contrast, since LTO iteratively performs the short-horizon TO, we do not need to spend time tuning N until we get global

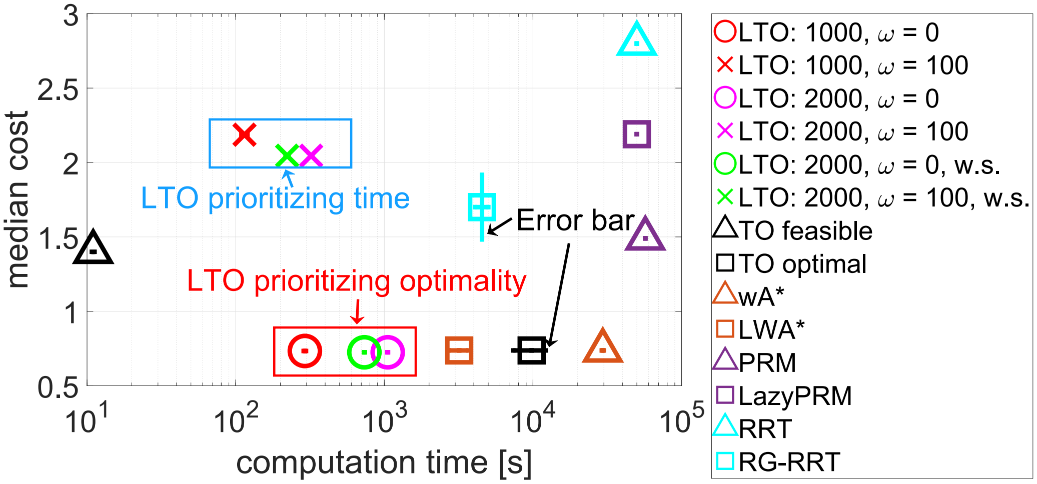


Figure 3.4: The results of Section 3.6.1. Error bars represent a 95 % confidence interval for a Gaussian distribution. Note that for some algorithms, the confidence intervals are very small and are not visible. Compared with TO, LTO prioritizing optimality (i.e., $\omega = 0$) finds the optimal solution about nine times faster without sacrificing the solution cost much (1.2 % worse).

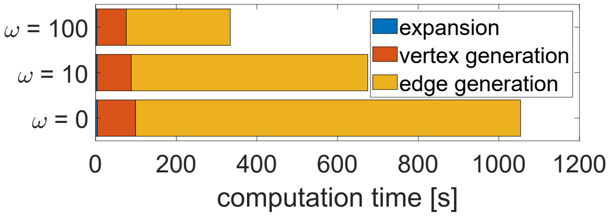


Figure 3.5: Consumed time with 2000 voxels in $\mathbb{R}^2$ for LTO. The larger the inflation factor ω is, the less time LTO spends by avoiding expensive edge generation.

feasible trajectories, resulting in less offline user time consumption. Another advantage of LTO is that the variance of the planning time of LTOs is much smaller than that of TO as shown in Fig. 3.4. Recognizing that the solver's behavior in terms of solving time is more uncertain for the large-scale optimization problem than for the small-scale problem [96], LTO employs the small-scale problem (i.e., short-horizon TO) to have the small variance of the planning time.

We also evaluate our algorithms in terms of parameters. Fig. 3.4 shows that ω provides a tuning knob to trade off between the solution cost and the planning time. Fig. 3.5 shows the individual time cost on Algorithm 2-Algorithm 4 within LTO with different ω for 2000 voxels. It shows that TO for generating an edge spends a large amount of time. It also indicates that by increasing ω, LTO spends less time to generate edge by navigating the robot to the region with fewer discrete variables, resulting in less total planning time. As for the number of voxels, one insight is that LTO with 1000 voxels and $\omega = 0$ designs the trajectory, resulting in 72 % decreased planning time with the 0.4 % worse solution cost compared with LTO with 2000 voxels and $\omega = 0$. As a result, we may use the solution from LTO with 1000 voxels and $\omega = 0$. One future work would be how we balance the number of vertices and the difficulty of TO: the more vertices, the easier TO is since TO solves within smaller voxels, but LTO needs to run TO more times. One advantage of LTO is that it does not require many voxels since TO can find the feasible vertices and edges while satisfying constraints. Therefore, we can get the optimal grid resolution by balancing the difficulty of TO and the number of calls to TO.

While LTO and TO have the success rate of 100 %, SBP and GSP have the success rate of 20 % except for RG-RRT, which has that of 40 %.

3.6.2 Legged Robots

We consider a M-legged robot motion planning problem. We denote the body position as $q_t \in \mathbb{R}^3$, its orientation as $\theta_t \in \mathbb{R}^3$, and toe i position as $p_{it} \in \mathbb{R}^3$. To realize a stable locomotion, we consider the reaction force $f_{it}^r \in \mathbb{R}^3$ at foot i. Thus, the robot solves the following MICP [75]:

$$\begin{array}{ll}
\text{minimize} & \sum_{\tau=0}^{N-1} \left(x_\tau - x_g\right)^\top Q \left(x_\tau - x_g\right) \\
\text{s.t.} & x_{\min} \leq x_t \leq x_{\max}, \quad t = 0, \dots, N \\
& |\Delta x_t| \leq \Delta x, \quad t = 1, \dots, N \\
& p_{it} \in \mathcal{R}_i(q_t, \theta_t), \quad t = 0, \dots, N \\
& x_0 = x_{\text{start}}, \quad x_N = x_{\text{goal}} \\
& x_t \in \mathcal{X}_{\text{safe}}, \quad t = 0, \dots, N \\
& \sum_{i=1}^{M} f_{it}^r + F_{tot} = 0 \\
& \sum_{i=1}^{M} \left(p_{it} \times f_{it}^r\right) + M_{tot} = 0
\end{array} \tag{3.5}$$

where x_t contains kinematics-related decision variables $q_t, \theta_t, p_{1t}, \cdots, p_{Mt}$. Here, $p_{it} \in \mathcal{R}_i(q_t, \theta_t)$ shows kinematics constraints. $\sum_{i=1}^{M} f_{it}^{r} + F_{tot} = 0$ and $\sum_{i=1}^{M} (p_{it} \times f_{it}^{r}) + M_{tot} = 0$ represent the static equilibrium of force and moment constraints, respectively. For kinematics constraints, we approximate them as linear constraints [14]. For the static equilibrium of moment containing bilinear terms, we use piecewise McCormick envelopes to relax the bilinear terms into convex terms, which uses binary variables z to specify the partition in an envelope [75]. Our current limitation of LTO is that it cannot deal with orientation in a heuristic with guaranteed suboptimality. To consider orientation, we may utilize the multi-heuristic approach [97]. To have a fair comparison, we just consider q_t, p_{it} for planning. We consider a six-legged robot with 3 DOF per leg, resulting in 21 DOF planning. We run LTO under 3000 and 6000 voxels with $\omega = 0, 10$ and with/without a warm-start option. We set $N = 7$ for TO inside LTO. For TO, we use $N = 56, 63, 70$ from the left to the right environment in Fig. 3.6, respectively, which are the minimum number of N for us to get a feasible trajectory. The number of continuous variables is 20220, 23106, 25674, the number of binary variables is 32400, 14190, 6754, and the number of constraints is 74680, 55827, 48532, from the left to the right environments, respectively.

The generated trajectories and the solution cost versus planning time are shown in Fig. 3.6. In the left and middle environments, LTO shows better performance compared with TO. In Fig. 3.4, TO feasible planning finds the trajectory most quickly, but Fig. 3.6 shows that it spends more time to just find a feasible solution if the planning problem is more difficult. In the right environment, we observe that TO optimal and LTO shows a similar result. Because the right environment has fewer discrete variables, we do not observe the advantage of using LTO. Therefore, LTO works well in environments with many discrete decision variables like the left and the middle environments.

3.7 Conclusion

We presented LTO for high DOF robots in cluttered environments. Because LTO deeply unifies TO and GSP algorithms, it can consider the original long-horizon TO problem with

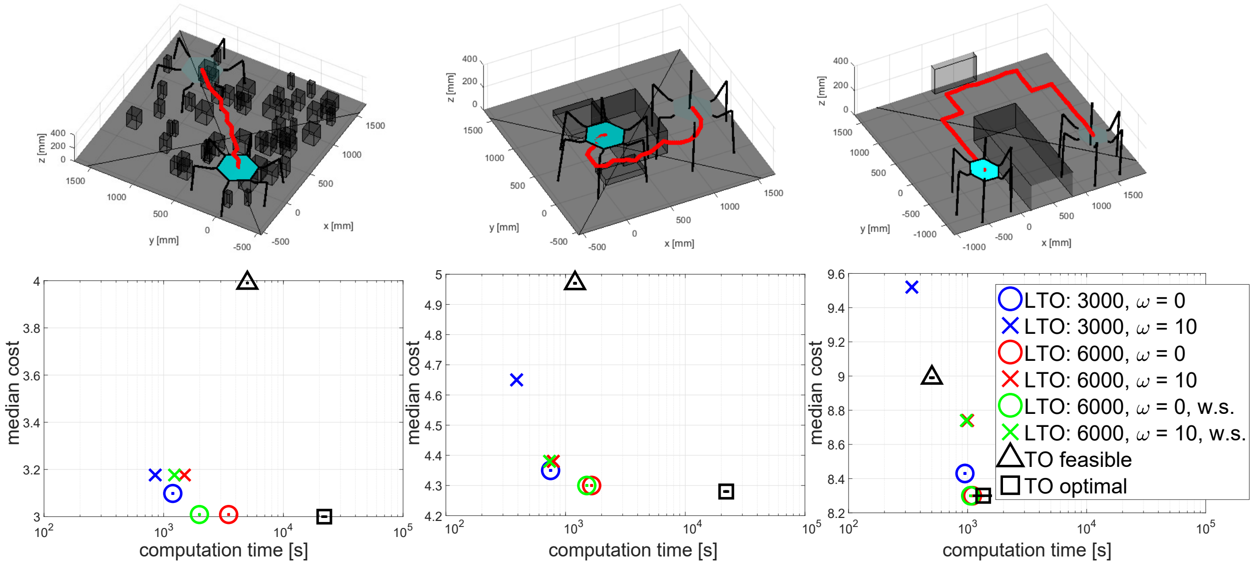


Figure 3.6: The results of Section 3.6.2 in $\mathbb{R}^{21}$. The red lines in the top figure indicate the body trajectory. Error bars represent a 95 % confidence interval for a Gaussian distribution. Note that for LTO, the confidence intervals are very small and are not visible. LTO with 6000 voxels with the warm-start finds the optimal solution about 11.3, 14.0, 1.4 times faster than TO without degrading the solution cost so much (about 0.33, 0.35, 0.01 % worse), from the left to the right environment, respectively. By increasing the inflation factor ω, LTO can generate globally suboptimal trajectories faster than TO feasible option.

a decreased planning time. We proposed a TO-aware cost function that considers the difficulty of TO. by recognizing that B&B depends on the number of discrete decision variables. Furthermore, LTO employs other edges in the graph as a warm-start to accelerate the planning process. We also presented proofs of the complexity and suboptimality. Finally, we performed planning experiments of a free-flying robot and a legged robot motion planning problems, showing that LTO is faster with a small variance of the planning time.

Since LTO has a small variance in planning time, we argue that it would be useful for safety-critical applications such as autonomous driving. Additionally, it consists of TO and GSP so that users can use other planning algorithms for each subcomponent according to their specifications.

Although we consider complex constraints such as kinematics, dynamics and collision avoidance constraints, the contact mode constraints are not considered, which is discussed in the next chapter.

CHAPTER 4

Planning through Contact using Decomposition-based Optimization for Multi-Limbed Robots

In this chapter, we present planning through contact for contact-rich robotic systems using ADMM. In particular, we focus on discussing motion planning of multi-limbed robots equipped with two-finger grippers for climbing on multiple discrete climbing holds although the proposed algorithm can be applied to any other contact-rich locomotion and manipulation tasks. In this chapter, we first argue that the current optimization-based planners have computational bottlenecks as the number of contact constraints increases. Then, we describe our patch contact model, which is important for the robot to achieve stable climbing. We present our ADMM-based optimization algorithm. Finally, we show some results including hardware experiments of climbing, computational complexity, and contact models.

This chapter has been partially adapted from one conference paper:

- **Y. Shirai**, X. Lin, A. Schperberg, Y. Tanaka, H. Kato, V. Vichathorn, and D. Hong, "Simultaneous Contact-Rich Grasping and Locomotion via Distributed Optimization Enabling Free-Climbing for Multi-Limbed Robots", in *Proc. 2022 IEEE/RSJ Int. Conf. Intel. Robot. Syst.*, pp. 13563-13570, 2022.

4.1 Overview

While legged robots have shown remarkable success in locomotion tasks such as running, legged robots with dexterous manipulation skills, defined as limbed robots, are relatively unexplored. There are a number of promising applications for limbed robots such as manip-

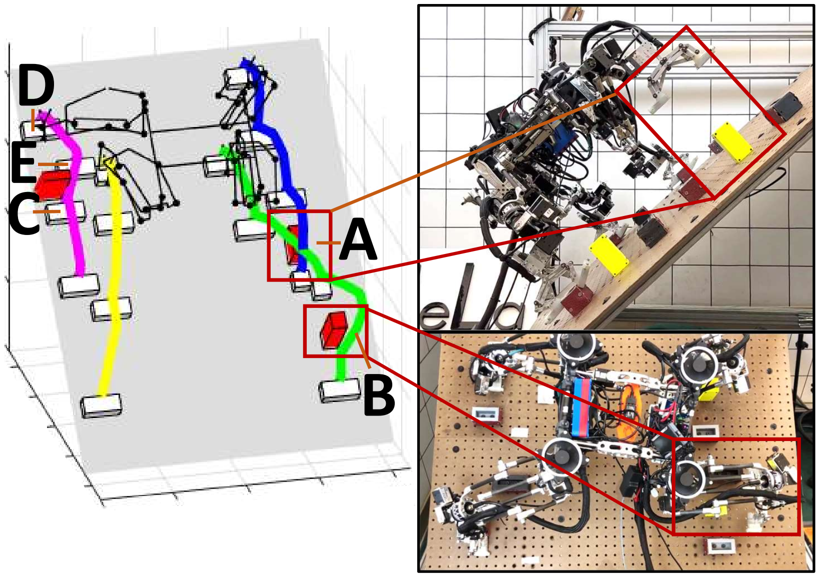


Figure 4.1: We consider motion planning of multi-limbed robots for free-climbing. Our proposed framework efficiently generates trajectories for multi-limbed robots equipped with multi-finger grippers while considering locomotion, grasping, and contacts. The left figure shows trajectories of one of the fingers of each gripper while the robot avoids obstacles. The trajectories around A-E are discussed in Sec 4.4.3. The right figure shows that our real four-limbed robot executes our planned trajectory.

ulating balls [98], sitting [99], bobbin rolling [100], pushing heavy objects [25], stair-climbing [27, 101], and free-climbing [102, 103, 104]. In this chapter, we focus on free-climbing tasks of limbed robots. Free-climbing capabilities would be useful for planetary exploration, inspection, and so on [105]. These tasks cannot be done by traditional legged robots by dismissing these problems as locomotion tasks. All of those previous works consider coupling effects between body stability of legged robots and frictional interaction of manipulators to some extent. To implement those capabilities in a real limbed robot, a variety of physical constraints need to be considered. Thus, motion planning plays a key role to generate physically feasible trajectories of limbed robots for those non-trivial tasks.

However, motion planning of limbed robots for such tasks is challenging. First, motion planning can be quite complicated because planners need to solve trajectories of legged robots, manipulators, or grippers together, which leads to NLP if motion planning is formulated as an optimization problem. Also, it is difficult to identify contact sequences prior to motion planning if the task is complicated such as free-climbing. Thus, it is required to consider locomotion, manipulation, and contacts together for generating trajectories, which results in computationally intractable MINLP.

Another problem arises when limbed robots interact with environments using patch contacts. With patch contacts, limbed robots can effectively increase friction forces and use friction torques generated on the patch, which is useful for manipulation tasks and even free-climbing tasks. However, multi-finger patch contacts are not discussed yet in previous motion planning works for limbed robots.

In this chapter, we propose a motion planning algorithm that efficiently solves locomotion, grasping, and contact dynamics together. We show that the resulting framework is computationally more tractable and less sensitive to parameters. Also, we explicitly discuss the patch contact constraints with micro-spine grippers, which enables the algorithm to realize dexterous multi-finger tasks. Our proposed motion planning is validated on a 9.6 kg four-limbed robot with spine grippers for free-climbing. To the best of our knowledge, it is one of the first works that demonstrate dexterous multi-finger grasping enabling free-climbing on a real multi-limbed robot.

This chapter presents the following contributions:

1. We present an optimization-based motion planning framework that simultaneously solves constraints from locomotion, grasping and contact dynamics.
2. We accelerate the entire optimization process by formulating the problem as a decomposition-based optimization.
3. We explicitly formulate patch contact constraints for micro-spine grippers.
4. We validate our framework in hardware experiments.

4.2 Patch Contact Model with Micro-Spines

In this section, we discuss the patch contact model used in our planner. We extend the previous works [106, 107, 48] and explicitly incorporate the limit surface in our proposed planner. As there is a normal force acting on the patch, the limit surface is composed of the gripper force failure model and the friction failure model. The total available reaction wrench $\mathbf{w} = [f^x, f^y, f^z, \tau^x, \tau^y, \tau^z]$ lives in a Minkowski sum of those two models described by:

$$\mathbf{w} \in \mathcal{W},\ \mathcal{W} = \{\mathbf{w}_{\mathrm{fr}} + \mathbf{w}_{\mathrm{sp}} | \mathbf{w}_{\mathrm{fr}} \in \mathcal{W}_{\mathrm{fr}}, \mathbf{w}_{\mathrm{sp}} \in \mathcal{W}_{\mathrm{sp}}\} \tag{4.1}$$

where we define z-axis as the direction of normal forces and x- and y-axis consist xy plane where shear forces exist (e.g., see $\Sigma_{c=3}$ in Fig. 4.2). $\mathbf{w}_{\mathrm{fr}} = [f^x_{\mathrm{fr}}, f^y_{\mathrm{fr}}, f^z_{\mathrm{fr}}, \tau^x_{\mathrm{fr}}, \tau^y_{\mathrm{fr}}, \tau^z_{\mathrm{fr}}]$ is the friction wrench and $\mathbf{w}_{\mathrm{sp}} = [f^x_{\mathrm{sp}}, f^y_{\mathrm{sp}}, f^z_{\mathrm{sp}}, \tau^x_{\mathrm{sp}}, \tau^y_{\mathrm{sp}}, \tau^z_{\mathrm{sp}}]$ is the wrench that can be supported by the micro-spines with the zero normal force. $\mathcal{W}_{\mathrm{fr}}, \mathcal{W}_{\mathrm{sp}}$ represent a frictional limit surface and a limit surface from micro-spines, respectively.

4.2.1 Frictional Limit Surface

Previous literature [106] models the friction wrench failure model as a simple 4D ellipsoid on $[f^x_{\mathrm{fr}}, f^y_{\mathrm{fr}}, f^z_{\mathrm{fr}}, \tau^z_{\mathrm{fr}}]$ as follows:

$$\mathcal{W}_{\mathrm{fr}} = \{\mathbf{w}_{\mathrm{fr}} \in \mathbb{R}^6 | \frac{(f^x_{\mathrm{fr}})^2 + (f^y_{\mathrm{fr}})^2}{(\mu f^z_{\mathrm{fr}})^2} + \frac{(\tau^z_{\mathrm{fr}})^2}{(k\mu f^z_{\mathrm{fr}})^2} \leq 1, 0 \leq f^z_{\mathrm{fr}} \leq f_{\mathrm{max}}, \tau^x_{\mathrm{fr}} = \tau^y_{\mathrm{fr}} = 0\} \tag{4.2}$$

such that μ is the coefficient of friction, k is an integration constant, f_{max} is the upper bound of f^z_{fr}. This work assumes that fingers make circular patch contact under uniform pressure distribution and thus we use $k = 0.67 r_p$ where r_p is the radius of contact [106].

4.2.2 Limit Surface of Micro-Spines

The authors in [107] construct the limit surface for spine grippers of any contact angles with the constant μ. If the contact angle ϕ in Fig. 4.2 begins to vary, μ may become a function of ϕ and requires an independent model. In practice, building such a model requires a large amount of data.

This work simplifies the model by assuming that the patch is always perpendicular to the surface during contact (i.e., $\phi = \frac{\pi}{2}$). Therefore, μ is constant. We also make the assumption that the moment $\tau_{\text{fr}}^x, \tau_{\text{fr}}^y$ are negligible since the size of the patch is relatively small. However, the moment τ_{fr}^z cannot be neglected according to our test data. Thus, we impose a following 3D limit surface on $[f_{\text{sp}}^x, f_{\text{sp}}^y, \tau_{\text{sp}}^z]$:

$$\begin{aligned} \mathcal{W}_{\text{sp}} = \{\mathbf{w}_{\text{sp}} \in \mathbb{R}^6 | -f_{\text{max}}^i \leq f_{\text{sp}}^i \leq f_{\text{max}}^i, i = \{x, y\}, \\ f_{\text{sp}}^z = 0, \tau_{\text{sp}}^x = \tau_{\text{sp}}^y = 0, -\tau_{\text{max}}^z \leq \tau_{\text{sp}}^z \leq \tau_{\text{max}}^z\} \end{aligned} \tag{4.3}$$

where $f_{\text{max}}^i, \tau_{\text{max}}^i$ represent the upper bound of each wrench.

4.2.3 Limit Surface for Two-Finger Micro-Spine Grippers

For the two-finger gripper used by our robot, each finger is equipped with a micro-spine patch with the total available variables $\mathbf{w}_1 = [f_1^x, f_1^y, f_1^z, \tau_1^z] \in \mathcal{W}$ and $\mathbf{w}_2 = [f_2^x, f_2^y, f_2^z, \tau_2^z] \in \mathcal{W}$, where $\mathbf{w}_1$ is for finger 1 and $\mathbf{w}_2$ is for finger 2. Since the rotational motion along z-axis for finger 1 and finger 2 are same and the linear motion along x- and y-axis are same (see Sec 4.3.1), we assume that the loading shear force and moment between two contact patches are identical:

$$f_1^i = f_2^i, i = \{x, y\}, \tau_1^z = \tau_2^z \tag{4.4}$$

This can be justified as there cannot be an additional twisting moment along z-axis from the object being grasped.

4.3 Simultaneous Contact-Rich Grasping and Locomotion Trajectory Optimization

In this section, we present our proposed optimization formulation which simultaneously solves grasping and locomotion while considering discrete dynamics such as gait sequence. Then, we derive the computationally tractable formulation of our proposed formulation based on

Table 4.1: Notation of variables. C or B indicates the variable is continuous or binary variables, respectively. In Σ column, we indicate the frame of variables. Subscript t indicates time-step.

Name	Description	Size	C/B	Σ
$\mathbf{r}_t$	body position	$\mathbb{R}^3$	C	W
$\boldsymbol{\theta}_t$	body orientation	$\mathbb{R}^3$	C	W
$\mathbf{p}_t^i$	i-th finger position	$\mathbb{R}^3$	C	W
$\mathbf{q}_t^i$	i-th finger orientation	$\mathbb{R}^3$	C	W
$\mathbf{d}_t^i$	l-th gripper distance between fingers	$\mathbb{R}^3$	C	W
$\boldsymbol{\lambda}_t^i$	i-th finger reaction force	$\mathbb{R}^3$	C	W
$\boldsymbol{\tau}_t^i$	i-th finger reaction moment	$\mathbb{R}^3$	C	W
$\mathbf{f}_t^{i,c}$	i-th finger local force at $\mathcal{C}_c$	$\mathbb{R}^3$	C	c
$\mathbf{m}_t^{i,c}$	i-th finger local moment at $\mathcal{C}_c$	$\mathbb{R}^3$	C	c
$\alpha_t^{i,c}$	i-th finger contact at $\mathcal{C}_c$	$\mathbb{Z}^1$	B	
$\beta_t^{i,v,h}$	i-th finger collision to h-th face of $\mathcal{V}_v$	$\mathbb{Z}^1$	B	
$\gamma_t^{i,c}$	direction of z-element of $m_t^{i,c}$	$\mathbb{Z}^1$	B	

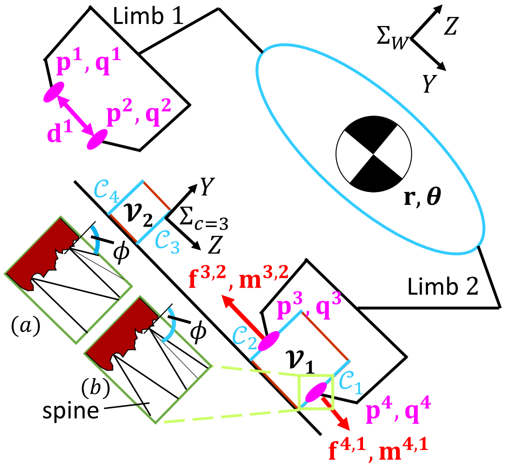
ADMM.

4.3.1 Preliminary

Here, we show our assumptions in our planner:

1. Each finger makes patch contact and follows two different frictional models discussed in Sec 4.2.
2. The environment consists of rigid static climbing holds whose geometry is modeled as cuboids.
3. Paired fingers align along the normal direction of fingertips. Paired fingers are in parallel and rotate only along z-axis in the world frame Σ_W.

We define the variables in Table 4.1 and Fig. 4.2. We also denote constants as follows.



Limb 1
Σ_W
Z
Y
$\mathbf{p}^1, \mathbf{q}^1$
$\mathbf{d}^1$
$\mathbf{p}^2, \mathbf{q}^2$
$\mathbf{r}, \boldsymbol{\theta}$
$\mathcal{C}_4$
$\mathcal{V}_2$
Y
$\Sigma_{c=3}$
$\mathcal{C}_3$
Z
ϕ
(a)
$\mathbf{f}^{3,2}, \mathbf{m}^{3,2}$
Limb 2
$\mathbf{p}^3, \mathbf{q}^3$
$\mathcal{C}_2$
ϕ
$\mathcal{V}_1$
$\mathcal{C}_1$
(b)
spine
$\mathbf{p}^4, \mathbf{q}^4$
$\mathbf{f}^{4,1}, \mathbf{m}^{4,1}$

Figure 4.2: Mathematical model of a multi-limbed robot. In this example, $n_f = 4, n_l = 2, C = 4, V = 2$. We also visualize two examples of frames $\Sigma_W, \Sigma_{c=3}$ where z-axis of Σ_W is perpendicular to the ground and z-axis of the local frame $\Sigma_{c=3}$ is perpendicular to the face of the climbing hold and along this axis we have non-zero z element of m. The spine makes contact with the contact angle (a): $\phi = \frac{\pi}{3}$ and (b): $\phi = \frac{\pi}{2}$.

N, n_f, n_l, C or V represent the time horizon, the total number of fingers, the total number of limbs, the total number of graspable regions, or the total number of obstacles, respectively. We denote $\mathcal{C}_c$ as the c-th graspable region, associated with a local frame Σ_c. Each obstacle $\mathcal{V}_v$ has n_v faces associated with a local face frame $\Sigma_{v_h}, h = 1, \ldots, n_v$. For any arbitrary vector $\mathbf{a}$, the notation $\|\mathbf{a}\|_A^2$ means a quadratic term with a positive-semi-definite matrix A. We define the coordinate transformation from frame Σ_A to Σ_B as ${}^A_B T$. We denote $X \implies Y$ as a conditional constraint and implement it using a big-M formulation.

4.3.2 Optimal Control Problem for Grasping and Locomotion

We propose the optimal control problem in (4.5). Our planner finds the optimal trajectory of body pose, limb poses, and wrenches, subject to limb and gripper kinematics, centroidal dynamics, bound of variables, gait, faces to grasp, collision-avoidance, and our proposed limit surface constraints.

$$\textbf{P1:} \quad \min_{\mathbf{x},\mathbf{u},\mathbf{y},\mathbf{z}} \sum_{t=0}^{N-1} J(\mathbf{x}_t, \mathbf{u}_t, \mathbf{y}_t, \mathbf{z}_t) \tag{4.5}$$

subject to:

1. For time-step $t = 0, \dots, N-1$
    - Bounds of decision variables (4.7).
    - Centroidal dynamics (4.8).
    - For fingers $i = 1, \dots, n_f$:
        - Kinematics (4.9).
        - For graspable regions $c = 1, \dots, C$:
            - Contact constraints (4.10).
            - Wrench transformation (4.13).
        - For obstacles $v = 1, \dots, V$:
            - Collision-avoidance (4.14).
2. Terminal state constraints.

We define $\mathbf{x}_t = [\mathbf{r}_t^\top, \boldsymbol{\theta}_t^\top, \dot{\mathbf{r}}_t^\top, \dot{\boldsymbol{\theta}}_t^\top, \mathbf{p}_t^{i\top}, \mathbf{q}_t^{i\top}, \mathbf{d}_t^{l\top}, \forall i, l]^\top$ as states, $\mathbf{u}_t = [\boldsymbol{\lambda}_t^{i\top}, \boldsymbol{\tau}_t^{i\top}, \forall i]^\top$ as control inputs, $\mathbf{y}_t = [\mathbf{f}_t^{i,c\top}, \mathbf{m}_t^{i,c\top}, \forall i, c]^\top$ as contact wrenches, and $\mathbf{z}_t = [\alpha_t^{i,c\top}, \beta_t^{i,v,h\top}, \gamma_t^{i,c\top}, \forall i, c, v, h]^\top$ as integer variables, where $i = 1, \dots, n_f$, $l = 1, \dots, n_l$, $c = 1, \dots, C$, $v = 1, \dots, V$, $h = 1, \dots, n_v$.

4.3.2.1 Cost Function and Bounds

Our cost function is:

$$J = \|\mathbf{x}_t - \mathbf{x}_g\|_Q^2 + \|\mathbf{u}_t\|_R^2 + \boldsymbol{\zeta}^\top \mathbf{x}_t + \sum_{l=1}^{n_l} \boldsymbol{\xi}_l^\top (\mathbf{p}_t^{2l} - \mathbf{p}_t^{2l-1}) + \|\mathbf{z}_{t+1} - \mathbf{z}_t\|_S^2 \tag{4.6}$$

The first term is the cost between the current state and the terminal state $\mathbf{x}_g$. The second term is the control effort cost. We aim to lift each limb as high as possible since potential hazards (e.g., obstacles) can exist near terrain. However, this capability has not been realized well. In [7, 13], the generated swing height is almost zero unless the authors give the reference trajectory. Hence, by assigning a negative value for elements of $\boldsymbol{\zeta} \in \mathbb{R}^{n_x}$ associated with limb heights in $\mathbf{x}_t$, our planner can swing limbs with reasonable heights.

We observe that the distance between the surface of the climbing hold and each finger when the robot release the fingers needs to be long enough. Otherwise, due to an imperfect position controller, the finger can stick to the climbing hold, resulting in the failure of releasing fingers. Hence, we maximize the distance between paired fingers with a negative value for elements of each $\boldsymbol{\xi}_l \in \mathbb{R}^3$, which indirectly increases the distance between the graspable region and the finger once the fingers release.

We observe that CITO randomly switches the discrete modes (e.g., contact on-off), which could lead to instability. By assigning a quadratic term for $\mathbf{z}_t, \mathbf{z}_{t+1}$ associated with the mode we do not want to switch frequently, the fifth term in (4.6) prevents mode changes between t and $t+1$.

We bound the range of desicion varibles as follows:

$$\mathbf{x}_t \in \mathcal{X}, \mathbf{u}_t \in \mathcal{U}, \mathbf{y}_t \in \mathcal{Y}, \mathbf{z}_t \in \mathcal{Z} \tag{4.7}$$

where $\mathcal{X} \subseteq \mathbb{R}^{n_x}, \mathcal{U} \subseteq \mathbb{R}^{n_u}$, and $\mathcal{Y} \subseteq \mathbb{R}^{n_y}$ are convex polytopes consisting of a finite number of linear inequality constraints. $\mathcal{Z} \subseteq \{0,1\}^{n_z}$ shows range of binary variables.

4.3.2.2 Centroidal Dynamics

The dynamics is given by:

$$M\ddot{\mathbf{r}}_t = \sum_{i=1}^{n_f} \boldsymbol{\lambda}_t^i + M\mathbf{g} \tag{4.8a}$$

$$I\dot{\boldsymbol{\omega}}_t + \boldsymbol{\omega}_t \times I\boldsymbol{\omega}_t = \sum_{i=1}^{n_f} \left(\mathbf{r}_t - \mathbf{p}_t^i\right) \times \boldsymbol{\lambda}_t^i + \boldsymbol{\tau}_t^i \tag{4.8b}$$

where M, I represent the mass and inertia of the robot. $\mathbf{g} \in \mathbb{R}^3$ is the gravity acceleration, and $\boldsymbol{\omega}_t$ is the angular velocity from $\boldsymbol{\theta}_t$ [14]. This work explicitly considers $\boldsymbol{\tau}_t^i$ to capture the effect of patch contacts. For implementation, we use the explicit-Euler method with time interval dt.

4.3.2.3 Kinematics

Our kinematics constraints are as follows:

$$|R(\boldsymbol{\theta}_t)(\mathbf{p}_t^i - \mathbf{r}_t) - \mathbf{a}^i| \leq \mathbf{b}^i, |\mathbf{q}_t^i - \mathbf{c}^i - \boldsymbol{\theta}_t| \leq \mathbf{d}^i \tag{4.9a}$$

$$\mathbf{p}_t^{2l} = \mathbf{d}_t^l + \mathbf{p}_t^{2l-1}, \mathbf{q}_t^{2l} = \mathbf{q}_t^{2l-1} \tag{4.9b}$$

$R(\boldsymbol{\theta}_t)$ is the rotation matrix from Σ_W to Σ_B where Σ_B is the body frame. $\mathbf{a}^i, \mathbf{b}^i$ are the nominal position and acceptable range from the nominal position of i-th finger in Σ_B. $\mathbf{c}^i, \mathbf{d}^i$ represent the nominal orientation and acceptable range from the nominal orientation of i-th finger. In (4.9b), one of the paired finger positions is determined by another finger position and $\mathbf{d}_t^l$. Since fingers on the same gripper are parallel, we set the orientation of those paired fingers as same. Later, we use (4.9) in MIQP and thus conservatively approximate (4.9) by linearizing $R(\boldsymbol{\theta}_t)$ at a certain angle.

4.3.2.4 Contact Constraints

Contact dynamics is inherently discrete phenomenon and thus it can be given by:

$$\alpha_t^{i,c} = 1 \Longrightarrow \begin{Bmatrix} \mathbf{f}_t^{i,c}, \mathbf{m}_t^{i,c} \in \mathcal{W}\left(\mu_c, k_c\right), \\ \dot{\mathbf{p}}_t^i, \dot{\mathbf{q}}_t^i = \mathbf{0}, {}_W^cT\left(\mathbf{p}_t^i, \mathbf{q}_t^i\right) \in \mathcal{C}_c \end{Bmatrix} \tag{4.10a}$$

$$\alpha_t^{i,c} = 0 \Longrightarrow \mathbf{f}_t^{i,c}, \mathbf{m}_t^{i,c} = \mathbf{0} \tag{4.10b}$$

$$\sum_{c=1}^{C} \alpha_t^{i,c} \leq 1 \tag{4.10c}$$

μ_c, k_c are parameters from (4.2) defined in $\mathcal{C}_c$. The constraints in (4.10a) mean that if the finger makes contact on $\mathcal{C}_c$, the local wrench $\mathbf{f}_t^{i,c}, \mathbf{m}_t^{i,c}$ needs to follow the patch constraints in (4.1) and the finger does not move. ${}_W^cT\left(\mathbf{p}_t^i, \mathbf{q}_t^i\right)$ represents the $\mathbf{p}_t^i, \mathbf{q}_t^i$ in Σ_c. If the finger is in the air, (4.10b) means that the local wrench is zero. Because the finger can only make contact on one of the graspable regions, (4.10c) does not allow the finger to make more than one contact.

Later, we use (4.10) in MIQP and here we approximate (4.10) as linear inequality constraints. In particular, the only constraints which need to be approximated are (4.2) inside

(4.10a). For notation simplicity, we use f^x, f^y, f^z as x, y, z elements of $\mathbf{f}_t^{i,c}$, respectively and m^z as a z element of $\mathbf{m}_t^{i,c}$.

$$|f^i| \leq \mu_c f^z - \frac{|m^z|}{k}, i = \{x, y\}, |m^z| \leq k\mu_c f^z \tag{4.11}$$

The issue in (4.11) is that we need to consider two absolute value of decision variables $|f^i|, |m^z|$ simultaneously. We employ a piece-wise linear representation with integer variables to deal with (4.11) as follows:

$$\gamma_t^{i,c} = 1 \Longrightarrow m_-^z = 0, \gamma_t^{i,c} = 0 \Longrightarrow m_+^z = 0 \tag{4.12a}$$

$$|f^i| \leq \mu_c f^z - \frac{m_+^z}{k}, |f^i| \leq \mu_c f^z - \frac{m_-^z}{k}, i = \{x, y\}, \tag{4.12b}$$

$$m^z = m_+^z - m_-^z, m_+^z \geq 0, m_-^z \geq 0, |m^z| \leq k\mu_c f^z \tag{4.12c}$$

where m_+^z, m_-^z are non-negative values and are the moment along z-axis in Σ_c in the positive and negative direction. Using (4.12a), we decompose (4.11) into two inequality constraints in the positive and negative direction of m^z.

4.3.2.5 Wrench Transformation

The wrench in Σ_W can be obtained from local wrenches:

$$\boldsymbol{\lambda}_t^i = \sum_{c=1}^{C} {}^W_c T \mathbf{f}_t^{i,c}, \ \boldsymbol{\tau}_t^i = \sum_{c=1}^{C} {}^W_c T \mathbf{m}_t^{i,c}, \tag{4.13}$$

With constraints (4.10), (4.13) converts a specific local wrench where the finger makes contact to the wrenches in Σ_W.

4.3.2.6 Collision-Avoidance

The constraints in (4.10) could allow the fingers to penetrate into the holds. To avoid the penetration, the collision-avoidance constraints are given by:

$$\beta_t^{i,v,h} = 0 \Longrightarrow {}^{v_h}_W T \mathbf{p}_t^i \cdot \mathbf{n}^{v_h} \leq s^{v_h}, \forall h = 1, \ldots, n_v \tag{4.14a}$$

$$\sum_{h=1}^{n_v} \beta_t^{i,v,h} \leq n_v - 1 \tag{4.14b}$$

where $\mathbf{n}^{v_h}$ is the normal vector to face h of obstacle v in Σ_{v_h} and s^{v_h} is a scalar to decide the location of the plane. ${}^{v_h}_{W}T\mathbf{p}^i_t$ represents $\mathbf{p}^i_t$ in Σ_{v_h}. (4.14) means that the finger needs to be outside at least one face of the obstacle.

4.3.3 Alternating Direction Method of Multipliers (ADMM)

ADMM solves the optimization problem with consensus constraints as follows:

$$\min_{\boldsymbol{\eta},\boldsymbol{\delta}} f(\boldsymbol{\eta}) + g(\boldsymbol{\delta}), \text{ s. t. } A\boldsymbol{\eta} + B\boldsymbol{\delta} = \mathbf{c} \tag{4.15}$$

where $\boldsymbol{\eta} \in \mathbb{R}^{n_\eta}, \boldsymbol{\delta} \in \mathbb{R}^{n_\delta}, A \in \mathbb{R}^{n_\epsilon \times n_\eta}, B \in \mathbb{R}^{n_\epsilon \times n_\delta}, \mathbf{c} \in \mathbb{R}^{n_\epsilon}$. By decomposing the original problem with two smaller-scale problems and solving each problem with considering consensus, ADMM effectively solves the original optimization problem with faster convergence [29].

The augmented Lagrangian of (4.15) can be given by:

$$L_\rho(\boldsymbol{\eta}, \boldsymbol{\delta}, \boldsymbol{\epsilon}) = f(\boldsymbol{\eta}) + g(\boldsymbol{\delta}) + \frac{\rho}{2}(\|A\boldsymbol{\eta} + B\boldsymbol{\delta} - \mathbf{c} + \boldsymbol{\epsilon}\|_2^2) \tag{4.16}$$

with $\rho > 0$. $\boldsymbol{\epsilon} \in \mathbb{R}^{n_\epsilon}$ is the dual variable associated with the constraints $A\boldsymbol{\eta} + B\boldsymbol{\delta} = \mathbf{c}$. Then, ADMM finds the solution by taking the following steps recursively:

$$\boldsymbol{\eta}^{k+1} := \operatorname*{argmin}_{\boldsymbol{\eta}} L_\rho\left(\boldsymbol{\eta}, \boldsymbol{\delta}^k, \boldsymbol{\epsilon}^k\right) \tag{4.17a}$$

$$\boldsymbol{\delta}^{k+1} := \operatorname*{argmin}_{\boldsymbol{\delta}} L_\rho\left(\boldsymbol{\eta}^{k+1}, \boldsymbol{\delta}, \boldsymbol{\epsilon}^k\right) \tag{4.17b}$$

$$\boldsymbol{\epsilon}^{k+1} := \boldsymbol{\epsilon}^k + A\boldsymbol{\eta}^{k+1} + B\boldsymbol{\delta}^{k+1} - \mathbf{c} \tag{4.17c}$$

The natural extension of the two-block ADMM is I-block ADMM:

$$\min_{{}^1\boldsymbol{\eta},{}^2\boldsymbol{\eta},\dots,{}^I\boldsymbol{\eta}} \sum_{i=1}^{I} f_i\left({}^i\boldsymbol{\eta}\right) \text{ s. t. } {}^i_j\eta = {}^{\mathcal{G}(i,j)}\delta, \forall (i,j) \in \mathcal{G} \tag{4.18}$$

where ${}^i_j\eta$ is the j-th local decision variable of i-th block optimization problem. ${}^g\delta = {}^{\mathcal{G}(i,j)}\delta$ is the h-th element of the global decision variable, $\boldsymbol{\delta} \in \mathbb{R}^G$. $\mathcal{G}$ is a bipartite graph formed from global decision variables and local decision variables where each edge represents a consensus

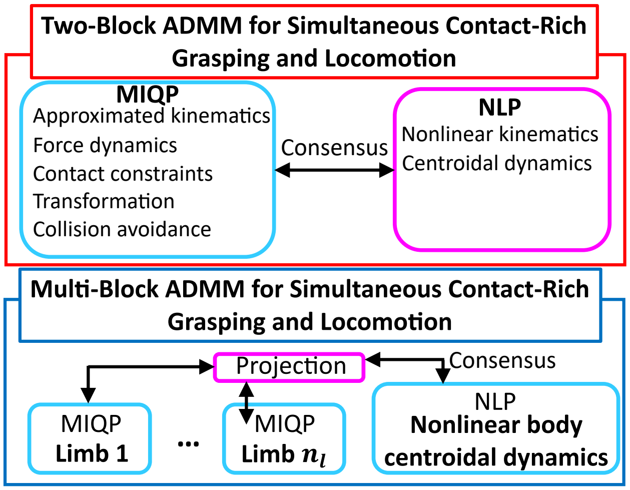

Two-Block ADMM for Simultaneous Contact-Rich Grasping and Locomotion
MIQP
Approximated kinematics
Force dynamics
Contact constraints
Transformation
Collision avoidance
Consensus
NLP
Nonlinear kinematics
Centroidal dynamics
Multi-Block ADMM for Simultaneous Contact-Rich Grasping and Locomotion
Projection
Consensus
MIQP Limb 1
...
MIQP Limb n_l
NLP Nonlinear body centroidal dynamics

Figure 4.3: We propose two decomposition-based optimization based on ADMM specific for motion planning of limbed robots. (top): Two-block ADMM where MIQP considers discrete constraints and NLP considers nonlinear constraints so that the planner effectively solves the MINLP once these two optimization problems achieve consensus. (bottom): Multi-block ADMM where the planner consists of n_l MIQP problems and one NLP.

constraint (see [29]). We also denote $\mathcal{N}(g) = \{i, j | (i, j) \in \mathcal{G}\}$ as the set of all local variables connected to ${}^{g}\delta$. Denote ${}^{i}_{j}\epsilon$ as the dual variable associated with the consensus constraints. The multi-block consensus problem can be solved as follows:

$$
\begin{aligned}
{}^{i}\boldsymbol{\eta}^{k+1} &:= \underset{{}^{i}\boldsymbol{\eta}}{\operatorname{argmin}}\, f_i({}^{i}\boldsymbol{\eta}) + \sum_{g \in \mathcal{G}(i,j)} \frac{\rho_i}{2} \| {}^{i}_{j}\eta - {}^{g}\delta^{k} + {}^{i}_{j}\epsilon^{k} \|_2^2, && \text{(4.19a)} \\
{}^{g}\delta^{k+1} &:= \underset{{}^{g}\delta}{\operatorname{argmin}} \sum_{i,j \in \mathcal{N}(g)} \frac{\rho_i}{2} \| {}^{i}_{j}\eta^{k+1} - {}^{g}\delta + {}^{i}_{j}\epsilon^{k} \|_2^2, && \text{(4.19b)} \\
{}^{i}_{j}\epsilon^{k+1} &:= {}^{i}_{j}\epsilon^{k} + ({}^{i}_{j}\eta^{k+1} - {}^{g}\delta^{k+1}), \forall (i, j) \in \mathcal{G} && \text{(4.19c)}
\end{aligned}
$$

4.3.4 Decomposition-based Optimal Control Problem

We propose the following decomposition-based optimization framework so that the original intractable MINLP problem (P1) becomes tractable. The key idea is that we use an NLP solver to solve NLP with nonlinear continuous constraints and a MIP solver to solve MIQP with discrete constraints so that we can employ the strength of each solver. Our proposed decomposition-based optimization from P1 solves the following optimization problems recursively (**P2**):

$$\mathbf{P2a} : \min_{\boldsymbol{\eta},y,z} \sum_{t=0}^{N-1} J(\boldsymbol{\eta}_t^k) + \frac{\rho}{2}\|\boldsymbol{\eta}_t^k - \boldsymbol{\delta}_t^k + \boldsymbol{\epsilon}_t^k\|_2^2 \tag{4.20a}$$

$$\begin{aligned} &\text{s.t. } \{\forall t, (4.7), (4.8a)\} \bigcap \{\forall t, i, (4.9)\} \\ &\bigcap \{\forall t, i, c, (4.10), (4.13)\} \bigcap \{\forall t, i, v, (4.14)\} \end{aligned} \tag{4.20b}$$

$$\mathbf{P2b} : \min_{\boldsymbol{\delta}} \sum_{t=0}^{N-1} \frac{\rho}{2}\|\boldsymbol{\eta}_t^{k+1} - \boldsymbol{\delta}_t^k + \boldsymbol{\epsilon}_t^k\|_2^2 \tag{4.20c}$$

$$\text{s.t. } \{\forall t, (4.7), (4.8)\} \bigcap \{\forall t, i, (4.9)\} \tag{4.20d}$$

$$\mathbf{P2c} : \boldsymbol{\epsilon}_t^{k+1} \longleftarrow \boldsymbol{\epsilon}_t^k + \boldsymbol{\eta}_t^{k+1} - \boldsymbol{\delta}_t^{k+1} \tag{4.20e}$$

where $\boldsymbol{\eta}_t^k = [x_k^\top, u_k^\top]^\top, \boldsymbol{\eta}^k = [\boldsymbol{\eta}_0^{k\top}, \ldots, \boldsymbol{\eta}_N^{k\top}]^\top$. The indexes in (4.20) are $t = 0, \ldots, N-1$, $i = 1, \ldots, n_f$, $c = 0, \ldots, C$, $v = 1, \ldots, V$. We create copies of variables $\boldsymbol{\eta}_t$ as $\boldsymbol{\delta}_t$ and $\boldsymbol{\delta}^k = [\boldsymbol{\delta}_0^{k\top}, \ldots, \boldsymbol{\delta}_N^{k\top}]^\top$. P2a is MIQP. P2b is continuous NLP. Since both MIQP and NLP can be efficiently solved using off-the-shelf solvers, our proposed method would converge earlier than the method solving P1 using MINLP solvers. Also, our method considers all constraints from grasping, locomotion, and contact once it achieves the consensus between MIQP and NLP. Thus, it does not suffer from the infeasible issue of hierarchical planning explained in Sec 2.1.

Remark 1: In P2, we do not consider the consensus of $\mathbf{y}, \mathbf{z}$. For $\mathbf{y}$, $\mathbf{u}$ indirectly has an effect on $\mathbf{y}$ via (4.13) so we do not explicitly enforce consensus constraints for $\mathbf{y}$, which enables more efficient ADMM computation. For $\mathbf{z}$, because it is quite difficult for NLP to satisfy discrete constraints, we do not enforce consensus between P2a and P2b.

4.3.5 Multi-Block Decomposition-based Optimal Control Problem

We propose another option to solve the MINLP which does not involve many difficult constraints (e.g., discrete constraints) based on multi-block ADMM as follows (**P3**):

$$\textbf{P3a}: \min_{{}^{i}\boldsymbol{\eta},y,z} \sum_{t=0}^{N-1} \left({}^{i}J({}^{i}\boldsymbol{\eta}_t^k) + \sum_{g \in \mathcal{G}(i,j)} \frac{\rho_i}{2} \| {}^{i}_{j}\eta_t - {}^{g}\delta_t^k + {}^{i}_{j}\epsilon_t^k \|_2^2 \right), \tag{4.21a}$$

$$\text{s.t. for MIQP: P2a, for NLP: P2b,} \tag{4.21b}$$

$$\textbf{P3b (projection)}: \min_{{}^{g}\delta} \sum_{t=0}^{N-1} \sum_{i,j \in \mathcal{N}(g)} \frac{\rho_i}{2} \| {}^{i}_{j}\eta_t^{k+1} - {}^{g}\delta_t + {}^{i}_{j}\epsilon_t^k \|_2^2 \tag{4.21c}$$

$$\textbf{P3c}: {}^{i}_{j}\epsilon^{k+1} := {}^{i}_{j}\epsilon_t^k + ({}^{i}_{j}\eta_t^{k+1} - {}^{g}\delta_t^{k+1}) \tag{4.21d}$$

where ${}^{i}\boldsymbol{\eta}_t^k = [{}^{i}x_k^\top, {}^{i}u_k^\top]^\top$. We run P3a with constraints from P2a for each limb to solve discrete constraints, resulting in total n_l MIQP problems in parallel. We run one P3a with constraints from P2b to solve nonlinear constraints. Then, we run P3b as projection (see Fig. 4.3).

4.4 Experimental Results

In this section, we validate our proposed methods for two tasks: walking and free-climbing. Through the experiments, we try to answer the following questions:

1. Can our proposed optimization generate open-loop trajectories efficiently?
2. Can our proposed formulation consider patch contacts with micro-spines explicitly?
3. How do the generated trajectories behave in a real four-limbed robot?

4.4.1 Implementation Details

For optimization settings, we implemented our method using Gurobi [94] for solving MIQP and ipopt [10] with PYROBOCOP [16] for solving NLP. The optimizations are done on a computer with the Intel i7-8565U. The trajectories discussed in this section are from two-block ADMM and we use the solution from MIQP (P2a).

We implemented the results of our proposed methods on a real four-limbed robot [108, 109] equipped with two-finger grippers [110]. The grippers make patch contact with microspines, which are mechanically constrained such that the patch is always perpendicular to the surface during contact. This satisfies one of our assumptions in Sec 4.2.2. Each limb consists of 7 DoF, where 6 DoF are to actuate the joints of the limb and 1 DoF is to actuate the gripper. The robot weighs 9.6 kg. The admittance control was used to track the reference wrenches from the planner [111]. Free-climbing experiments were conducted on a rugged wall with gradient 45°. Hardware experiments can be viewed in the accompanying video.

Remark 2: We set dt to the large value and have tight bounds on $\dot{\mathbf{p}}_t^i$ for free-climbing, resulting in slow motions in hardware. This is because our robot uses linear actuators to actuate fingers, whose internal motor velocity is slow.

4.4.2 Computation Results

4.4.2.1 Convergence Analysis

We discuss the convergence of our ADMMs. For walking with point contacts on a flat plane without, we set $dt = 0.08$, $N = 40$, $\rho = 1.5$, $\mu = 0.6$. For walking, we consider point contacts (i.e., no fingers). Since we consider a single flat plane walking without obstacles, we set $C = 1, V = 0$. For climbing, we set $dt = 2.0$, $N = 40$, $\rho = 10$, $\mu = 2.2$, $r_p = 0.02$ m. This scenario considers four climbing holds which consists of five faces so we set $C = 20$, $V = 4$. Both cases run ADMM for 10 iterations.

We show the evolution of residual for walking and free-climbing using our two- and five-block ADMM in Fig. 4.4 and Fig. 4.5. Overall, we observe that our proposed ADMM converges with small enough norms of residual. We also observe that for free-climbing, our two-block ADMM shows faster convergence than our five-block ADMM and for walking, both two- and five-block ADMM shows similar convergence. This is because the problem is so

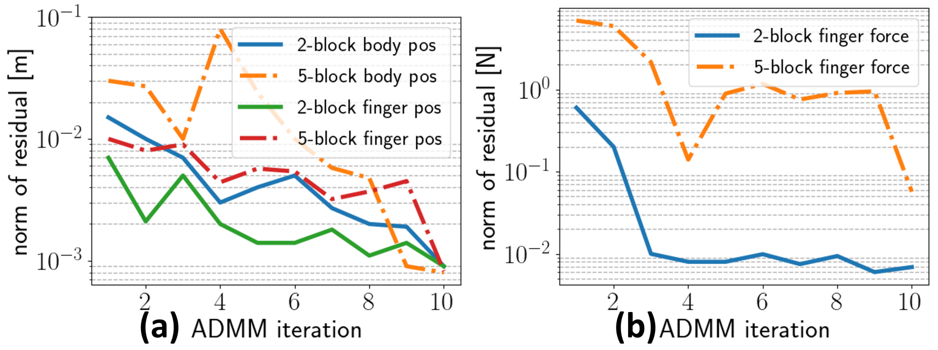


Figure 4.4: Evolution of residuals for walking using two- and five-block ADMM. (a): Residual of body and finger positions, (b): residual of finger forces.

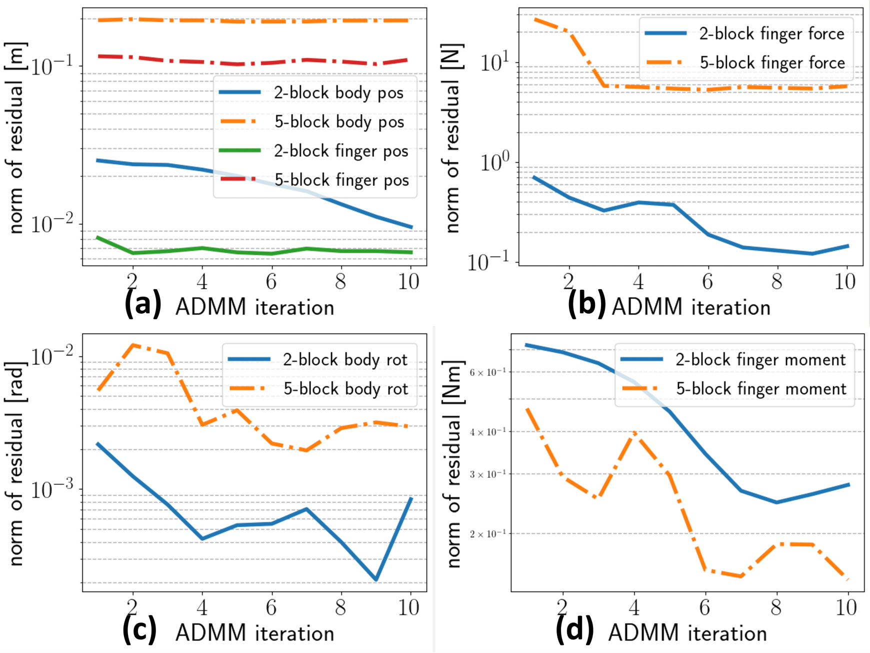


Figure 4.5: Evolution of residuals for free-climbing using two- and five-block ADMM. We show residual of (a): body and finger positions, (b): finger forces, (c): body rotation, (d): finger moments.

complicated that it is quite difficult for five-block ADMM to achieve consensus, resulting in higher norms of residual. In contrast, our two-block ADMM only needs to achieve consensus for two optimization problems, resulting in lower norms of residuals.

We also discuss the generated trajectories for free-climbing by our two-block ADMM as shown in Fig. 4.6. The trot gait trajectory is the result of our ADMM after 1 iteration. This trot gait is physically infeasible (i.e., tumbling) since MIQP is not aware of nonlinear moment constraints (4.8b). After 6 iterations of our ADMM, our planner generates a physically feasible one leg gait. Because ADMM converges, MIQP is now aware of (4.8b) so that the generated trajectory does not make the robot tumble anymore, resulting in physically feasible trajectories. We do not observe this mode change for the walking task since the walking task is naturally more stable than the free-climbing task so our ADMM does not face the need for mode change.

4.4.2.2 Computation Time, Success Rate

We compare our ADMM with the benchmark optimization using NLP with respect to the computation time and the success rate of finding a feasible solution for walking and climbing problems. As a benchmark, we solve P1 as NLP using a technique in [1]. We sample ten feasible initial and terminal states and calculate the average computation time and the success rate where the optimization could find a solution. We use the same parameters in Sec 4.4.2.1. Table 4.2 shows that our ADMMs show smaller computation time and a higher success rate against the benchmark method. For walking, our five-block ADMM shows smaller computation time and for free-climbing, our two-block ADMM shows smaller computation time. In other words, if the problem is not so complicated (e.g., walking), our five-block ADMM can be an option to solve the problem even though it takes more iteration to converge. This is because our five-block ADMM spends less time for each iteration so that the total computation time can be small.

Our proposed ADMM does not employ a warm-start. This is because each block on our ADMM solves a smaller-scale optimization problem, which is less sensitive to initial

Table 4.2: Comparison of computation time and success rate for walking and climbing problems between benchmark optimization based on P1 using [1], our two-block ADMM, and our multi-block ADMM. For ADMMs, the computation time is calculated as the total computation time until the norm of residual for body and foot positions converge to 0.03 m and the norm of residual for reaction forces converge to within 0.5 N. For computation time, we show the mean time and 99 % confidence interval.

Walking	Computation time [s]	Success rate [%]
Benchmark	3275	10
Our two-block ADMM	168 ± 50	100
Our five-block ADMM	44 ± 21	100
Climbing		
Benchmark	N/A	0
Our two-block ADMM	619 ± 99	100
Our five-block ADMM	970 ± 60	90

guesses compared to larger-scale complicated optimization problems (e.g., MINLP). We did not observe a significant reduction of computation time with our warm-start, but designing warm-start for consensus constraints would be promising.

4.4.3 Results of Our Generated Trajectories

4.4.3.1 Collision-Avoidance

This scenario focuses on environments with obstacles. We set $dt = 2.0$, $N = 120$, $\rho = 15$, $\mu = 2.2$, $r_p = 0.02$ m. We consider 16 climbing holds and 3 obstacles and run our two-block ADMM for 3 iterations.

The generated trajectory with snapshots of the hardware experiment is shown in Fig. 4.1. Our ADMM successfully generates collision-free trajectories under a number of discrete constraints. Here we discuss three points in Fig. 4.1. Around point A, since the height of the obstacle is not so high, our ADMM generates the trajectory that gets over the obstacle. Around point B, since the height of the obstacle is high, the robot cannot get over the ob-

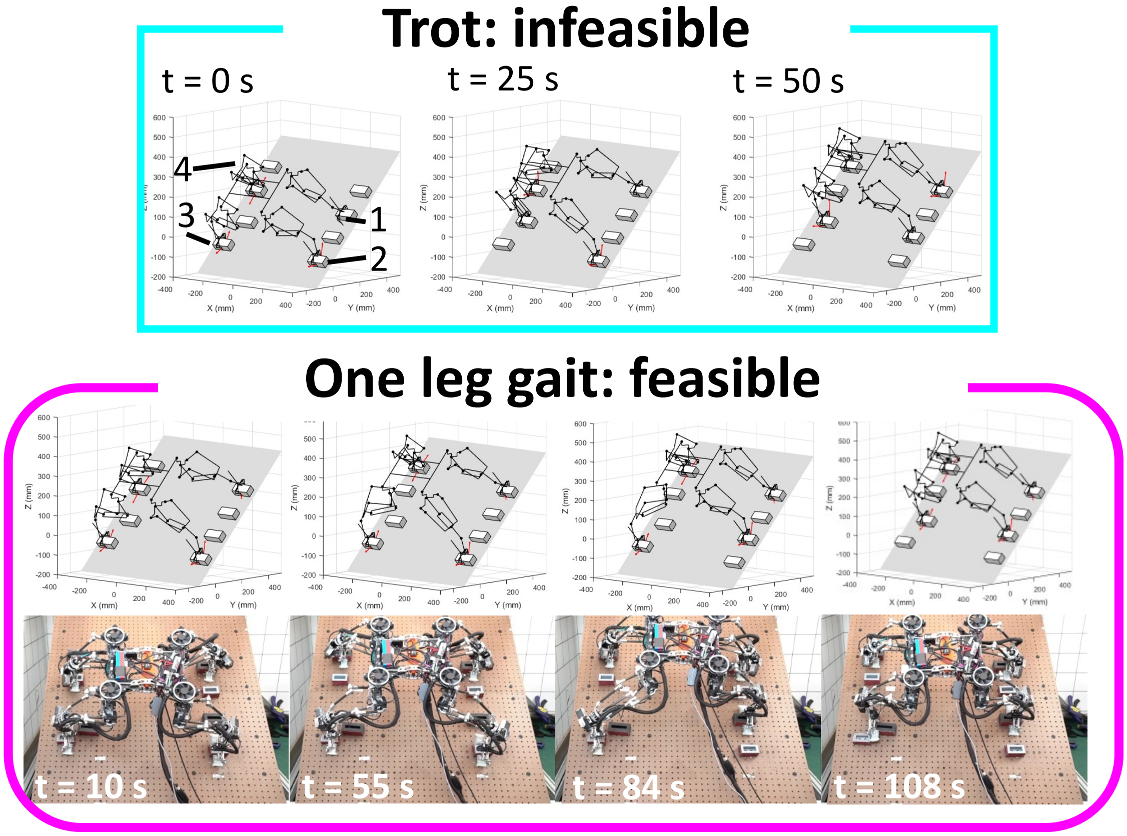


Figure 4.6: Change of modes as our ADMM proceeds with snapshots of hardware experiments. (top): after 1 iteration of our ADMM, the planner generates a trot gait, which is physically infeasible since MIQP is not fully influenced by nonlinear constraints yet. (bottom): after 6 iterations, the planner finds a physically feasible one leg gait sequence (1 → 4 → 2 → 3).

stacle. Thus, our ADMM generates the trajectory that takes a detour around the obstacle. Around point D, limb 4 could directly make contact on D from C, but due to the obstacle, the robot first makes a contact on E and then goes to D.

4.4.3.2 Slippery Rotated Holds

This scenario shows that our ADMM designs trajectories with varying coefficients of friction on rotated holds. We set $dt = 2.0$, $N = 40$, $\rho = 15$, $r_p = 0.02$ m. The environment consists of 8 climbing holds with different orientations along z-axis in Σ_W. In Fig. 4.7, each climbing hold has a μ where the front and back faces are covered by 36-sand papers ($\mu = 2.2$), and the left and right faces are covered by the material with $\mu = 0$.

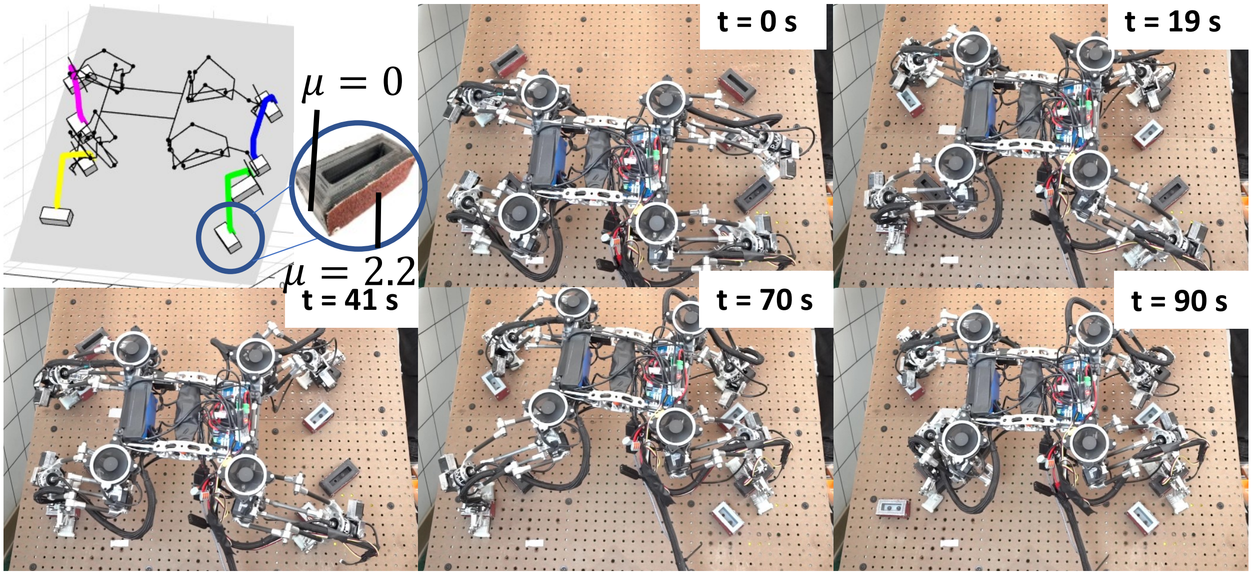


Figure 4.7: Our planned trajectories on holds with varying coefficients of frictions. The robot grasps the faces whose coefficients of friction are high.

The generated trajectory with snapshots of experiments is shown in Fig. 4.7, where the robot grasps the front or back face (high friction). To grasp the front or back face, the robot rotates the grippers so that the fingers make contacts perpendicular to the faces. In short, our planner could find feasible trajectories subject to the pose, wrenches, and contacts together. In contrast, hierarchical planners may only find the infeasible solution since it cannot consider coupling constraints in general.

4.4.4 Contact Modeling Results

4.4.4.1 Results of Micro-Spine Limit Surface

We force one of the fingers to have zero normal forces during contact (i.e., z element of $\mathbf{f}_t^{i,c}$ is set to zero for all $c = 1, \ldots, C$). Since (4.3) enables the planner to generate non-zero shear forces even under zero normal forces, we expected that our planner can still find feasible solutions for free-climbing. The result is illustrated in Fig. 4.8. Our planner is able to generate non-zero shear force trajectories under zero normal forces.

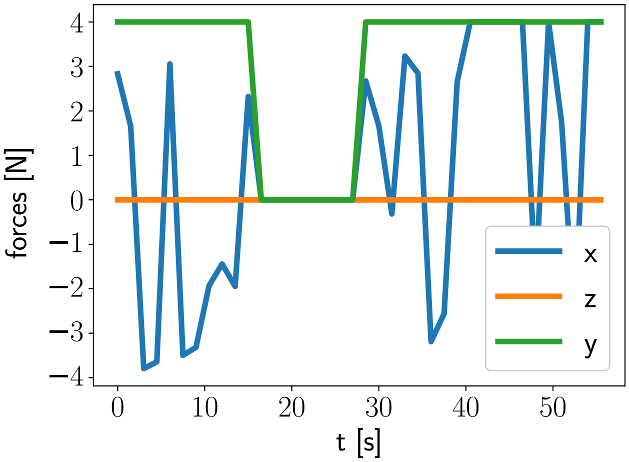


Figure 4.8: Time history of reaction forces for one finger. Here, we enforce z element of reaction force in Σ_{C_c} is always zero with $f^i_{\max} = 4$ N in (4.3).

Table 4.3: Comparison of the number of slipping between a generated trajectory with patch constraints (Traj A) and without patch constraints (Traj B) over 5 samples for each case.

	$m_z = 0.0$ Nm	$m_z = 0.3$ Nm	$m_z = 0.6$ Nm
Traj A	1 / 5	0 / 5	0 / 5
Traj B	2 / 5	5 / 5	5 / 5

4.4.4.2 Results of Frictional Limit Surface

We investigate if our proposed planner can generate physically feasible trajectories under patch constraints (4.1) in hardware. During free-climbing, the loading shear forces and moments exist at the tip of the finger, which can lead to instability of the contact state. We hope that considering (4.12) counteracts these loading shear forces and moments so that the contact state is stable. Thus, our planner creates two different trajectories, one considering patch constraints (Traj A) and the other one not considering them (Traj B). To simplify the analysis, for both cases, the planner set the same constant shear force and the moment (f_x, m_z in (4.11)) during contacts. We tested on the box covered by 36-grit sandpapers with specified $f_x = 9.0$ N and $m_z = \{0, 0.3, 0.6\}$ Nm.

Table 4.3 shows the empirically obtained number of slipping for Traj A is much smaller

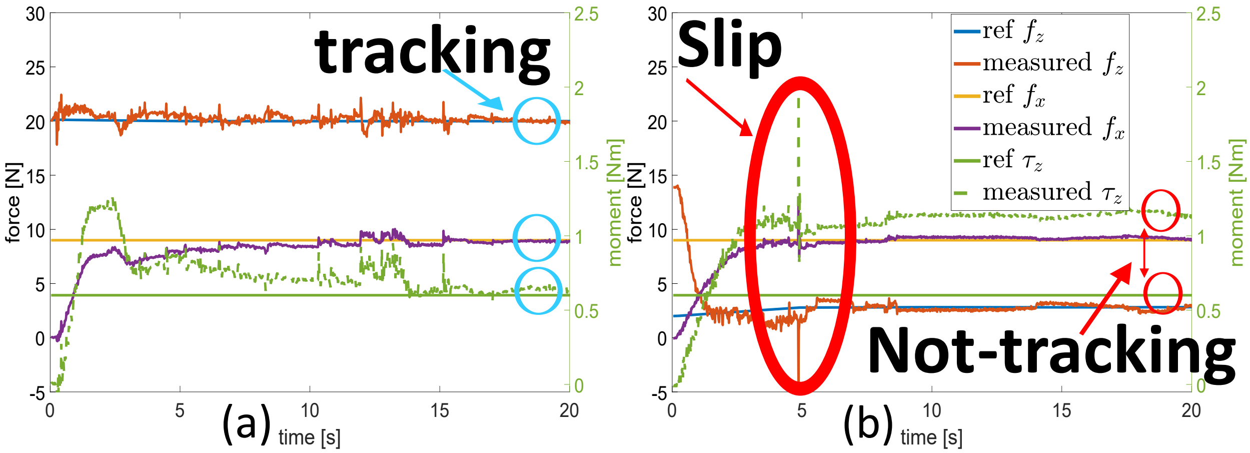


Figure 4.9: Time history of wrench trajectories between the trajectory (a): considering patch constraints and (b): not considering patch constraints.

than that for Traj B. This is because Traj A generates higher normal forces to avoid slipping because of patch constraints. We also show the time history of Traj A and B in Fig. 4.9 given $m_z = 0.6$ Nm, $f_x = 9.0$ N settings. In Fig. 4.9 (a), since our ADMM generates larger normal forces because of (4.11), the finger does not slip and the admittance controller could track all wrenches. In contrast, in Fig. 4.9 (b), since our ADMM generates smaller normal forces, the finger slips, and the controller could not track the wrenches. Therefore, we successfully verify in hardware that considering (4.12) helps avoid slipping for patch contacts, resulting in a more stable contact state.

4.5 Discussion

This chapter presents a model-based motion planning algorithm based on decomposition-based optimization for solving nonlinear contact-rich systems efficiently. We first propose the complete optimization formulation and extend it to our proposed decomposition-based formulations. We also discuss the limit surface of two-finger grippers with patch contacts and micro-spines. We verify the efficiency of our proposed formulation and demonstrate the generated trajectories in hardware experiments.

One limitation of our ADMM is that the computation is demanding once the number of discrete constraints increases. In order to use our framework in MPC, we need to run it with a much faster runtime. One promising direction is to design heuristics online [112]. We hope that we can accelerate our framework as ADMM iteration proceeds based on previous solutions. Another limitation during hardware experiments is that it is important to use accurate physical parameters. Otherwise, the robot may not be able to execute the planned trajectory. With this motivation, in the next chapters, we present robust planning algorithms under uncertain parameters.

CHAPTER 5

Risk-Aware Motion Planning for Multi-Limbed Robots

In the previous chapter, we present our planning through contact. However, it does not consider uncertain parameters such as friction coefficients, which are major sources of uncertainty. Starting in this chapter, we present our contributions for robust planning through contact. In this chapter, we present a motion planning algorithm with probabilistic guarantees for limbed robots with stochastic gripping forces because of stochastic contact dynamics between the robot and the environment. This chapter does not discuss the change of contact modes but focuses on the chance-constrained optimization formulation for motion planning of multi-limbed robots. We describe why stochastic optimization is useful, why it is challenging to get stochastic contact model, how we formulate chance-constrained optimization, how we get stochastic contact model using machine learning, and how well the proposed algorithm works in hardware experiments.

This chapter has been partially adapted from one journal paper:

- **Y. Shirai**, X. Lin, Y. Tanaka, A. Mehta, and D. Hong, "Risk-Aware Motion Planning for a Limbed Robot with Stochastic Gripping Forces Using Nonlinear Programming", *IEEE Robot. Auto. Lett.*, vol. 5, no. 4, pp. 4994-5001, 2020.

5.1 Overview

Planning complex motions for limbed robots is challenging because planners need to design footsteps and body trajectories while considering the robot kinematics and reaction forces. Motion planning for limbed robots has been studied by a number of researchers. Sampling-based planning, such as the Probabilistic-Roadmap (PRM), samples the environment and generates the motion while satisfying static equilibrium and kinematics for a robot [113], [93]. Optimization-based planning, such as MIQP and NLP, solves the solution given constraints using optimization algorithms such as gradient descent [75, 9].

While many papers discuss motion planning for the robot, few studies have investigated how planning is affected by stochastic gripping forces. One of the open problems in motion planning of a limbed robot equipped with grippers is the stochastic nature of gripping [107]. For example, the gripping forces caused by spine grippers depend on the stochastically distributed asperity strength (Fig. 5.2). Thus, *risk* results from the randomness of the gripping force. By considering risk in a probabilistic manner, the planner can design a variety of trajectories based on various specifications.

In this chapter, we address a motion planning algorithm formulated as NLP for a limbed robot with stochastic gripping forces. Our proposed planner solves for stable postures and forces simultaneously with guaranteed bounded risk. In addition, chance constraints are introduced into the planner that restrict contact forces in a probabilistic manner. We employ a Gaussian Process (GP), a non-parametric Bayesian regression tool, to acquire the PDF of the gripping force. Our proposed motion planning algorithm is validated on an 11.5 kg hexapod robot with spine grippers for multi-surface climbing. While we focus on multi-surface robotic climbing with spine grippers in this chapter, our proposed planner can be applied to other robots with any type of grippers for performing any task (e.g., planning of walking, grasping) as long as the robot has contact points with stochastic models.

The contributions of this paper are as follows:

1. We formulate risk-bounded NLP-based planning that considers the stochasticity of gripping forces.
2. We employ the Gaussian Process to model gripping forces as random variables.
3. We validate the algorithm in hardware experiments.

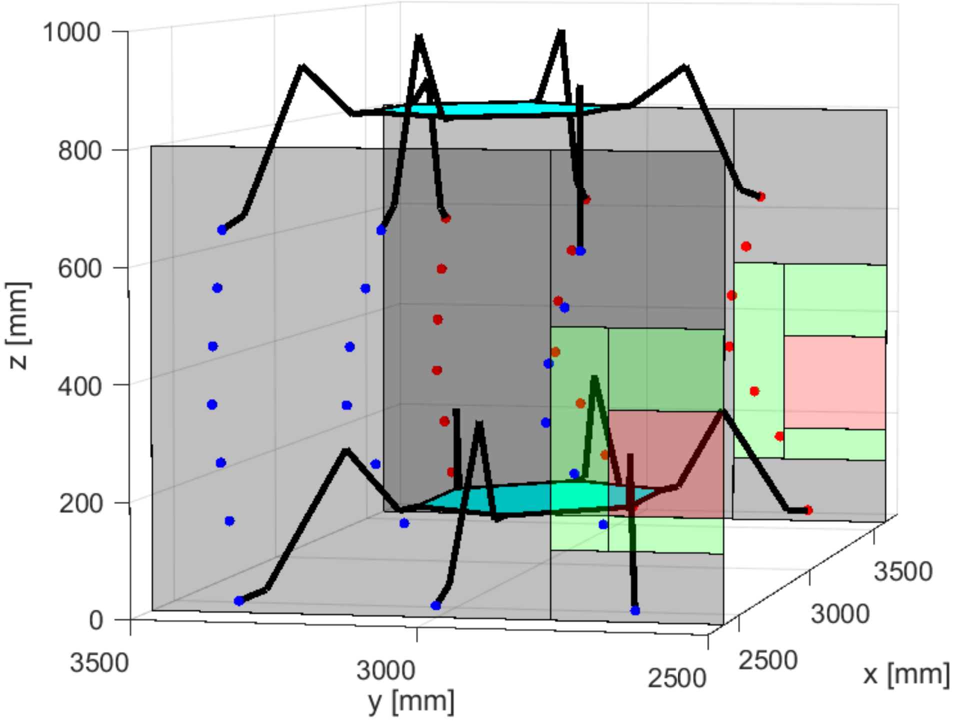


Figure 5.1: A planned trajectory for wall climbing that considers risk arising from slippery terrain. The black area shows a high friction area, the green area shows a low friction area, and the red area shows a zero friction area. Blue and red dots show the planned foot positions, and the hexagons show the body of the robot.

5.2 Problem Formulation

This section describes the friction cone considering maximum gripping forces, model of a position-controlled limbed robot with multi-contact surfaces, and the modeling process of a gripping force through GP.

5.2.1 Friction Cone with Stochastic Gripping Forces

With grippers, the friction cone constraint can be relaxed on the contact point. For our spine-based gripper, even under a zero normal load, the spines insert into the microscopic

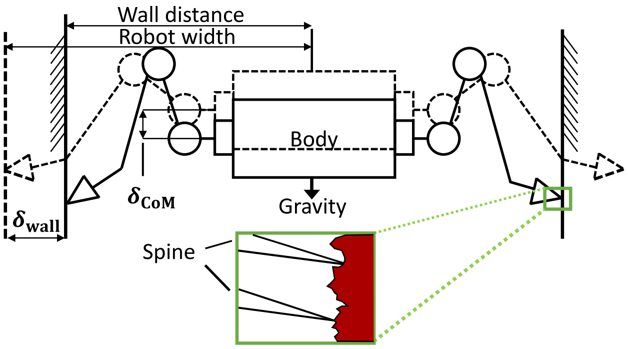


Figure 5.2: Deflection of a multi-limbed robot bracing between walls

gaps on the surface (Fig. 5.2), generating a significant amount of shear force (Fig. 5.6) [114]. For a magnet-based gripper, the reaction forces include the additional magnetic force imposed by the gripper itself, offsetting the friction cone as seen by the rest of the robot.

Thus, we modify the regular friction cone, adding in an offset shear force when a normal force is zero to account for the gripping force. As the normal force increases, the maximal allowable shear force increase in the same way as a regular frictional force, with a coefficient of friction λ that is assumed to be a constant only depending on the property of the contact surface. This contact model is illustrated in Fig. 5.3, where $\boldsymbol{f}^r$ is the reaction force between the surface and the gripper. $\boldsymbol{f}^{g,m}$ is the maximum gripping force from grippers under a zero normal force. Note that $\boldsymbol{f}^{g,m}$ is measured per gripper as a unit. In general, $\boldsymbol{f}^{g,m}$ can have both normal and shear components. However, for our spine grippers, the normal component of $\boldsymbol{f}^{g,m}$ is relatively small, so we assume that the gripper generates only shear adhesion. The gripper does not slip when $\boldsymbol{f}^r$ is within this friction cone, as indicated by the shaded region in Fig. 5.3. Since the interaction between the micro-spines and the surface is highly random, $\boldsymbol{f}^{g,m}$ is naturally modeled as a Gaussian random variable. However, the orientation of the spine and the number of spines in contact with the surface also change as the orientation of the gripper changes, which leads to a shift of the mean and standard deviation of $\boldsymbol{f}^{g,m}$. We learn this model from data by GP. With GP, our proposed planner is able to deal with the stochastic nature of gripping taking into account the gripper orientation.

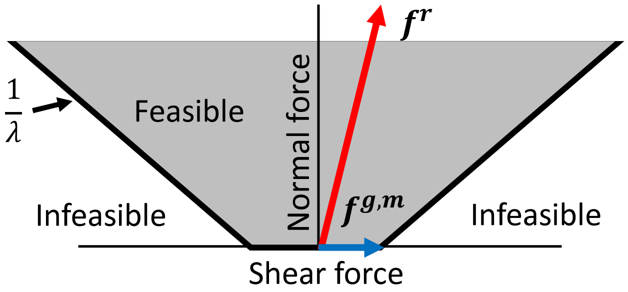


Figure 5.3: Friction cone considering stochastic gripping forces.

5.2.2 Model of Reaction Force Using Limb Compliance

During multi-surface locomotion, the robot leverages the compliance from its motors in order to squeeze itself between multi-surfaces, as depicted in Fig. 5.2. One difficulty multi-limbed robots have is that reaction forces are statically indeterminate [115]. Consequently, reaction forces that cannot be determined by static equilibrium equations when the robot supports its weight more than three contact points. Hence, in order to calculate the reaction force under this condition, the deformation of the robotic system should be considered.

From the standard elasticity theory, $\boldsymbol{f}^r$ can be described as the spring force using the Virtual Joint Method [116]:

$$\boldsymbol{f}^r = \boldsymbol{K}\left(\boldsymbol{\delta}_{\text{wall}} - \boldsymbol{\delta}_{\text{CoM}}\right) \tag{5.1}$$

$$\boldsymbol{K} = \left(\boldsymbol{J}\boldsymbol{k}^{-1}\boldsymbol{J}^{\top}\right)^{-1},\ \boldsymbol{k} = diag(k_i), i = 1, \ldots, H \tag{5.2}$$

where $\boldsymbol{K}$ is the stiffness matrix for H degree-of-freedom limb. $\boldsymbol{k}$ is a diagonal matrix that has k_i diagonal elements, and k_i is the spring coefficients of the position-controlled servos. $\boldsymbol{J}$ is a $3 \times H$ Jacobian matrix. The deflection is imposed by terrain where $\boldsymbol{\delta}_{\text{wall}}$ is the displacement of the terrain and $\boldsymbol{\delta}_{\text{CoM}}$ is the body deflection, sag-down, due to the compliance as shown in Fig. 5.2.

5.2.3 Model of Gripping Force Using Gaussian Process Regression

The objective of using GP is to predict the maximum gripping force $\boldsymbol{f}^{g,m}$ in a probabilistic way.

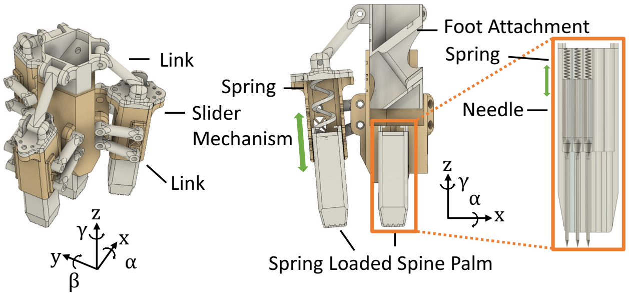


Figure 5.4: Mechanical design of the spine gripper

There are many design decisions that go into the formulation of the GP problem, including choice of kernel, distance metric, and associated weighting between state variables [117]. We can start with the simplest formulation with all state variables equally weighted under the Euclidean distance metric using the squared exponential kernel as a starting point. In practice, this choice was observed to work well enough to not necessitate further design. A more general characterization of the effects of these hyperparameters can be found in [117]. In this work, we assume that the maximum gripping forces by spine grippers is a function of the gripper orientation and the coefficient of friction [107], [118], [114], [119]. This is because with a microscopic view, the spine-asperity interaction is different depending on how a spine is inclined with respect to the asperity as shown in Fig. 5.2. GP can handle the effects of other unmodeled parameters by treating them into uncertainty. Hence, the state $\boldsymbol{s}$ is a four-dimensional vector with $\boldsymbol{s} = [\alpha, \beta, \gamma, \lambda]^\top$ where α, β, γ are the Euler angles along x, y, z axis defined in Fig. 5.4.

Here, we assume that the maximum shear force follows Gaussian distribution. Given a data set $S = \{\boldsymbol{s}_1, \cdots, \boldsymbol{s}_n\}$ with the measured shear forces $\boldsymbol{y}^{g,m} = [y_1^{g,m}, \dots, y_n^{g,m}]^\top$, the maximum shear force $\boldsymbol{f}^{g,m}$ by a gripper can therefore be modeled as:

$$\boldsymbol{f}^{g,m}(\boldsymbol{s}) \sim \mathcal{GP}(\boldsymbol{\mu}^{g,m}(\boldsymbol{s}), \boldsymbol{\kappa}^{g,m}(\boldsymbol{s}, \boldsymbol{s}_*)) \tag{5.3}$$

where, $\boldsymbol{f}^{g,m} = [f_1^{g,m}, \dots, f_n^{g,m}]^\top$. n is the number of samples from a GP. We denote $\boldsymbol{\mu}^{g,m}(\boldsymbol{s}) = [\mu_1^{g,m}(\boldsymbol{s}), \dots, \mu_n^{g,m}(\boldsymbol{s})]^\top$ as the mean and $[\boldsymbol{\kappa}^{g,m}]_{i,j} = \kappa^{g,m}(\boldsymbol{s}_i, \boldsymbol{s}_j)$ as the covariance matrix, where $\kappa^{g,m}(\cdot, \cdot)$ is a positive definite kernel. In this work, we employ the squared exponential kernel as follows:

$$\kappa^{g,m}\left(\boldsymbol{s}_i,\boldsymbol{s}_j\right)=\sigma_f^2\exp\left(-\frac{1}{2}\frac{\left|\boldsymbol{s}_i-\boldsymbol{s}_j\right|^2}{\ell^2}\right) \tag{5.4}$$

where σ_f^2 represents the amplitude parameter and l defines the smoothness of the function $\boldsymbol{f}^{g,m}$.

Here, let $\boldsymbol{D}=\left[\boldsymbol{s}_1,\cdots,\boldsymbol{s}_n\right]^\top$ be the matrix of the inputs. In order to predict the mean and variance matrix at $\boldsymbol{D}_*$, we obtain the predictive mean and variance of the maximum shear force by assuming that it is jointly Gaussian as follows:

$$\hat{\boldsymbol{f}}^{g,m}=\mathbb{E}\left[\boldsymbol{f}^{g,m}\left(\boldsymbol{D}_*\right)\right]=\boldsymbol{\kappa}_*^\top\left(\boldsymbol{K}_D+\sigma_n^2\boldsymbol{I}\right)^{-1}\boldsymbol{y}_g \tag{5.5}$$

$$\hat{\boldsymbol{\Sigma}}^{g,m}=\mathbb{V}\left[\boldsymbol{f}^{g,m}\left(\boldsymbol{D}_*\right)\right]=\boldsymbol{\kappa}_{**}-\boldsymbol{\kappa}_*^\top\left(\boldsymbol{K}_D+\sigma_n^2\boldsymbol{I}\right)^{-1}\boldsymbol{\kappa}_* \tag{5.6}$$

where $\boldsymbol{\kappa}_*=\boldsymbol{\kappa}^{g,m}\left(D_*,D\right)$, $\boldsymbol{K}_D=\boldsymbol{\kappa}^{g,m}\left(D,D\right)$, $\boldsymbol{\kappa}_{**}=\boldsymbol{\kappa}^{g,m}\left(D_*,D_*\right)$, and σ_n^2 is the variance of the Gaussian observation noise with zero mean.

Our GP procedure can be generalizable to model other gripping forces as long as the gripping force changes continuously as the orientation of the gripper changes. For instance, the GP approach can be used to model the gecko gripper force [120] using the detachment angle as the state of the GP.

5.2.4 Spine Gripper for Wall Climbing

A three-finger spine-based gripper was designed (Fig. 5.4) using spine cells based on [119]. Each finger consists of a spine cell with 25 machine needles loaded with 5 mN/mm springs, and a slider mechanism holds the cell with one compliant plastic spring. The diameter of the needle at the tip is 0.93 mm, and it is made of carbon steel. The gripper center module includes one spine palm with the same spine configurations as the cells. The attachment component is fixed at the tip of the robot limb at 37° from the limb axis to maximize the contact area. The finger, center, and attachment members are assembled with a one-slider, two linkage mechanism (Fig. 5.4). This linkage system is designed to provide a passive micro grip as the center palm presses up against a wall. The three fingers are located at 120° apart from each other in z-axis and tilted about 15° from z-axis.

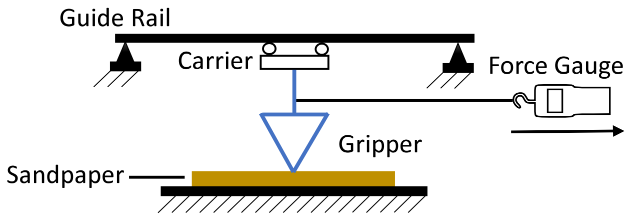


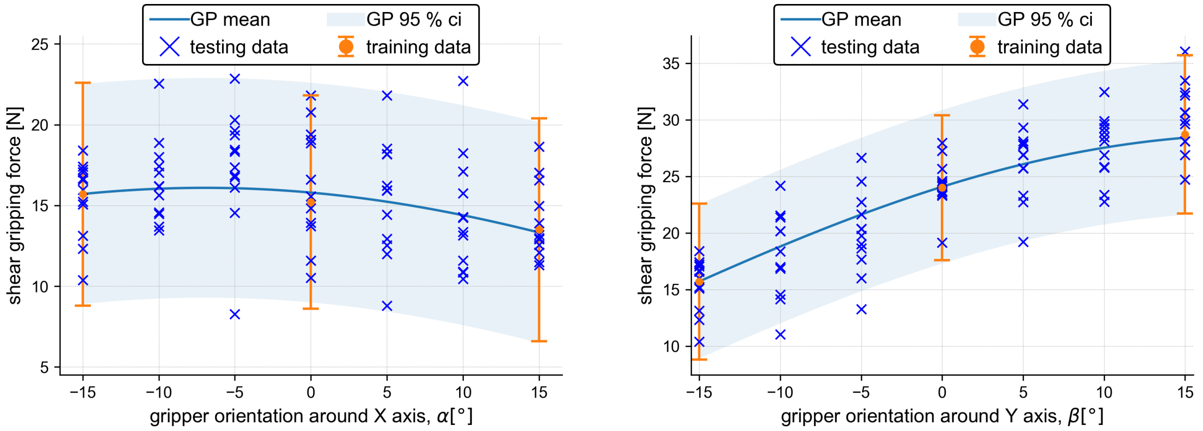
Figure 5.5: Experiment setup to evaluate maximum gripping forces on sandpaper

Table 5.1: Varied orientations for collecting datasets of GP

Training	$\alpha, \beta = -15°, 0°, 15°,\ \gamma = 0°, 30°, 60°,\ \lambda = 1.1, 2.3$
Testing	$\alpha = -15°, -10°, \cdots, 15°\lvert\{\beta = -15°, \gamma = 30°, \lambda = 2.3\}$
	$\beta = -15°, -10°, \cdots, 15°\lvert\{\alpha = -15°, \gamma = 30°, \lambda = 2.3\}$
	$\gamma = 0°, 15°, 30°, \cdots, 60°\lvert\{\alpha = -15°, \beta = -15°, \lambda = 2.3\}$
	$\lambda = 1.1, 1.4, 1.82, 2.3\lvert\{\alpha = 0°, \beta = 0°, \gamma = 0°\}$

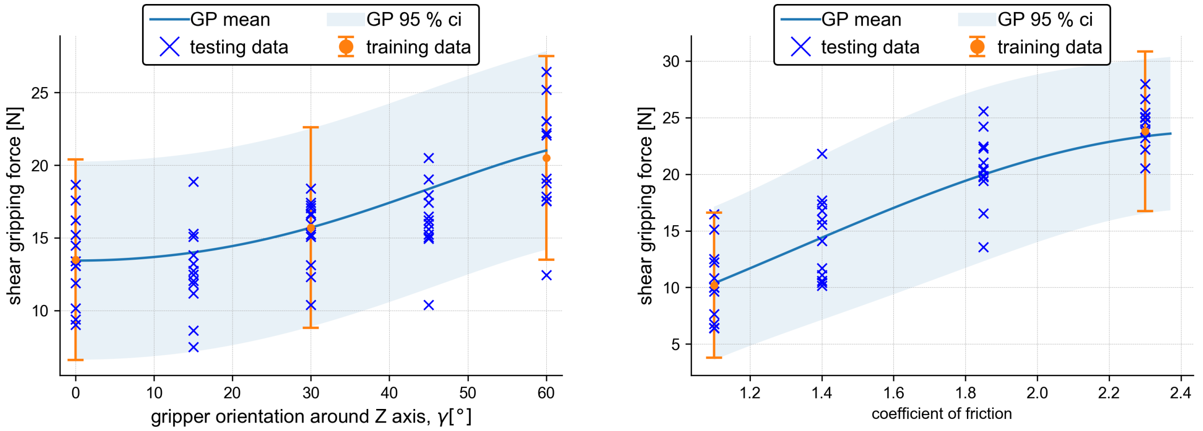
5.2.5 Data Collection

To collect a dataset, maximum gripping forces $\boldsymbol{f}^{g,m}$ were evaluated with a minimal normal force at varied orientations as summarized in Table 5.1. We collected 20 data sequences for every orientation as a training dataset and 10 data sequences as a testing dataset. The coefficient of friction between spines and environments was measured by loading a constant mass on the gripper. A small activation force is necessary to compress spine springs and ensure that the spines are touching the wall, but is assumed to be negligible. These orientations were selected to cover possible gripper angles during regular wall climbing. The gripper was fixed to a linear slider at an orientation and pulled by a force gauge on 36-grit and 80-grit sandpapers that are commonly used to emulate rough surface with microscopic asperities [119], as shown in Fig. 5.5. The GP hyperparameters were optimized using the BFGS algorithm [121]. The obtained testing data with the predicted PDF of the maximum gripping force and the PDF of the training data is illustrated in Fig. 5.6. The predicted maximum gripping force and the training data are displayed as a mean ± with a 95 % confidence interval. Overall, we show that the GP prediction works well with different states.



(a) $\beta = -15°, \gamma = 30°, \lambda = 2.3$

(b) $\alpha = -15°, \gamma = 30°, \lambda = 2.3$



(c) $\alpha = -15°, \beta = -15°, \lambda = 2.3$

(d) $\alpha = 0°, \beta = 0°, \gamma = 0$

Figure 5.6: The predicted maximum gripping force PDF from GP, the training data PDF, and the testing dataset

5.3 Chance-Constrained Nonlinear Programming for Locomotion

In this section, we present a complete risk-aware motion planning algorithm formulated as (5.8a)-(5.8k). The objective of our proposed planner is to find the optimal trajectory for the Center of Mass (CoM) position, its orientation, the foot position, and the reaction force for each foot in order to arrive at the destination while satisfying constraints. Our proposed planner enables the robot to find feasible trajectories that consider risk from the grippers under various environments.

We define one *round* of movement made by a robot when its body and all of its limbs have moved onto the next footholds. Note that for each round, the planner investigates

$$\underset{\Gamma}{\text{minimize}}\ \Psi_{tot} \tag{5.8a}$$

s.t., for each round $j = 1, \ldots, N$

and for each limb $i = 1, \ldots, L$

$$|\Delta \boldsymbol{P}_{\text{CoM}}| \leq \Delta \boldsymbol{P}_{\text{Th}} \qquad \text{(linear stride)} \tag{5.8b}$$

$$|\Delta \boldsymbol{\Theta}_{\text{CoM}}| \leq \Delta \boldsymbol{\Theta}_{\text{Th}} \qquad \text{(angular stride)} \tag{5.8c}$$

$$|\Delta \boldsymbol{p}_i| \leq \Delta \boldsymbol{p}_{\text{Th}} \qquad \text{(foot stride)} \tag{5.8d}$$

$$\boldsymbol{p}_{i,j} \in \mathcal{R}(\boldsymbol{P}_{\text{CoM},j}, \boldsymbol{\Theta}_{\text{CoM},j}, \boldsymbol{\theta}_{i,j}) \qquad \text{(kinematics)} \tag{5.8e}$$

$$\boldsymbol{p}_{i,j} \in \mathcal{T} \qquad \text{(contact region)} \tag{5.8f}$$

$$\sum_{i=1}^{L} \boldsymbol{f}^{r}_{i,j} + \boldsymbol{F}_{tot} = \boldsymbol{0} \qquad \text{(force eqm)} \tag{5.8g}$$

$$\sum_{i=1}^{L} \left(\boldsymbol{p}_{i,j} \times \boldsymbol{f}^{r}_{i,j}\right) + \boldsymbol{M}_{tot} = \boldsymbol{0} \qquad \text{(moment eqm)} \tag{5.8h}$$

$$\boldsymbol{\tau}_{i,j} = \boldsymbol{J}\left(\boldsymbol{\theta}_{i,j}\right)^{\top} \boldsymbol{f}^{r}_{i,j} \qquad \text{(joint torque)} \tag{5.8i}$$

$$\|\boldsymbol{\tau}_{i,j}\|_2 \leq \boldsymbol{\tau}_{\text{Th}} \qquad \text{(torque limit)} \tag{5.8j}$$

$$\boldsymbol{f}^{r}_{i,j} \in \mathcal{F}\left(\lambda_{i,j}(\boldsymbol{p}_{i,j}), \boldsymbol{n}_{i,j}, \boldsymbol{f}^{g,m}_{i,j}\right) \qquad \text{(friction cone)} \tag{5.8k}$$

several critical instants between two postures with a pre-defined gait as explained in detail in Section 5.4. At j-th round, Γ is the decision variables that are given as:

$$\Gamma = \{\boldsymbol{p}_{i,j}, \boldsymbol{P}_{\text{CoM},j}, \boldsymbol{\Theta}_{\text{CoM},j}, \boldsymbol{\theta}_{i,j}, \boldsymbol{f}^{r}_{i,j}, \hat{\boldsymbol{f}}^{g,m}_{i,j}, \hat{\boldsymbol{\Sigma}}^{g,m}_{i,j}\} \tag{5.7}$$

where $\boldsymbol{p}_{i,j}$ is the foot i position, $\boldsymbol{P}_{\text{CoM},j}$ is the position of the body, $\boldsymbol{\Theta}_{\text{CoM},j}$ is the orientation of the body, $\boldsymbol{\theta}_{i,j}$ are the joint angles for the limb i, and $\boldsymbol{f}^{r}_{i,j}$ is defined in Section 5.2.1. In this study, $\boldsymbol{f}^{g,m}_{i,j}$ is treated as a random variable based on the model of GP, which follows $\boldsymbol{f}^{g,m}_{i,j} \sim N\left(\hat{\boldsymbol{f}}^{g,m}_{i,j}(s_{i,j}), \hat{\boldsymbol{\Sigma}}^{g,m}_{i,j}(s_{i,j})\right)$. Equation (5.8a) is the cost function that depends on the robot's state. Equation (5.8b), (5.8c), and (5.8d) bound the range of travel between rounds. Equation (5.8e) represents the forward kinematics constraints. In (5.8f), it ensures that $\boldsymbol{p}_{i,j}$ is within the feasible terrain where the robot is able to put its limb. In this chapter, we assume that the robot generates a quasi-static motion. Hence, the planner has the static equilibrium constraints expressed by (5.8g) and (5.8h), where $\boldsymbol{F}_{tot}$ and $\boldsymbol{M}_{tot}$ is the external force and moment, respectively. In this work, only gravity is considered as the external force. Equation (5.8i) and (5.8j) ensure that the motor torque is lower than the maximum motor torque where $J\left(\boldsymbol{\theta}_i\right)$ is a Jacobian matrix. The reaction force $\boldsymbol{f}_i^r$ is constrained by (5.8k), which describes the friction cone constraints to prevent the robot from slipping where $\lambda_{i,j}(\boldsymbol{p}_{i,j})$ denotes the coefficient of friction at $\boldsymbol{p}_{i,j}$. Note that this constraints (5.8k) is also stochastic constraints due to $\boldsymbol{f}_i^r$. Equation (5.8k) can be converted into deterministic constraints, which is explained in Section 5.3.2. If a robot is position-controlled, the planner needs to add additional constraints to compute the control input to the motor to generate the planned forces, which are defined in (5.1), (5.2).

Compared to sampling-based approaches such as RRT, NLP is able to formulate relatively complicated constraints such as friction cone constraints (5.8k), which are typically difficult for the sampling-based approaches to handle in terms of computation. In addition, MICP approaches such as [75], [9], [122] can increase the computation speed by decoupling the pose state from wrench states. However, they potentially limit the robot's mobility. The robot may not choose the trajectory on the low friction terrain in case the planner first solves the pose problem and then solves the wrench problem later since the pose optimization problem does not consider the wrench information. Although MICP can plan the trajectories considering both wrench and pose state simultaneously, it needs to sacrifice the accuracy by assuming an envelope approximation on bilinear terms [75] or allow relatively expensive computation as the number of the integer variables increases, which is intractable for high degree-of-freedom (DoF) robots (e.g., our robot has 24 DoF). In contrast, NLP can simultaneously solve the trajectory reasoning both the pose and the wrench with relatively less computation [14].

5.3.1 Deterministic Constraints

Here, we explain two deterministic constraints (5.8e), (5.8f), that are not explicitly shown in (5.8a)-(5.8k).

5.3.1.1 Kinematics

Forward kinematics (5.8e) is given as:

$$\boldsymbol{p}_{i,j} = \boldsymbol{R}(\boldsymbol{\Theta}_{\mathrm{CoM},j})\boldsymbol{p}^{b}_{i,j} + \boldsymbol{P}_{\mathrm{CoM},j} \tag{5.9}$$

where $\boldsymbol{R}(\boldsymbol{\Theta}_{\mathrm{CoM}})$ is the rotation matrix from the world frame to the body frame, $\boldsymbol{p}^{b}_{i}$ is the foot position relative to the body frame.

5.3.1.2 Feasible Contact Regions

We utilize NLP to formulate the planning algorithm so that any nonlinear terrain (i.e., non-flat terrain), such as tube and curve, can be directly described. Obstacle avoidance can be realized by defining these constraints, which do not include the obstacle terrain.

If a robot traverses on the flat terrain, the footstep regions are convex polygons as follows:

$$\boldsymbol{C}_{r}\boldsymbol{p}_{i,j} \leq \boldsymbol{D}_{r} \tag{5.10}$$

5.3.2 Chance Constraints

Here, we show that the friction cone constraints in (5.8k) can be expressed using chance constraints, which allow the planner to convert the stochastic constraints into deterministic constraints.

One key characteristic of robotic climbing is that climbing is a highly risky operation: a robot can easily fall without planning its motion correctly. Hence, it needs to restrict reaction forces using the friction cone constraints given as:

$$\boldsymbol{n}^{\top}_{i,j}\boldsymbol{f}^{r}_{i,j} \geq 0 \tag{5.11}$$

$$\left\|\boldsymbol{f}_{i,j}^{r}-\left(\boldsymbol{n}_{i,j}^{\top}\boldsymbol{f}_{i,j}^{r}\right)\boldsymbol{n}_{i,j}^{\top}\right\|_{2}\leq\lambda_{i}\left(\boldsymbol{n}_{i,j}^{\top}\boldsymbol{f}_{i,j}^{r}\right)+\boldsymbol{f}_{i,j}^{g,m}\tag{5.12}$$

To decrease the computation of solving for the NLP solver, we simplify the (5.12) by linearizing them as follows:

$$\left|\boldsymbol{\zeta}_{i,j}^{\top}\boldsymbol{f}_{i,j}^{r}\right|\leq\lambda_{i}\left(\boldsymbol{n}_{i}^{\top}\boldsymbol{f}_{i,j}^{r}\right)+\boldsymbol{f}_{i,j}^{g,m}\tag{5.13}$$

$$\left|\boldsymbol{\xi}_{i,j}^{\top}\boldsymbol{f}_{i,j}^{r}\right|\leq\lambda_{i}\left(\boldsymbol{n}_{i}^{\top}\boldsymbol{f}_{i,j}^{r}\right)+\boldsymbol{f}_{i,j}^{g,m}\tag{5.14}$$

where $\boldsymbol{\zeta}_{i,j}$, $\boldsymbol{\xi}_{i,j}$ are any tangential direction vectors on the wall plane.

Regarding (5.8k) formulated as (5.11), (5.13), (5.14), we rearrange the equations and the joint chance constraint is given by:

$$\Pr\left(\bigwedge_{j=1,\ldots,N}\bigwedge_{k=1,\ldots,M}\boldsymbol{\alpha}_{i,j}^{k\top}\boldsymbol{f}_{i,j}^{g,m}\leq\beta_{i,j}^{k}\right)\geq 1-\Delta\tag{5.15}$$

where $\boldsymbol{\alpha}_{i,j}^{k}$ are coefficient vectors, and $\beta_{i,j}^{k}$ are coefficient scalars that consist of the convex polytopes defined in (5.11), (5.13), (5.14). In (5.15), M denotes the number of constraints defining the polytopes. Δ is the user-defined violation probability, where the probability of violating constraints is under the Δ. We can regard Δ as relating to the likelihood that gripper slip will be responsible for the failure of the robot. For example, if Δ is high, the planner can explore a larger space because the feasible region expands in optimization. As a result, the robot tends to plan a trajectory with a high violation probability by assuming that the gripper generates enough force. For a robotic climbing task, these chance constraints enable the robot to perform challenging motions that would be infeasible without considering the gripping force. In contrast, if Δ is small, the planner tends to generate more conservative motions because the robot assumes that the gripper does not output enough force to support the weight of the robot.

Imposing (5.15) is computationally intractable. Thus, using Boole's inequality, Blackmore [36], showed that the feasible solution to (5.15) is the feasible solution to the following equations:

$$\Pr\left(\boldsymbol{\alpha}_{i,j}^{k\top}\boldsymbol{f}_{i,j}^{g,m}\leq\beta_{i,j}^{k}\right)\geq 1-\Delta_{j,k}\tag{5.16}$$

$$\sum_{j=1}^{N}\sum_{k=1}^{M}\Delta_{j,k}\leq\Delta\tag{5.17}$$

for all $j = 1, \ldots, N, \quad k = 1, \ldots, M$. The violation probability for each constraint per round $\Delta_{j,k}$ is constrained in (5.17), in order not to exceed the given Δ. Because non-uniform risk allocation (5.17) is also computationally expensive [123], we use the following relation:

$$\Delta_{j,k} = \frac{\Delta}{NM} \tag{5.18}$$

$\boldsymbol{\alpha}_{i,j}^{k\top} \boldsymbol{f}_{i,j}^{g,m}$ is a multivariate Gaussian distribution which follows the Gaussian distribution $\boldsymbol{\alpha}_{i,j}^{k\top} \boldsymbol{f}_{i,j}^{g,m} \sim N\left(\boldsymbol{\alpha}_{i,j}^{k\top} \hat{\boldsymbol{f}}_{i,j}^{g,m}, \boldsymbol{\alpha}_{i,j,k^\top} \hat{\boldsymbol{\Sigma}}_{i,j}^{g,m} \boldsymbol{\alpha}_{i,j}^{k}\right)$. Thus, the stochastic constraints (5.16) can be then converted into a deterministic constraint as given by:

$$\begin{aligned} \Pr\left(\boldsymbol{\alpha}_{i,j}^{k\top} \boldsymbol{f}_{i,j}^{g,m} \leq \beta_{i,j}^{k}\right) &= \Phi\left(\frac{\beta_{i,j}^{k} - \boldsymbol{\alpha}_{i,j}^{k\top} \hat{\boldsymbol{f}}_{i,j}^{g,m}}{\sqrt{\boldsymbol{\alpha}_{i,j}^{k\top} \hat{\boldsymbol{\Sigma}}_{i,j}^{g,m} \boldsymbol{\alpha}_{i,j}^{k}}}\right) \\ &\geq 1 - \Delta_{j,k} \end{aligned} \tag{5.19}$$

where Φ is the cumulative distribution function of the standard normal distribution. It can be transformed further as follows:

$$\boldsymbol{\alpha}_{i,j}^{k\top} \hat{\boldsymbol{f}}_{i,j}^{g,m} + \sqrt{\boldsymbol{\alpha}_{i,j}^{k\top} \hat{\boldsymbol{\Sigma}}_{i,j}^{g,m} \boldsymbol{\alpha}_{i,j}^{k}}\, \Phi^{-1}\left(1 - \Delta_{j,k}\right) \leq \beta_{i,j}^{k} \tag{5.20}$$

where Φ^{-1} is the inverse function of Φ.

5.3.3 Cost Function

The cost function consists of intermediate costs and a terminal cost. In this work, the target mission is to arrive at the destination. Thus, the terminal cost is the distance from the position of the last pose to the destination.

$$\Psi_{\mathrm{D}} = \left(\boldsymbol{q}_N - \boldsymbol{q}_D\right)^{\top} \mathbf{W}_{\mathrm{D}} \left(\boldsymbol{q}_N - \boldsymbol{q}_d\right) \tag{5.21}$$

where $\mathbf{W}_{\mathrm{D}}$ is the weighting matrix and $\boldsymbol{q}_N = \left[\boldsymbol{p}_{1,N}, \ldots, \boldsymbol{p}_{L,N}\right]$ while $\boldsymbol{q}_d$ is the configuration at the destination. The intermediate costs restrict a large amount of shifting in terms of linear and rotational motion of a body and the foot position as follows:

$$\begin{aligned}
\Psi_{\mathrm{BPos}} &= \Delta \boldsymbol{P}_{\mathrm{CoM}}^{\top} \mathbf{W}_{\mathrm{BPos}} \Delta \boldsymbol{P}_{\mathrm{CoM}} \\
\Psi_{\mathrm{Foot}} &= \sum_{i=1}^{L} \Delta \boldsymbol{p}_{i}^{\top} \mathbf{W}_{\mathrm{Foot}} \Delta \boldsymbol{p}_{i} \\
\Psi_{\mathrm{BRot}} &= \Delta \Theta_{\mathrm{CoM}}^{\top} \mathbf{W}_{\mathrm{BRot}} \Delta \Theta_{\mathrm{CoM}}
\end{aligned} \tag{5.22}$$

where $\mathbf{W}_{\mathrm{BPos}}$, $\mathbf{W}_{\mathrm{Foot}}$, and $\mathbf{W}_{\mathrm{BRot}}$ are the weighting matrix.

5.3.4 Two Step Optimization for a Position-Controlled Robot

Although our proposed motion planner works for any limbed robot, there is a drawback for a position-controlled robot when wall-climbing. For the position-controlled robot, it is necessary to compute how much $\boldsymbol{\delta}_{\mathrm{wall}}$ is necessary to generate the planned reaction forces. Therefore, the planner needs to include additional constraints from (5.1), (5.2) to realize the planned trajectory. However, we observed that the nonlinear solver has a numerical issue with (5.2), so it is intractable for the solver to solve our proposed NLP in (5.8a)-(5.8k) with (5.1), (5.2). To avoid this problem, we decouple the optimization problem into two-step problems shown in (5.23a)-(5.23l) and (5.24a)-(5.24d):

$$\underset{\Gamma}{\text{minimize}}\ \Psi_{\mathrm{D}} + \sum_{j=1}^{N-1}(\Psi_{\mathrm{BPos}} + \Psi_{\mathrm{Foot}} + \Psi_{\mathrm{BRot}}) \tag{5.23a}$$

$$\text{s.t.} |\boldsymbol{P}_{\mathrm{CoM},j+1} - \boldsymbol{P}_{\mathrm{CoM},j}| \leq \Delta \boldsymbol{P}_{\mathrm{Th}} \tag{5.23b}$$

$$|\boldsymbol{\Theta}_{\mathrm{CoM},j+1} - \boldsymbol{\Theta}_{\mathrm{CoM},j}| \leq \Delta \boldsymbol{\Theta}_{\mathrm{Th}} \tag{5.23c}$$

$$|\boldsymbol{p}_{i,j+1} - \boldsymbol{p}_{i,j}| \leq \Delta \boldsymbol{p}_{\mathrm{Th}} \tag{5.23d}$$

$$\boldsymbol{p}_{i,j} = \boldsymbol{R}(\boldsymbol{\Theta}_{\mathrm{CoM},j})\boldsymbol{p}^{b}_{i,j} + \boldsymbol{P}_{\mathrm{CoM},j} \tag{5.23e}$$

$$\boldsymbol{C}_r \boldsymbol{p}_{i,j} \leq \boldsymbol{D}_r \tag{5.23f}$$

$$\sum_{i=1}^{L} \boldsymbol{f}^{r}_{i,j} + \boldsymbol{F}_{tot} = \boldsymbol{0} \tag{5.23g}$$

$$\sum_{i=1}^{L} \left(\boldsymbol{p}_{i,j} \times \boldsymbol{f}^{r}_{i,j}\right) + \boldsymbol{M}_{tot} = \boldsymbol{0} \tag{5.23h}$$

$$\boldsymbol{\tau}_{i,j} = \boldsymbol{J}\left(\boldsymbol{\theta}_{i,j}\right)^{\top} \boldsymbol{f}^{r}_{i,j} \tag{5.23i}$$

$$\|\boldsymbol{\tau}_{i,j}\|_2 \leq \Delta \boldsymbol{\tau} \tag{5.23j}$$

$$\boldsymbol{\alpha}^{k\top}_{i,j} \hat{\boldsymbol{f}}^{g,m}_{i,j} + \sqrt{\boldsymbol{\alpha}^{k\top}_{i,j} \hat{\boldsymbol{\Sigma}}^{g,m}_{i,j} \boldsymbol{\alpha}^{k}_{i,j}}\, \Phi^{-1}\left(1 - \Delta_{j,k}\right) \leq \beta^{k}_{i,j} \tag{5.23k}$$

$$\Delta_{j,k} = \frac{\Delta}{NM} \tag{5.23l}$$

$$\text{find } \boldsymbol{\delta}_{\mathrm{wall},i,j}, \boldsymbol{\delta}_{\mathrm{CoM},i,j} \tag{5.24a}$$

$$\text{s.t. } \|\boldsymbol{\delta}_{\mathrm{wall},i,j}\|_2 \leq \boldsymbol{\delta}_{\mathrm{Th,wall},i,j} \tag{5.24b}$$

$$\boldsymbol{f}^{r}_{i,j} = \boldsymbol{K}_i \left(\boldsymbol{\delta}_{\mathrm{wall},i,j} - \boldsymbol{\delta}_{\mathrm{CoM},i,j}\right) \tag{5.24c}$$

$$\boldsymbol{K_{i,j}} = \left(\boldsymbol{J}\left(\boldsymbol{\theta_{i,j}}\right) \boldsymbol{k}^{\boldsymbol{-1}} \boldsymbol{J}\left(\boldsymbol{\theta_{i,j}}\right)^{\top}\right)^{\boldsymbol{-1}} \tag{5.24d}$$

the first planner is in charge of the pose and the reaction force of the robot, and the second planner finds $\boldsymbol{\delta}_{\mathrm{wall},i,j}, \boldsymbol{\delta}_{\mathrm{CoM},i,j}$, which are the control inputs to a position-controlled robot. In (5.24a)-(5.24d), the constraint (5.24b) ensures that $\boldsymbol{\delta}_{\mathrm{wall},i,j}$ is bounded under a certain threshold.

We argue that this decoupling is reasonable because the first planner solves the *"essential"*

Table 5.2: NLP specifications for climbing on non-uniform walls

# of rounds N	Variables	Constraints	Average T-solve (Ipopt)
1	1744	779	0.4 minutes
2	3937	1680	6 minutes
4	11761	4994	16 minutes
7	23479	9965	248 minutes

problem (e.g., How much reaction force is necessary? What is the footstep trajectory?) to plan the force and pose trajectory. The second planner only computes the control input to the motors, and it does not have a significant effect on the entire motion planning. As explained, if the robot is force-controlled, the planner does not need to consider (5.1), (5.2). As a result, the second optimization is not necessary for a force-controlled robot, and the whole motion is planned only based on the first optimization problem.

5.4 Results

In this section, we evaluate our proposed planner by testing the robot's performance in three different tasks: energy-efficient climbing, climbing on non-uniform terrains, and climbing with a tripod gait.

We utilize Ipopt solver [10] to solve the planning problem on an Intel Core i7-8750H machine. The derivative of constraints are provided by CasAdi [124]. The optimizer is initialized with the default configuration of the robot (Fig. 5.1, bottom configuration), and the specifications of the computation for Section 5.4.2 is summarized in Table 5.2.

We implement the results of our proposed planning algorithm (i.e., the motion plan), on a six-limbed robot, each limb of which has three DoF. Each joint uses pairs of Dynamixel MX-106 motors, providing a maximum torque at 27 Nm. The robot is equipped with a battery, computer, and IMU. The robot runs a PID loop to regulate its body orientation. No other sensor is used to control its linear position. The robot weighs 11.5 kg. The width of the robot's body is 442 mm while its height at its standing state is 180 mm. In each experiment, the robot climbs between two walls at a distance of 1200 mm, where the wall is covered with sandpapers of different grit sizes to adjust the coefficient of friction. All hardware demonstrations can be viewed in the accompanied video[1].

5.4.1 Energy Efficient Planning

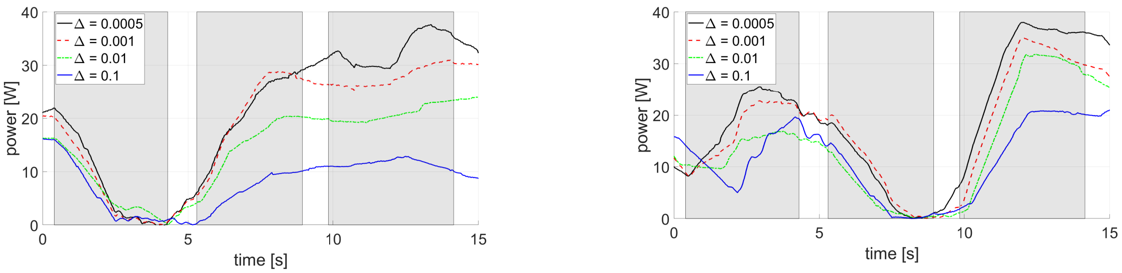
The objective of this task is to assess the consumed energy of climbing with two different violation probabilities. While the robot can grip the wall with a low violation probability (e.g., $\Delta = 0.0005$), there is a disadvantage of consuming more energy. On the other hand, the robot may perform an energy-efficient motion with a higher violation probability (e.g., $\Delta = 0.1$). Here, we set $N = 7$, $M = 6$ to compute $\Delta_{j,k}$. To show the trade-off between the consumed energy and the violation probability, we let the robot climb on the walls with one leg gait where the robot first lifts its right front limb, puts it on the next position, pushes its body up, lifts its right middle limb, and so on. Within each round, the planner investigates 12 critical instants for one leg gait: 6 instants after the robot lift one limb, and 6 instants after the robot places the limb on the next position and pushes its body. The planner solves the optimization problem for these 12 instants. We measure the current $I_{i,t}$ and the voltage $V_{i,t}$ of each limb i online when the robot climbs on the wall covered by the 36-grit sandpapers with one leg gait and estimated the power per one limb at every sampling time t. The power $P_{i,t}$ is estimated as follows:

$$P_{i,t} = V_{i,t} \times I_{i,t} \tag{5.25}$$

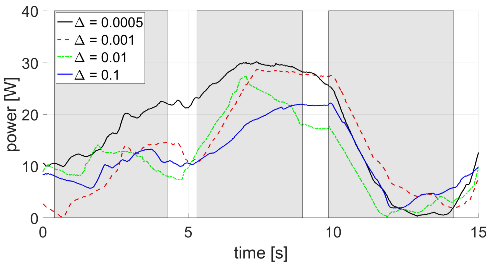
We plot the consumed power for two consecutive limbs from the hardware experiment in Fig. 5.7. Fig. 5.7 shows that the consumed power of a limb decreases when the limb is in the air while the other limbs increase the consumed power to increase the reaction force. Furthermore, the robot consumes more power with smaller Δ, which means that the robot needs to push the wall to increase f^r. In contrast, if $\Delta = 0.1$, the solution requires less

---

[1] Video of hardware experiments: https://youtu.be/ZDqvf1J4nS4



(a) Time history of the consumed power for right front limb (b) Time history of the consumed power for right middle limb



(c) Time history of the consumed power for right back limb

Figure 5.7: Time history of the consumed power under the different violation probabilities. The shaded regions are when the robot lifts a specific limb and puts it on the next position, and white regions are when the robot pushes its body up. The figure shows that the consumed power of a particular limb decreases when the limb is in the air, while it increases when the limb is on the wall to generate the normal force on the wall.

power, but has a larger probability of slipping. In Fig. 5.8, the total consumed energy from these limbs was calculated by integrating their power over time spent climbing. In our robot, the robot could decrease the energy by 46.5 % under $\Delta = 0.1$ compared with the energy under $\Delta = 0.0005$.

5.4.2 Climbing on Non-Uniform Walls

This scenario demonstrates that the robot designs different trajectories under the different violation probability to climb on walls with varying coefficients of friction. The planned trajectories are shown in Fig. 5.9. In this example, the robot climbs between two walls where the terrain shown in black is covered by 36-grit sandpapers ($\lambda = 2.3$), the terrain shown in green is covered by 80-grit sandpapers ($\lambda = 1.1$), and the terrain shown in red is

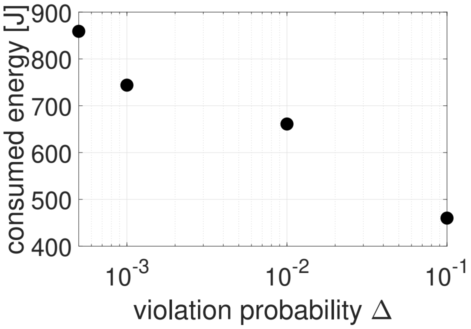


Figure 5.8: Consumed energy with different Δ during $t = 0 - 15$ s

covered by the material with $\lambda = 0$ as shown in Fig. 5.9. The varying coefficients of friction are modeled by a parabola function, which encourages the solver to converge on a solution. In the left panel of Fig. 5.9, the violation probability Δ is 0.1 while in the right panel, the violation probability Δ is 0.001 for $M = 6$ and $N = 7$.

The left panel of Fig. 5.9 illustrates that the robot avoids the red area (zero friction) and puts its foot mostly in the black area (high friction), but sometimes also in the green area (low friction) to minimize the trajectory length. In the right panel of Fig. 5.9, the violation probability is decreased, and the robot footsteps completely remain inside the high friction area. As a result, our proposed NLP-based planner operates the pose and forces together and makes a trade-off between a shorter but more risky trajectory and a longer but safer trajectory. This cannot be achieved if the planner decouples the footstep and force planning, such as in [122]. Fig. 5.10 shows the trajectory with higher risk bound $\Delta = 0.1$ and compares the foot location at $t = 146$ with the foot location with $\Delta = 0.001$ in the hardware experiment. We notice that at $t = 146$ s, the foot touches the white area where the coefficient of friction is 0, which never happened with $\Delta = 0.001$. Since the robot only controls its body orientation based on IMU feedback and does not control its linear position, the implemented trajectory does not strictly follow the planned one. We observe that a lower risk bound is beneficial in this situation to avoid failure since it compensates for the tracking error by the imperfect controller.

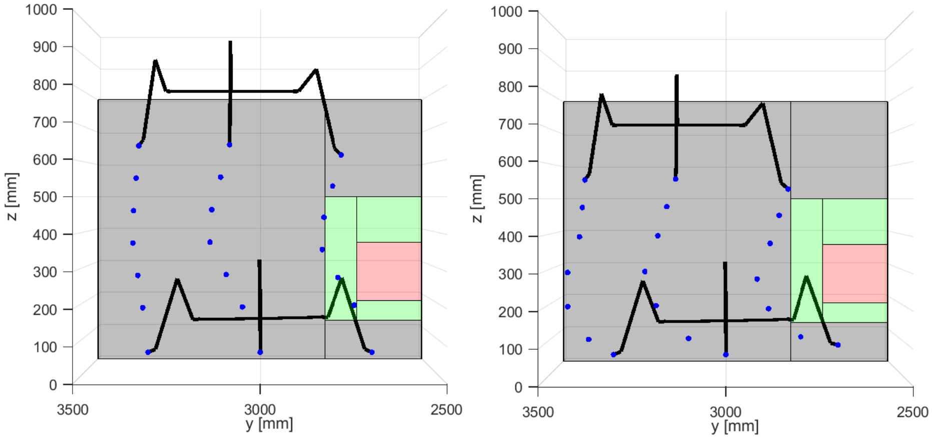


Figure 5.9: Side view of planned footsteps on non-uniform walls under: left $\Delta = 0.1$, right $\Delta = 0.001$. In the left panel, the robot puts its feet on low and high friction terrain by taking a high risk bound. In the right panel, the robot puts its feet only on high friction terrain.

5.4.3 Climbing with Less Stable Gait: Tripod Gait

In this scenario, we demonstrate that the robot can conduct a tripod gait, when it lifts three legs simultaneously, by setting the violation probability much higher. Before installing the gripper on the current six-limbed robot, it was almost infeasible to climb on the walls with the tripod gait because of the torque limits of the motors. With the grippers installed, however, the robot has a greater chance to climb on the walls with a tripod gait. If we set $\Delta = 0$, the problem is infeasible since the constraints under the worst-case uncertainty are conservative. This result would be equivalent to the results of another robust algorithm such as [125], where the optimization-based robust approach with the worst-case uncertainty is proposed. If we apply the the optimization-based robust approach considering the worst-case uncertainty [125] to plan the trajectory, the planning problem is infeasible since the constraints under worst-case uncertainty are conservative. However, by utilizing the chance constraints and increasing the violation probability, the planner generates a feasible solution. In our trial, we set the violation probability $\Delta = 0.4$ for $M = 6$ and $N = 3$, and allowed the robot to climb on a wall covered by 36-grit sandpapers. The planner investigates 4 critical instants: 2 instants after the robot lifts three limbs, and 2 instants after the robot places them down and pushes its body. The planned trajectory is illustrated in the left panel of

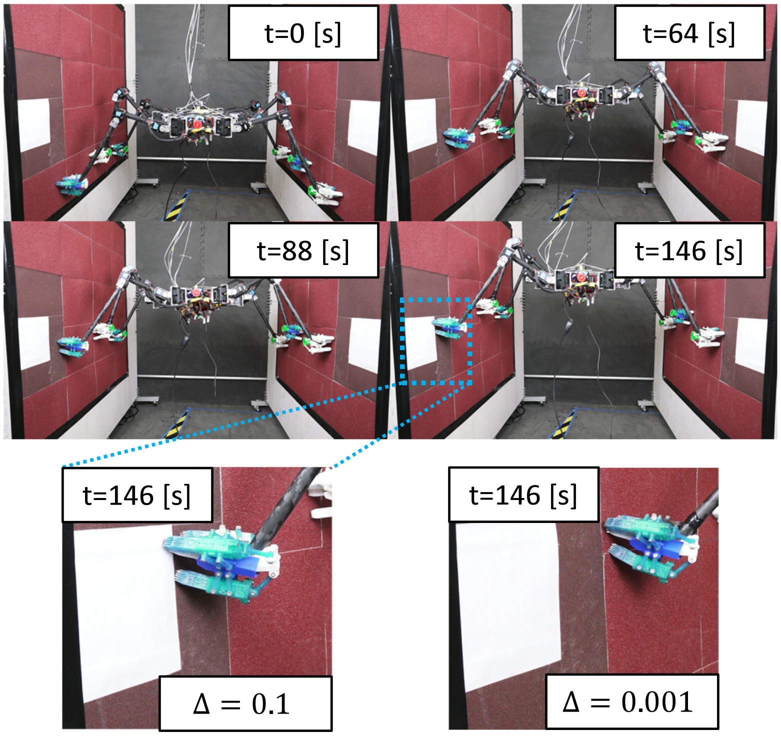


Figure 5.10: Snapshots of climbing experiments on non-uniform walls under the different violation probabilities

Fig. 5.11. As shown in the right under the condition, the robot succeeded in climbing on the walls with the tripod gait and its climbing velocity was 2.5 cm/s, which is three times faster than the one leg gait. Hence, the robot showed that it has the capability of climbing with a tripod gait equipped with grippers by taking a large relative risk. However, we observed that the robot failed to climb more than 40 % using tripod gait since the robot sometimes has a large amount of rotational sag-down, which is not modeled in this chapter. Therefore, modeling that considers this rotational sag-down is for future research.

5.5 Conclusion

In this chapter, we presented a motion planning algorithm for limbed robots with stochastic gripping forces. Our proposed planner exploits NLP to simultaneously plan a pose and force with guaranteed bounded risk. Maximum gripping forces are modeled as a Gaussian distri-

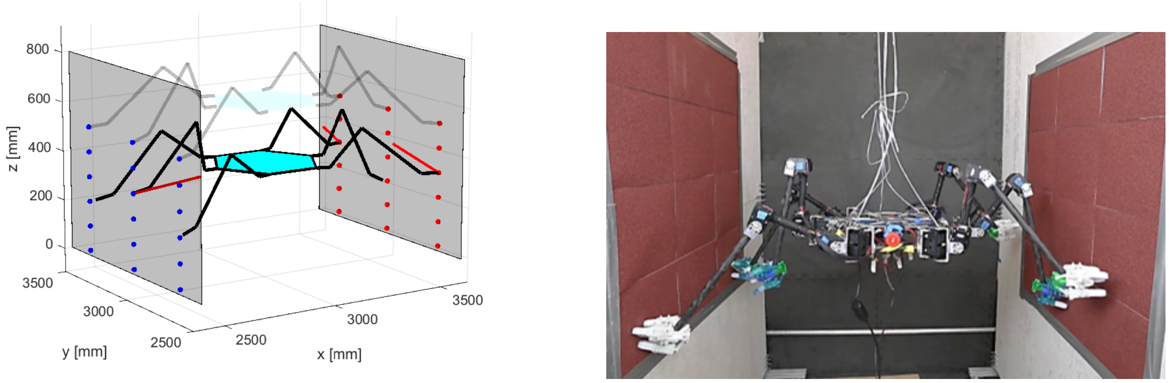


Figure 5.11: Climbing with tripod gait. Left: A planned trajectory of the tripod gait under $\Delta = 0.4$. Red arrows indicate the reaction forces from the walls. Right: A snapshot of climbing experiments with the tripod gait under $\Delta = 0.4$.

bution by employing the GP, which provides the planner with the mean and the covariance information to formulate the chance constraints. We showed that under our planning framework, the robot demonstrates rich - sometimes drastically different - behaviors, including planning a risky but energy-efficient motion versus a safe but exhausting motion, avoiding danger zones like low friction environments and choosing fast but less stable motions (i.e., a tripod gait) based on the different violation probabilities Δ in hardware experiments.

In this chapter, we do not consider the change of contact modes. We instead formulate the chance-constrained optimization algorithm that prevents the robot from changing the contact modes. Although the algorithm presented in this chapter works in practice, there are several research challenges we observe. One of them is that we do not consider the propagation of uncertainty due to stochastic parameters such as friction constants. We describe how stochastic parameters lead to stochastic states in the next chapters, which is necessary for achieving robust manipulation and locomotion with multiple contacts.

CHAPTER 6

Chance-Constrained Optimization for Uncertain Contact-Rich Systems

In Chapter 5, we present chance-constrained optimization for multi-limbed robots which has several assumptions. First, in Chapter 5, we do not consider the propagation of uncertainty since dynamics is modeled as quasi-static. Second, we assume that contact mode does not change due to the uncertainty, which is not true in reality.

In this chapter, we relax these assumptions. We discuss the linear dynamical system with complementarity constraints under uncertainty. We explicitly consider the change of the contact modes and the propagation of the uncertainty for the contact-rich systems. We propose MIP-based optimization framework to design robust open-loop controller. We verify that our framework is able to achieve robust performance with chance constraints under uncertainty.

This chapter has been partially adapted from the following papers:

- **Y. Shirai**, D. Jha, A. Raghunathan, and D. Romeres, "Chance-Constrained Optimization for Contact-rich Systems using Mixed Integer Programming", (under review for the Nonlinear Analysis: Hybrid Systems).
- **Y. Shirai**, D. Jha, A. Raghunathan, and D. Romeres, "Chance-Constrained Optimization in Contact-rich Systems", in *Proc. 2023 American Cont. Conf.*, pp. 14-21, 2023.

6.1 Overview

Contacts are central to most manipulation problems. Consequently, contact modeling has been an active area of research in robotics for the last several decades [126, 8, 12, 127, 128, 26]. One of the most popular approaches to model contact dynamics is using Linear Complementarity Problem (LCP). The distinguishing feature of a complementarity problem is the set of complementarity conditions. Each of these conditions requires that the product of two or more non-negative quantities (either a decision variable or a function of decision variables) should be zero [129]. Linear Complementarity systems have been widely used to succinctly represent hybrid dynamical systems [130]. LCP models are widely used for modeling contact dynamics since they allow a compact representation of hybrid dynamics compared to mode enumeration. They have been also used in several physics simulation engines such as Bullet, ODE, etc. Consequently, linear complementarity systems have been extensively explored in robotics research in various domains like manipulation and locomotion. For example, contact-implicit trajectory optimization (CITO) models contact as complementarity constraint between contact forces and relative accelerations, and the optimization is formulated as a mathematical program with complementarity constraints (MPCC) [131]. Such techniques have been widely used to solve complex manipulation [132, 131, 133, 134, 135] and locomotion problems [7]. Similarly, Lyapunov stability of linear systems with complementarity systems has also been studied [136, 137, 138]. However, almost all of these works assume deterministic contact models for planning. In reality, contact-rich systems could suffer from several uncertainties which lead to stochastic dynamics and thus, it is important to consider uncertainty during planning. Modeling uncertainty in LCP-based contact models leads to a Stochastic Discrete-time Linear Complementarity System (SDLCS).

Figure 6.1 shows an example of a stochastic planar pushing system which naturally leads to stochastic evolution of system states due to stochastic frictional interaction during pushing. However, the complementarity constraints in SDLCS pose unique challenges for the formulation of robust or stochastic optimization of SDLCS. This is mostly because of the non-differentiability of the complementarity constraints which makes uncertainty propaga-

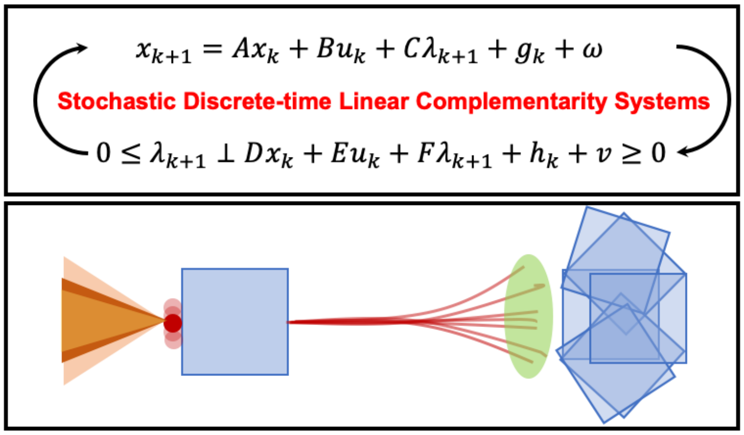

$x_{k+1} = Ax_k + Bu_k + C\lambda_{k+1} + g_k + \omega$
Stochastic Discrete-time Linear Complementarity Systems
$0 \leq \lambda_{k+1} \perp Dx_k + Eu_k + F\lambda_{k+1} + h_k + v \geq 0$

Figure 6.1: This chapter presents chance-constrained optimization for SDLCS. The figure shows the case of stochastic planar pushing with uncertain dynamics. Note that w and v are additive uncertainty terms. We show a MIP-based stochastic MPC formulation for control of stochastic planar pushing system.

tion challenging. In some recent works that consider stochastic complementarity constraints, an expected residual minimization (ERM)-based [42] penalty is used to solve the robust optimization problem [2]. A major shortcoming of such an approach is that it fails to capture the stochastic evolution of system dynamics due to the stochastic complementarity constraint. Similarly, in [3], the authors augment the formulation in [2] with chance constraints. However, this formulation has certain fundamental shortcomings which prevent constraint satisfaction guarantees. One should notice that uncertainty naturally leads to stochastic evolution of system states in SDLCS. A robust optimization formulation for SDLCS should consider the uncertainty in state evolution. Motivated by these problems and weaknesses, we present a formulation that circumvents these shortcomings by using a mixed integer formulation. Using a relaxation of the complementarity constraints, we formulate the chance-constrained optimization for SDLCS as a Mixed Integer Quadratic Program with Chance Constraints (MIQPCC).

Since worst-case robust optimization is quite conservative and does not explicitly discuss stochastic evolution of states [37], this work considers probabilistic optimization with stochastic evolution of states. We illustrate some challenges in performing principled stochastic optimization for SDLCS. We introduce some simplifying assumptions which are important in order to formulate a tractable optimization problem. In particular, we consider the case where the coefficient matrices multiplying the complementarity variables are stochastic while assuming that the complementarity variables are deterministic. This corresponds to the case when one might have uncertainty arising from errors in parameter identification leading to a SDLCS. An alternative to this, and a more accurate formulation, is to allow the complementarity variables to also be stochastic. However, such treatment is out of the scope of the current work. Our treatment of SDLCS leads to stochastic evolution of system states, while we treat the complementarity variables as deterministic. The assumption of determinacy in complementarity variables is similar to several previous works [42], [2], [3], [41]. Robustness to uncertainty is provided by enforcing probabilistic satisfaction of state constraints. Under certain simplifications, we show that the chance-constrained problem can be reformulated as a MIQPCC.

Contributions. This chapter has the following contributions:

1. We present a novel formulation for chance-constrained optimization of SDLCS.
2. The proposed optimization is used in a stochastic MPC method for control of stochastic nonlinear complementarity systems.
3. We compare our proposed approach with several previously proposed techniques and demonstrate that our method outperforms the recent techniques in [2, 3].
4. We present a formulation for performing stochastic MPC for stochastic complementarity systems using the proposed formulation. The proposed MPC formulation is verified using a stochastic planar pushing system.

6.2 Problem Preliminary

For completeness of the chapter, we first provide a brief introduction to linear complementarity problems and their stochastic form. Then the problem formulation for robust trajectory optimization of stochastic linear complementarity systems is provided. We also point out several key differences of our approach w.r.t. previous attempts for robust trajectory optimization of stochastic complementarity system.

6.2.1 Discrete-time Linear Complementarity System (DLCS)

A DLCS is a discrete-time linear dynamical system with complementarity constraints [138] represented by:

$$x_{k+1} = Ax_k + Bu_k + C\lambda_{k+1} + g_k \tag{6.1a}$$

$$0 \leq \lambda_{k+1} \perp Dx_k + Eu_k + F\lambda_{k+1} + h_k \geq 0 \tag{6.1b}$$

where k is the time-step index, $x_k \in \mathbb{R}^{n_x}$ is the state, $u_k \in \mathbb{R}^{n_u}$ is the control input, and $\lambda_k \in \mathbb{R}^{n_c}$ is the algebraic variable (e.g., contact forces). The matrices, A, B, C, D, E, F and vectors g_k, h_k are of compatible dimensions. The i-th element of vector p_k (p_k can be x_k, u_k, λ_k) is represented as $p_{k,i}$. The i-th diagonal element of matrix P_k is represented as $P_{k,ii}$. The notation $0 \leq a \perp b \geq 0$ denotes the complementarity constraints $a \geq 0, b \geq 0, ab = 0$. The variables a and b are known as complementarity variables. These variables could be decision variables or functions of decision variables.

Given a pair of state and control values x_k, u_k, a unique solution λ_{k+1} to (6.1b) exists if F is P-matrix [139]. The matrix F is said to be a P-matrix if every principal minor of F is positive. If F does not satisfy the P-matrix property, it is possible that λ_{k+1} satisfying (6.1b) is non-unique or non-existent.

6.2.2 Contact-Implicit Trajectory Optimization

Trajectory optimization for the DLCS (6.1) can be formulated as:

$$\min_{x,u,\lambda} \sum_{k=0}^{N-1} J(x_k, u_k, \lambda_k) \tag{6.2a}$$

$$\text{s.t. } x_{k+1} = Ax_k + Bu_k + C\lambda_{k+1} + g_k, \tag{6.2b}$$

$$0 \leq \lambda_{k+1} \perp Dx_k + Eu_k + F\lambda_{k+1} + h_k \geq 0, \tag{6.2c}$$

$$x_0 = x_s, x_N = x_g, x_k \in \mathcal{X}, u_k \in \mathcal{U}, \lambda_k \leq \lambda_u \tag{6.2d}$$

where x_s, x_g represent the initial and the terminal values, respectively, $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ and $\mathcal{U} \subseteq \mathbb{R}^{n_u}$ are convex polytopes consisting of a finite number of linear inequality constraints, λ_u is the upper bound of λ_k, and N is the time horizon. This approach is widely known as contact-implicit trajectory optimization in locmotion and manipulation literature [7, 16].

While (6.2) is widely used in various robotic applications (see [133, 7]), it can be fragile under uncertainty, which is often the case in model-based manipulation. It is desirable to consider a robust version of the optimization problem in (6.2). However, the non-differentiability of complementarity constraints pose unique challenges for uncertainty propagation. We present a novel, stochastic version of (6.2) so that the resulting trajectory would be robust under uncertainty.

6.2.3 Stochastic Discrete-time Linear Complementarity Systems (SDLCS)

We consider the following SDLCS:

$$x_{k+1} = Ax_k + Bu_k + C\lambda_{k+1} + g_k + w_k \tag{6.3a}$$

$$0 \leq \lambda_{k+1} \perp y_{k+1} \geq 0 \tag{6.3b}$$

where $y_{k+1} = Dx_k + Eu_k + F\lambda_{k+1} + h_k + v_k$, and $w_k \in \mathbb{R}^{n_x}, v_k \in \mathbb{R}^{n_c}$ are known additive uncertainties. The variables y_{k+1} and λ_{k+1} are the complementarity variables. One should notice that SDLCS would lead to stochastic evolution of states as well as the complementarity variables. In fact, in a recent publication [44], authors have shown the stochastic evolution of SDLCS. However, the resulting distribution of the complementarity variables is quite complex, which makes uncertainty propagation for SDLCS quite challenging. Thus, we consider the case where the coefficient matrix C in (6.3a) and F in (6.3b) are stochastic while the complementarity variables are deterministic. This is used as an alternative approach to model the stochastic effect due to complementarity constraints while admitting a tractable computational approach. Our treatment of SDLCS leads to stochastic evolution of system states x_k, while we treat λ_{k+1} as deterministic. While this assumption is limiting, we show that we can compute robust trajectories for the underlying system.

The authors in [2] use ERM to solve robust TO of SDLCS with the following cost function:

$$\sum_{k=0}^{N-1}\left(J\left(x_k, u_k, \lambda_{k+1}\right)+\beta \mathbb{E}\left[\left\|\psi\left(\lambda_{k+1}, y_{k+1}\right)\right\|^2\right]\right) \tag{6.4}$$

where ψ is a Nonlinear Complementarity Problem (NCP) function and β is a weighting scalar. The NCP function $\psi(\lambda_{k+1}, y_{k+1})$ has the property that $\psi(\lambda_{k+1}, y_{k+1}) = 0$ if and only if the complementarity constraints (6.3b) hold. An example such a function is $\min(\lambda_{k+1}, y_{k+1})$, where the minimum is applied componentwise. We compare the robustness of our formulation with (6.4) in Section 6.4.

6.3 Robust Trajectory Optimization for SDLCS

In this section, we present our formulation for robust trajectory optimization of SDLCS. We consider the following optimization problem:

$$\min_{x,u,\lambda} \mathbb{E}\left[\sum_{k=0}^{N-1} J(x_k, u_k, \lambda_k)\right] \tag{6.5a}$$

$$\text{s.t. } x_{k+1} = Ax_k + Bu_k + C\lambda_{k+1} + g_k + w_k, \tag{6.5b}$$

$$\Pr\left(0 \leq \lambda_{k+1} \perp y_{k+1} \geq 0, x_k \in \mathcal{X}, \forall k\right) \geq 1-\Delta, \tag{6.5c}$$

$$x_0 \sim \mathcal{N}\left(x_s, \Sigma_s\right), u_k \in \mathcal{U}, \lambda_k \leq \lambda_u \tag{6.5d}$$

where $\Pr(\cdot)$ denotes the probability associated with an event and $\Delta \in (0, 0.5]$ is the user-defined maximum probability of violating the constraints. (6.5c) is the joint chance constraint for the state to lie in the desired set as well as the complementarity constraints at every instant of time. The quantities x_s, Σ_s are the mean and covariance matrix of the state at $k = 0$ respectively . $\mathcal{X}$ and $\mathcal{U}$ are convex polytopes, consisting of a finite number of linear inequality constraints. We make the following simplifying assumptions:

1. The noise terms w_k, v_k follow a Gaussian distribution.
2. The complementarity variable λ_{k+1} is deterministic.
3. Each element of vectors $C\lambda_{k+1}$ and $F\lambda_{k+1}$ are independent Gaussian variables.

Problem (6.5) might be intractable with this formulation of the constraints (6.5c). Therefore, in Section 6.3.1, we propose how to convert (6.5c) in order to obtain a tractable optimization problem. In Section 6.3.2 we explain the rationale for the above assumptions as a simplification in order to solve (6.5) .

We explain the reasoning behind our formulation presented in (6.5). Since the underlying SDLCS is uncertain, we consider a chance-constrained formulation for optimization to capture stochastic evolution of states where we impose multiple constraints simultaneously. This is represented as joint chance constraints for the complementarity constraints as well as the states, which is succintly written in Equation (6.5c). Note that we represent the chance constraints on all the variables jointly (as is common in stochastic optimization for dynamic systems) using the cumulative distribution function (cdf) for the state as well as complementarity variables. In the remainder of this section, we will show how the joint constraints can be decomposed into individual chance constraints using Boole's inequality. It is also important to note that unlike the formulation in (6.5), the method in [2, 3, 140] fails to capture the stochastic evolution of states in their formulation.

6.3.1 Joint Linear Chance Constraints

We consider joint chance constraint such that multiple constraints are satisfied simultaneously with a pre-specified probability. More specifically, we consider the joint chance constraints (6.5c) so that the complementarity constraints and state bound constraints over the whole time horizon of the optimized trajectory are satisfied with probability $(1-\Delta)$. We succintly denote the complementarity relationship in (6.3b) as $(\lambda_{k+1,i}, y_{k+1,i}) \in \mathcal{S}$ for $i = 1, \ldots, n_c$, i.e. $(\lambda_{k+1,i}, y_{k+1,i})$ satisfies (6.3b) if and only if $(\lambda_{k+1,i}, y_{k+1,i}) \in \mathcal{S}$. Hence, we can rewrite the joint chance constraints in the optimization problem (6.5) as:

$$\begin{aligned}&\Pr\left(0 \leq \lambda_{k+1} \perp y_{k+1} \geq 0, x_k \in \mathcal{X}, \forall k\right) \geq 1-\Delta \iff \\ &\Pr\left(\bigwedge_{k=0}^{N}\left(\bigwedge_{i=1}^{n_c}(\lambda_{k+1,i}, y_{k+1,i}) \in \mathcal{S}\right) \bigwedge \left(\bigwedge_{l=1}^{L} a_l^\top x_k \leq b_l\right)\right) \\ &\geq 1-\Delta\end{aligned} \tag{6.6}$$

where $\bigwedge$ is the logical AND operator. The parameter L represents the number of inequalities modeling $\mathcal{X}$ and $a_l \in \mathbb{R}^{n_x}$ is the coefficient vector and b_l represents the right-hand side of the inequality.

Obtaining a cumulative density function (cdf) of (6.6) is challenging because the joint probability of states and complementarity variables is considered. A popular approach to decompose joint chance constraints is the application of Boole's inequality [43] which converts the original computationally intractable joint chance constraints into conservative but tractable independent chance constraints. Hence, similar to previous works, we use Boole's inequality [43] to get the conservative approximation of (6.6) as follows:

$$\begin{aligned}\Pr\left(\bigwedge_{k=0}^{N}\left(\bigwedge_{i=1}^{n_c}(\lambda_{k+1,i}, y_{k+1,i}) \in \mathcal{S}\right)\right) \geq 1-\Delta_1, \\ \Pr\left(\bigwedge_{k=0}^{N}\left(\bigwedge_{l=1}^{L} a_l^\top x_k \leq b_l\right)\right) \geq 1-\Delta_2, \Delta_1 = \Delta_2 = \frac{\Delta}{2}\end{aligned} \tag{6.7}$$

Using Boole's inequality again, we can further obtain the conservative chance constraints given by:

$$\Pr\left((\lambda_{k+1,i}, y_{k+1,i}) \in \mathcal{S}\right) \geq 1 - \frac{\Delta_1}{N n_c}, \tag{6.8a}$$

$$\Pr\left(a_l^\top x_k \leq b_l\right) \geq 1 - \frac{\Delta_2}{NL}, \tag{6.8b}$$

We discuss how to handle (6.8a) in Section 6.3.2. We formulate (6.8b) as its equivalent deterministic form (see [141, 142, 3]):

$$\Pr\left(a_l^\top x_k \leq b_l\right) \geq 1 - \frac{\Delta_2}{NL} \Longleftrightarrow \tag{6.9a}$$

$$a_l^\top \bar{x}_k \leq b_l - \sqrt{a_l^\top \Sigma_{x_k} a_l}\,\Phi^{-1}(1 - \frac{\Delta_2}{NL}) \tag{6.9b}$$

where $\bar{x}_k, \Sigma_{x_k}$ are the mean and covariance matrix of x_k, respectively. Φ^{-1} is an inverse of the cdf of the standard normal distribution.

6.3.2 Chance Complementarity Constraints (CCC) for SDLCS

We explain the rationale behind some of the assumptions specified in Section 6.3. One of the key assumptions is that the complementarity variable λ_{k+1} is deterministic. A more general formulation would allow the complementarity variable λ_{k+1} to be stochastic. Indeed, in our previous work [44], we have shown that the complementarity variable λ_{k+1} is in fact stochastic. One should notice, however, that the complementarity constraints naturally leads to truncated distribution of the complementarity variables (as $\lambda_{k+1} \geq 0$). This makes uncertainty propagation for complementarity variables challenging. This could potentially make the stochastic optimization problem computationally challenging (see [44]). Thus, in this work, we make the assumption that λ_{k+1} is deterministic, which improves the computational requirements for the resulting optimization problem. Furthermore, we believe that allowing C and F to be stochastic can model a similar effect to having λ_{k+1} stochastic in the SDLCS. Finally, note that in cases where the distribution of λ_{k+1} is known, our proposed formulation can be easily extended to incorporate stochasticity in λ_{k+1}. However, for brevity, we skip these details. Assuming the uncertainty to be Gaussian is motivated by our interest in leveraging the equivalent reformulation of the chance constraints to deterministic inequalities.

While [3] proposed a promising CCC, their formulation possesses empty solutions when $\Delta \leq 0.5$ (see [3]). The formulation in [3] can result in a very fragile trajectory since the total violation probability over N steps would be always more than 1 if $N \geq 1$ (using Boole's

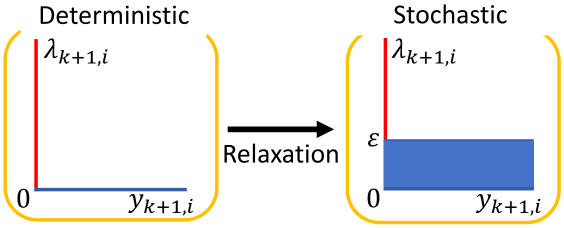


Figure 6.2: Deterministic and stochastic complementarity constraints. We have the complementarity constraints $0 \leq \lambda_{k+1,i} \perp y_{k+1,i} \geq 0$ where $y_{k+1,i}$ has uncertainty and accepts the violation of ϵ.

inequality). Since the optimization is formulated as a Non-Linear Program (NLP), all the CCC are imposed simultaneously. Consequently, the resulting formulation struggles to find feasible solutions.

In our formulation, the stochastic complementarity constraints are decomposed into two modes (see Fig. 6.2) as follows:

$$\Pr\left((\lambda_{k+1,i}, y_{k+1,i}) \in \mathcal{S}\right) \geq 1-\theta \tag{6.10a}$$

$$\iff \Pr\begin{pmatrix} (\lambda_{k+1,i} \geq 0, y_{k+1,i} = 0) \\ \bigvee (\lambda_{k+1,i} = 0, y_{k+1,i} \geq 0) \end{pmatrix} \geq 1-\theta \tag{6.10b}$$

$$\iff \begin{cases} \lambda_{k+1,i} \geq 0, \Pr\left(y_{k+1,i} = 0\right) \geq 1-\theta \\ \text{or } \lambda_{k+1,i} = 0, \Pr\left(y_{k+1,i} \geq 0\right) \geq 1-\theta \end{cases} \tag{6.10c}$$

where $\theta = \frac{\Delta_1}{N n_c}$ and $\bigvee$ denotes the logical OR. Note that now $y_{k+1} \sim \mathcal{N}\left(\bar{y}_{k+1}, \Sigma_{y_{k+1}}\right)$. To obtain lower violation probabilities, we propose an Mixed Integer Programming (MIP) framework. First, we propose the following CCC:

$$z_{k,i,0} = 1, \Longrightarrow \lambda_{k+1,i} \geq 0, \Pr\left(y_{k+1,i} = 0\right) \geq 1-\theta, \tag{6.11a}$$

$$z_{k,i,1} = 1, \Longrightarrow \lambda_{k+1,i} = 0, \Pr\left(y_{k+1,i} \geq 0\right) \geq 1-\theta \tag{6.11b}$$

where $z_{k,i,0}, z_{k,i,1}$ denote the binary variables to represent the two modes which satisfies $z_{k,i,0} + z_{k,i,1} = 1$ for i-th complementarity constraint at instant k.

However, $\Pr\left(y_{k+1,i} = 0\right)$ is zero (as the probability measure for singleton sets is zero) so that we cannot directly use (6.11). To alleviate this issue while avoiding negative values for

λ, we propose the following CCC using a relaxation for complementarity constraints (see Fig. 6.2):

$$z_{k,i,0} = 1, \Longrightarrow \lambda_{k+1,i} \geq 0, \Pr\left(0 \leq y_{k+1,i} \leq \epsilon\right) \geq 1 - \theta, \tag{6.12a}$$

$$z_{k,i,1} = 1, \Longrightarrow \lambda_{k+1,i} = 0, \Pr\left(y_{k+1,i} \geq \epsilon\right) \geq 1 - \theta \tag{6.12b}$$

where $\epsilon > 0$ is the acceptable violation in the complementarity constraints.

We have two-sided linear chance constraints in (6.12a). We decompose (6.12a) as two one-sided chance constraints so that we can use the same reformulation as in (6.9). Note that each one-sided chance constraints, obtained from the two-sided chance constraint, are formulated with a maximum violation probability of $\frac{\theta}{2}$.

Since we have integer constraints, MIP can impose individual constraints for each mode separately. This allows to derive a lower bound for θ as function of ϵ, $\bar{y}_{k+1,i}$, and $\Sigma_{y_{k+1},ii}$, which is presented in Lemma 6.3.1. On the other hand, the NLP formulation imposes all mode constraints jointly (see [3]). Consequently, the NLP formulation achieves a higher bound for θ. We provide arguments describing the advantage of our approach over the NLP approach in Remark 1.

Lemma 6.3.1. *Suppose the CCC are formulated as (6.12) and* ϵ, $\bar{y}_{k+1,i}$, *and* $\Sigma_{y_{k+1},ii}$ *are specified. Then (i) (6.12a) is feasible for all* $\theta > 2(1 - \Phi(\frac{\epsilon}{2\Sigma_{y_{k+1},ii}}))$ *and (ii) (6.12b) is feasible for all* $\theta > 1 - \Phi\left((\bar{y}_{k+1,i} - \epsilon)/\Sigma_{y_{k+1}ii}\right)$.

Proof. Consider case (i): From (6.9b) and (6.12a), the two-side chance constraints in (6.12a) are converted to their deterministic forms which are given as: $\Sigma_{y_{k+1},ii}\Phi^{-1}\left(1 - \theta/2\right) \leq \bar{y}_{k+1,i} \leq \epsilon - \Sigma_{y_{k+1},ii}\Phi^{-1}\left(1 - \theta/2\right)$. To have a nonempty solution, we must have $\epsilon - 2\Sigma_{y_{k+1},ii}\Phi^{-1}\left(1 - \theta/2\right) > 0$. Simplifying this equation, we obtain the bound specified in (i). Consider case (ii): From (6.9b) and (6.12b), the one-side chance constraints in (6.12b) are converted as: $\bar{y}_{k+1,i} \geq \epsilon + \Sigma_{y_{k+1},ii}\Phi^{-1}\left(1 - \theta\right)$. Simplifying this equation, we obtain the bound specified in (ii). □

Remark 6.3.1. *From Lemma 6.3.1, it is easy to show that* $\theta < \frac{1}{2}$ *if* $\frac{\epsilon}{2\Sigma_{y_{k+1},ii}} > \Phi^{-1}(\frac{3}{4})$ *for case (i), and if* $(\bar{y}_{k+1,i} - \epsilon)/\Sigma_{y_{k+1}ii} > \Phi^{-1}(\frac{1}{2})$ *for case (ii). In contrast, the formulation in [3] cannot enforce the chance constraints for any* $\theta < 0.5$.

The evolution of the mean of the states in SDLCS is described by the following equations:

$$\bar{x}_{k+1} = A\bar{x}_k + Bu_k + \overline{C\lambda}_{k+1} + g_k + \bar{w}_k, \tag{6.13a}$$

$$\Sigma_{x_{k+1}} = A\Sigma_{x_k}A^\top + \Sigma_{C\lambda_{k+1}} + W \tag{6.13b}$$

where W represents the noise covariance matrix and $\overline{C\lambda}_{k+1}$ represents a mean of $C\lambda_{k+1}$, and $\Sigma_{C\lambda_{k+1}} = \mathbb{E}\left[\left(C\lambda_{k+1} - \overline{C\lambda}_{k+1}\right)\left(C\lambda_{k+1} - \overline{C\lambda}_{k+1}\right)^\top\right]$ is a diagonal matrix because of the independence of random variables.

The mean and variance of y_k in SDLCS is described by the following equations:

$$\bar{y}_k = D\bar{x}_k + Eu_k + \overline{F\lambda}_{k+1} + h_k + \bar{v}_k, \tag{6.14a}$$

$$\Sigma_{y_k} = D\Sigma_{x_k}D^\top + \Sigma_{F\lambda_{k+1}} + V \tag{6.14b}$$

where V represents the noise covariance matrix from v_k and $\overline{F\lambda}_{k+1}$ represents a mean of $F\lambda_{k+1}$, and $\Sigma_{F\lambda_{k+1}} = \mathbb{E}\left[\left(F\lambda_{k+1} - \overline{F\lambda}_{k+1}\right)\left(F\lambda_{k+1} - \overline{F\lambda}_{k+1}\right)^\top\right]$ is a diagonal matrix because of the independence of random variables.

6.3.3 Mixed-Integer Quadratic Programming with Chance Constraints

In this section, we present our MIQPCC formulation to solve (6.5). The proposed CCC could be imposed in either an MIP or an NLP framework. However, our MIP-based method solves disjunctive inequalities while NLP needs to impose all CCC simultaneously, which yields an empty solution for $\Delta \leq 0.5$. For this reason we do not consider an NLP framework.

The proposed MIQPCC is formulated as follows:

$$\min_{x,u,\lambda,z} \sum_{k=0}^{N-1} \bar{x}_k^\top Q \bar{x}_k + u_k^\top R u_k \tag{6.15a}$$

$$\text{s. t. } \bar{x}_{k+1} = A\bar{x}_k + Bu_k + \overline{C\lambda}_{k+1} + g_k + \bar{w}_k, \tag{6.15b}$$

$$\Sigma_{x_{k+1}} = A\Sigma_{x_k}A^\top + \Sigma_{w,C\lambda_{k+1}} + W, \tag{6.15c}$$

$$x_0 \sim \mathcal{N}\left(x_s, \Sigma_s\right), u_k \in \mathcal{U}, \lambda_k \leq \lambda_u, \tag{6.15d}$$

$$a_l^\top \bar{x}_k \leq b_l - \alpha\kappa, \tag{6.15e}$$

$$z_{k,i,0} + z_{k,i,1} = 1, \tag{6.15f}$$

$$0 \leq \lambda_{k+1,i} \leq M z_{k,i,0}, \tag{6.15g}$$

$$\zeta\psi z_{i,k,0} + (\epsilon + \eta\psi) z_{k,i,1} \leq \bar{y}_{k+1,i} \tag{6.15h}$$

$$\bar{y}_{k+1,i} \leq (\epsilon - \zeta\psi) z_{k,i,0} + M z_{k,i,1}, \tag{6.15i}$$

where $Q = Q^\top \geq 0, R = R^\top > 0$, $\alpha = \Phi^{-1}(1 - \frac{\Delta}{2NL}), \zeta = \Phi^{-1}(1 - \frac{\Delta}{4Nn_c}), \eta = \Phi^{-1}(1 - \frac{\Delta}{2Nn_c}), \kappa = \sqrt{a_l^\top \Sigma_{x_k} a_l}, \psi = \sqrt{\Sigma_{y_{k+1,ii}}}$. $z_{k,i,0}, z_{k,i,1}$ are the binary decision variables for the i-th complementarity constraint at k to represent mode 1, 2, respectively. Using these binary variables, we employ big-M formulation to deal with disjunctive inequalities in our CCC. The parameter M is a valid upper bound for λ_k, y_k.

6.3.4 Stochastic Model Predictive Control with Complementarity Constraints

MPC is very popular and well understood for control of smooth dynamical systems. However, it remains mostly unexplored for complementarity systems and stochastic complementarity systems. Using our proposed MIQPCC, we present a stochastic MPC method for uncertain contact systems. More formally, we present a formulation to implement SMPC for stochastic non-linear complementarity system (SNCS) where the dynamics equation is represented as:

$$x_{k+1} = f(x_k, u_k, \lambda_{k+1}) + w_k \tag{6.16a}$$

$$0 \leq \lambda_{k+1} \perp y_{k+1} \geq 0 \tag{6.16b}$$

where $f(x_k, u_k, \lambda_{k+1})$ is nonlinear dynamics. The goal is to find a control sequence to track a reference state trajectory for the SNCS. We first create the reference trajectory x^*, u^*, λ^*, y^* by solving the optimization with deterministic complementarity constraints (no chance constraints) using Mathematical Program with Complementarity Constraints (MPCC) [16]. Then, we linearize the dynamics along the reference trajectory which is used for uncertainty propagation.

The modified MIQPCC for SMPC is given by:

$$\min_{x,u,\lambda,z} x_N^{e\top} Q_N x_N^e + \sum_{k=0}^{N-1} x_k^{e\top} Q x_k^e + u_k^{e\top} R u_k^e \tag{6.17a}$$

$$\text{s.t.} x_{k+1}^e = A x_k^e + B u_k^e + C {\lambda^e}_{k+1} + \bar{w}_k, \tag{6.17b}$$

$$\text{(6.15c), (6.15e), (6.15f),} \tag{6.17c}$$

$$x_0 \sim \mathcal{N}(x_s, \Sigma_s), u_k^e + u_k^* \in \mathcal{U}, \lambda_k^e + \lambda_k^* \leq \lambda_u, \tag{6.17d}$$

$$0 \leq \lambda_{k+1,i}^e + \lambda_{k+1,i}^* \leq M z_{k,i,0} \tag{6.17e}$$

$$\zeta\psi z_{i,k,0} + (\epsilon + \eta\psi) z_{k,i,1} \leq y_{k+1,i}^e + y_{k+1,i}^* \tag{6.17f}$$

$$y_{k+1,i}^e + y_{k+1,i}^* \leq (\epsilon - \zeta\psi) z_{k,i,0} + M z_{k,i,1} \tag{6.17g}$$

where $x^e = \bar{x} - x^*, u^e = u - u^*, \lambda^e = \lambda - \lambda^*, y^e = \bar{y} - y^*, A = \left.\frac{\partial f(x,u,\lambda)}{\partial x}\right|_{x^*,u^*,\lambda^*}, B = \left.\frac{\partial f(x,u,\lambda)}{\partial u}\right|_{x^*,u^*,\lambda^*}, C = \left.\frac{\partial f(x,u,\lambda)}{\partial \lambda}\right|_{x^*,u^*,\lambda^*}$. It is worth pointing out that this formulation does not fix or penalize the discrete mode sequence. (6.17e) means that $\lambda_{k+1,i} \geq 0$ if $z_{k,i,0} = 1$ and $\lambda_{k+1,i} = 0$ if $z_{k,i,1} = 1$. Thus, (6.17e) allows deviation from the reference discrete mode sequence while satisfying complemenatarity constraints. Therefore, the controller may change the mode sequence from the reference. In prior work, Hogan et al. [143] proposed a similar MIQP formulation. However, they penalize deviation from the reference mode sequence which might be infeasible for a number of cases (due to state and control bounds). Additionally, it does not consider stochastic dynamics and complementarity constraints during control.

6.4 Numerical Simulations

We validate our proposed method using three benchmark DLCS which are shown in Fig. 6.3. See [137] for more details of these three benchmarks. Through the numerical experiments performed in this paper, we answer the following questions:

1. Can our proposed optimization generate robust open-loop trajectories?
2. Can our proposed formulation satisfy the probabilistic constraints imposed during optimization?
3. How does the proposed method compare against the previous methods for robust optimization in SDLCS?

6.4.1 Implementation Details

The proposed method is implemented in Python. We use Gurobi [94] to solve the proposed MIQPCC, and PyRoboCOP [16, 16] to solve the MPCC arising from the ERM-based method in [2] and the CCC method in [3]. The examples are implemented on a computer with Intel i7-12800H processor.

We verify the robustness of open-loop trajectories obtained from our proposed optimization using Monte Carlo simulations. To simulate SDLCS with a given control input, we use MINPACK [144] to solve the nonlinear complementarity problem that arises at each time-step in order to simulate the system. The noise term is sampled from the distribution which was used during optimization. We simulate each control trajectory for 1000 times to estimate the probability of constraint violation for the proposed as well as the baseline methods.

For notation simplicity, we present the continuous-time dynamics for all the test systems. We discretize continuous-time dynamics into discrete-time dynamics using the explicit Euler method with sample time $dt = 0.033$. We denote x_0, Σ_0 as the mean and covariance matrix at $k = 0$ for states of systems, respectively.

6.4.2 Example Details

6.4.2.1 Cartpole with Softwalls

The continuous-time dynamics with complementarity constraints for the cartpole with softwalls (see Fig. 6.3a) is as follows:

$$\dot{x}_1 = x_3, \dot{x}_2 = x_4, \dot{x}_3 = g\frac{m_p}{m_c}x_2 + \frac{1}{m_c}u_1, \tag{6.18a}$$

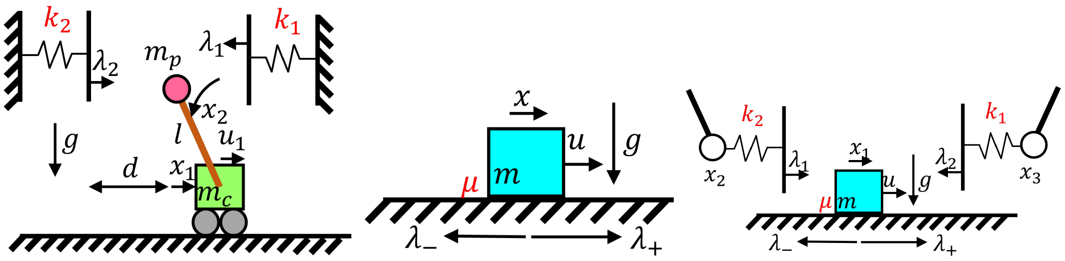
$$\dot{x}_4 = \frac{g\left(m_c + m_p\right)}{lm_c}x_2 + \frac{1}{lm_c}u_1 + \frac{\lambda_1}{lm_p} - \frac{\lambda_2}{lm_p}, \tag{6.18b}$$

$$0 \leq \lambda_1 \perp lx_2 - x_1 + \frac{1}{k_1}\lambda_1 + d \geq 0, \tag{6.18c}$$

$$0 \leq \lambda_2 \perp x_1 - lx_2 + \frac{1}{k_2}\lambda_2 + d \geq 0 \tag{6.18d}$$

where x_1 is the cart position, x_2 is the pole angle, the x_3 and x_4 are their derivatives. u_1 is the control and λ_1, λ_2 are the reaction forces at from the wall 1, 2, respectively. We consider the additive noise w, the zero-mean i.i.d. Gaussian noise which standard deviation is 2×10^{-4}, to $x_{1,k}, x_{2,k}$. $k_1 = 10, k_2 = 10$ are the stiffness of walls 1 and 2, respectively. In this example, we assume that the uncertainty also arises from the $\frac{1}{k_1}, \frac{1}{k_2}$ which standard deviations are 10^{-5}. We denote by $g = 9.81$ is the gravitational acceleration, and by $m_p = 0.1, m_c = 1.0$ denote the mass of the pole, cart, respectively. Further, $l = 0.5$ is the length of the pole and $d = 0.15$ is the distance from the origin of the coordinate to the walls.

The optimization setup is as follows: $N = 20, M = 100, Q = \mathrm{diag}(0,0,0,0), R = 0.01$, $\epsilon = 0.002$, $x_0 = [-0.15, 0, 0, 0]^\top, \Sigma_0 = \mathrm{diag}(0,0,0,0)$. We also impose the following chance constraints: $\Pr(x_{1,k} \leq 0.05) \geq 1 - \frac{\Delta}{4N}, \Pr(x_{2,k} \leq 0.15) \geq 1 - \frac{\Delta}{4N}, \forall k = 0, \ldots, N-1$, $\Pr(-0.02 \leq x_{1,N} \leq 0.02) \geq 1 - \frac{\Delta}{4N}, \Pr(-0.04 \leq x_{2,N} \leq 0.04) \geq 1 - \frac{\Delta}{4N}$.



(a) Cartpole with softwalls (b) Sliding box (c) Dual manipulation.

Figure 6.3: Problems described in Section 6.4.

6.4.2.2 Sliding Box with Friction

The continuous-time quasi-static dynamics with complementarity constraints for sliding box with Coulomb friction (see Fig. 6.3b) is as follows:

$$\dot{x}_1 = x_2, \alpha \dot{x}_1 = u + \lambda_+ - \lambda_-, \tag{6.19a}$$

$$0 \leq \gamma \perp \mu m g - \lambda_+ - \lambda_- \geq 0, \tag{6.19b}$$

$$0 \leq \lambda_+ \perp \gamma + u + \lambda_+ - \lambda_- \geq 0, \tag{6.19c}$$

$$0 \leq \lambda_- \perp \gamma - u - \lambda_+ + \lambda_- \geq 0 \tag{6.19d}$$

where x_1 is the box position and x_2 is the box velocity. u is the control input and λ_+, λ_- are the positive and negative components of the friction force, respectively. γ is the slack variable. $\alpha = 4$ is the damping constant, $m = 1$ is the mass of the box, and $\mu = 0.1$ is the coefficient of friction. We consider additive i.i.d. Gaussian noise w in the dynamics equation as $x_{1,k+1} = x_{1,k} + x_{2,k} dt + w$. The standard deviation of w is 4×10^{-4}. $g = 9.81$ is the gravitational acceleration. We assume that the coefficient of friction, μ, is also uncertain and standard deviation for μ is 10^{-5}.

The optimization setup is as follows: $N = 20, M = 100, Q = \mathrm{diag}(0,0,0,0), R = 0.01$, $\epsilon = 0.01, x_0 = [1, -1]^\top, \Sigma_0 = \mathrm{diag}(0,0)$. We also impose the following chance constraints: $\Pr(x_{1,k} \geq 0.885) \geq 1 - \frac{\Delta}{2N}, \forall k = 0, \ldots, N-1$, $\Pr(0.89 \leq x_{1,N} \leq 0.91) \geq 1 - \frac{\Delta}{2N}$, $\Pr(-0.1 \leq x_{2,N} \leq 0.1) \geq 1 - \frac{\Delta}{2N}$.

6.4.2.3 Dual Manipulator System

We consider the example where a box is manipulated on a planar surface with Coulomb friction and contact forces from two manipulators (see Fig. 6.3c). The continuous-time quasi-static dynamics is as follows:

$$\begin{aligned}
&\dot{x}_1 = x_2, \alpha\dot{x}_1 = \lambda_1 - \lambda_2 + \lambda_+ - \lambda_-, \\
&\dot{x}_3 = x_4, \dot{x}_4 = u_1, \dot{x}_5 = x_6, \dot{x}_6 = u_2, \\
&0 \leq \lambda_1 \perp x_1 - x_3 + \frac{1}{k}\lambda_1 \geq 0, \\
&0 \leq \lambda_2 \perp x_5 - x_1 + \frac{1}{k}\lambda_2 \geq 0, \\
&0 \leq \gamma \perp \mu m g - \lambda_+ - \lambda_- \geq 0, \\
&0 \leq \lambda_+ \perp \gamma + \lambda_1 - \lambda_2 + \lambda_+ - \lambda_- \geq 0, \\
&0 \leq \lambda_- \perp \gamma - \lambda_1 + \lambda_2 - \lambda_+ + \lambda_- \geq 0
\end{aligned} \tag{6.20}$$

x_1, x_3, x_5 are the positions of the box, the left arm, the right arm, respectively and x_2, x_4, x_6 are their derivatives. u_1, u_2 represent the controls of the left and the right arm, respectively. λ_+, λ_- are the positive and negative component of the friction force, respectively. γ is the slack variable. λ_1, λ_2 are the contact forces from the left arm and the right arm, respectively. We set $g = 9.81$, $m = 1$, $k = 100$, $\mu = 0.1$. We discretize the dynamics (6.20) with $dt = 0.033$. As in the previous systems, we the zero-mean i.i.d. Gaussian noise w with standard deviation 0.0002 to the dynamics, $x_{1,k+1} = x_{1,k} + x_{2,k}dt + w$. The standard deviation of μ and $\frac{1}{k}$ are 0.0001.

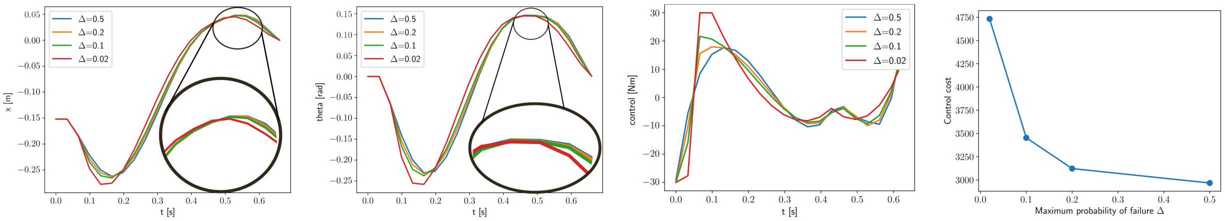
The optimization setup in this example is as follows: $Q = \text{diag}(0, 0, 0, 0, 0, 0), R = \text{diag}(1, 1), N = 20, M = 50, \epsilon = 0.0042$, $x_0 = [0.1, -1.1, 0, 0, 0.1, 0]^\top$, $\Sigma_0 = \text{diag}(0, 0, 0, 0, 0, 0)$. We impose the following chance constraints: $\Pr(x_{1,k} \geq -0.17) \geq 1 - \frac{\Delta}{2N}, \forall k = 0, \ldots, N-1$, $\Pr(-0.01 \leq x_{1,N} \leq 0.01) \geq 1 - \frac{\Delta}{2N}$.

Remark 6.4.1. *It is noted that the proposed method can be used for robust optimization as long as the dynamics is linear. In the presence of non-linear dynamics, uncertainty propagation becomes more challenging and it can not be modeled by the current framework. The* *uncertainty in SDLCS can arise from various sources like parametric uncertainty (e.g., coefficient of friction, uncertain stiffness coefficients, etc.). As long as the underlying dynamics can be modeled as a SDLCS, the proposed formulation could be used for robust optimization.*

6.4.3 Robustness of Open-Loop Trajectories

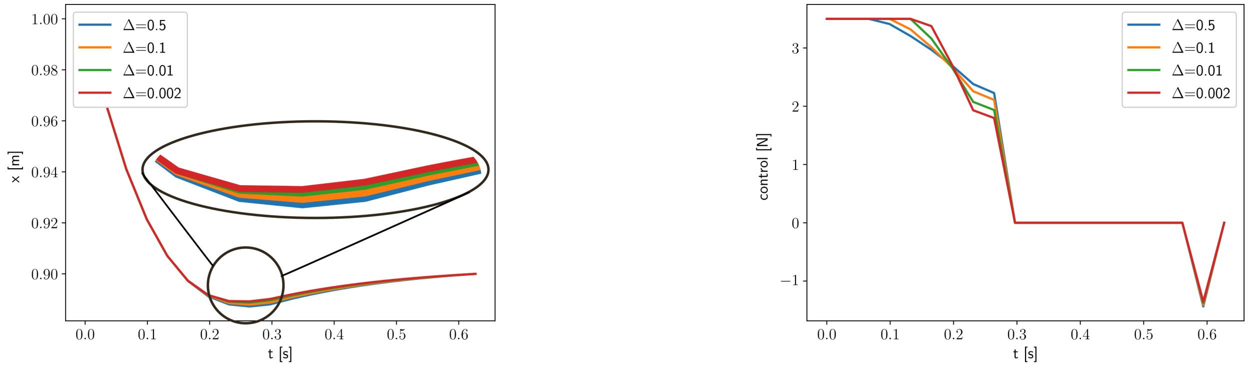
The optimized control and state trajectories for the three systems using our proposed method are shown in Fig. 6.4-Fig. 6.6. Overall, these figures show that the optimal state trajectories move further away from the bound specified in the chance constraints as the violation probability decreases (see the state constraints specified in Section 6.4.2). For instance, Fig. 6.4a shows that the computed trajectories move further away from $x = 0.05$ as Δ decreases (note that $\Pr(x_{1,k} \leq 0.05) \geq 1 - \frac{\Delta}{4N}$ is the chance constraint specified for optimization, see Section 6.4.2). We observe the same behavior for the other systems too in Fig. 6.5 and Fig. 6.6. In addition, these figures illustrate that the control costs increase as Δ decreases. This illustrates the trade-off between safety and cost.

Remark 6.4.2. *At this point, we would like to discuss the magnitude of uncertainty we consider in these problems. Compared to other stochastic optimal control works [145, 141], the uncertainty in these problems is relatively smaller. There are several reasons why we need to have a smaller uncertainty. Note that as we have explained in Section 6.2, our approach satisfies joint constraints on multiple constraints together. First, our formulation has chance complementarity constraints in addition to chance constraints on states, which are commonly used. Our formulation has more number of chance constraints, and consequently, the lower uncertainty is required because of the conservative approximation of Boole's inequality to resolve joint chance constraints into individual constraints as explained in Section 6.3.1, and Section 6.3.2. Second, we need to have a small ϵ to avoid large violation of complementarity constraints, which requires small uncertainty. Finally, we would like to emphasize that allowing larger uncertainties requires either better resolution of joint chance constraints or covariance steering approaches [141, 44], which is out of scope for the current study.*



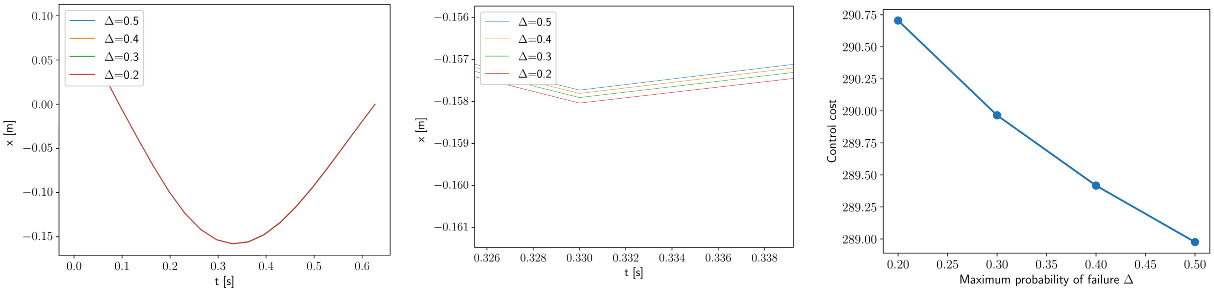
(a) Trajectories of x_1. (b) Trajectories of x_2. (c) Trajectories of u_1. (d) Control cost

Figure 6.4: Results with different Δ for the cartpole with softwalls system. First, the cart moves in the negative direction to utilize the contact force λ_2 because the control input is bounded. Once the cart obtains enough λ_2, the cart is accelerated in the positive direction. We can observe the effect of our proposed chance constraints in particular around $t \in [0, 0.1]$ and $t \in [0.4, 0.5]$. When $t \in [0, 0.1]$, the mode changes from the "contact on the wall 2" to the "no contact" and the cart tries to be far from wall 2 to satisfy the CCC. When $t \in [0.4, 0.5]$, the trajectories are farther away from $x_1 = 0.05$ and $x_2 = 0.15$ as Δ decreases.



(a) Trajectories of x. (b) Trajectories of u.

Figure 6.5: Results with different Δ for the sliding box with friction example. First, the box is accelerated in the positive direction. Then, the control decreases with time to regulate the box around the origin by employing friction forces. We can observe the effect of our proposed chance constraints in particular around $t \in [0.2, 0.3]$ where the trajectories are farther away from $x = 0.88$ as Δ decreases.



(a) Trajectories of x_1.

(b) State x_1 around $t \in [0.2, 0.3]$

(c) Control cost with different Δ.

Figure 6.6: Results with dual manipulation. First, the box is pushed by the right arm in the negative direction. Next, the left arm regulates the box to the origin. In particular, around $t \in [0.2, 0.3]$ s, the trajectories are farther away from $x_1 = -0.17$ m as Δ decreases.

6.4.4 Monte Carlo Simulation Results

Table 6.1-Table 6.3 present the comparison of constraint violation for the proposed method and the baseline methods proposed in [2] and [3] using Monte Carlo simulation. We compute the empirical constraint violation for all the three problems with different values of Δ. We run the ERM method in [2] with different objective weights β and the CCC [3] with violation probability $\Delta_z = 0.5$. The parameter β was chosen so that the magnitude of the ERM cost is of similar order as the other costs in the objective. Since in the original work in [2] and [3], the authors had only considered the terminal constraint violation, we only measure the failure of these methods for violation of the terminal constraint. For the proposed method, we measure the constraint violations which was specified in Section 6.4.2.

Table 6.1 shows that the proposed method outperforms both the baseline methods for the cartpole with softwalls system. The controller based on the ERM method in [2] achieves 100% constraint violation with $\beta = 10^4, 10^5$. With $\beta = 10^3$, the ERM results are comparatively better. One should notice that higher weight of the ERM in the objective function results in more fragile trajectories since the state constraints are always violated. This indicates that the ERM-based objective is not able to capture the robustness in the state trajectories of SDLCS. The CCC in [3] could also show relatively good violation probability compared to the ERM-based method with $\beta = 10^4$ and $\beta = 10^5$ but shows the worse violation probability

Table 6.1: Comparison of the constraint violation probability specified in the different optimization problems against the observed constraint probabililty obtained from simulation of the “cartpole with softwalls” over 1000 samples. In the table, Δ represents the constraint violation probability for our approach, β for the ERM-based approach in [2], and the Δ_z is for the CCC method in [3].

	$\Delta = 0.5$	$\Delta = 0.2$	$\Delta = 0.1$	$\Delta = 0.02$
Obtained Δ (Ours)	0.190	0.147	0.085	0.020

	$\beta = 10^3$	$\beta = 10^4$	$\beta = 10^5$	$\Delta_z = 0.5$
Obtained Δ (ERM)	0.75	1.0	1.0	-
Obtained Δ (CCC)	-	-	-	0.91

Table 6.2: Comparison of the constraint violation probability specified in the different optimization problems against the observed constraint probabililty obtained from simulation of the “a sliding box with friction” over 1000 samples. In the table, Δ represents the constraint violation probability for our approach, β for the ERM-based approach in [2], and the Δ_z is for the CCC method in [3].

	$\Delta = 0.5$	$\Delta = 0.1$	$\Delta = 0.01$	$\Delta = 0.002$
Obtained Δ (Ours)	0.080	0.051	0.027	0.010

	$\beta = 10^3$	$\beta = 10^4$	$\beta = 10^5$	$\Delta_z = 0.5$
Obtained Δ (ERM)	1.0	1.0	1.0	-
Obtained Δ (CCC)	-	-	-	0.91

compared to our method with $\Delta = 0.5$ and the ERM with $\beta = 10^3$.

Table 6.2 shows similar results for the sliding box system for the proposed method. However, we observe that the controller can not satisfy the constraints for the cases with $\Delta = 0.01, 0.002$. There could be several reasons that contribute to the violation of the chance constraints. Unlike the cartpole example, F is not a P matrix for the sliding box system (see Section 6.4.2) so we can get the multiple solutions for the complementarity variable λ. Also, even though ϵ is small, it is not zero so the actual trajectory in the simulator cannot be exactly the same as the trajectory from the optimization even in the absence of noise. While we can ignore these effects with relatively large Δ, we cannot ignore these effects anymore

Table 6.3: Comparison of the constraint violation probability specified in the different optimization problems against the observed constraint probabililty obtained from simulation of the “dual manipulation” over 1000 samples. In the table, Δ represents the constraint violation probability for our approach, β for the ERM-based approach in [2], and the Δ_z is for the CCC method in [3].

	$\Delta = 0.5$	$\Delta = 0.4$	$\Delta = 0.3$	$\Delta = 0.2$
Obtained Δ (Ours)	0.419	0.317	0.257	0.217

	$\beta = 10^3$	$\beta = 10^4$	$\beta = 10^5$	$\Delta_z = 0.5$
Obtained Δ (ERM)	1.0	1.0	1.0	-
Obtained Δ (CCC)	-	-	-	1.0

with the small Δ. We believe some of these effects lead to some constraint violation observed for this system. Although the proposed method could not satisfy chance constraints for all Δ in this example, our method achieves much lower violation probabilities compared to the ERM in [2] and the CCC in [3]. Table 6.3 shows that we obtain similar results for the dual manipulator system as for the sliding-box example.

Fig. 6.7 and Fig. 6.8 show that our proposed planner could successfully drive the system to the goal state. We also observe that with decreasing Δ, the system trajectories move further away from state set boundaries to satisfy tighter chance constraints. For Fig. 6.9, while the majority of the sampled trajectories converge to the specified terminal constraints, some of them clearly converged to other states. This result also shows that the true distribution of the uncertainty for the SDLCS is not Gaussian.

6.4.5 Computation Results

We show computational time for the proposed method in Table 6.4. We observe that the compute time for our method increases as the number of integer variables increase. This is expected since we use mixed-integer programming. We also added the computational results for the ERM-based and the CCC method in Table 6.5. Similar to our method, both the ERM-based method and the CCC method incur larger computation time as the number of

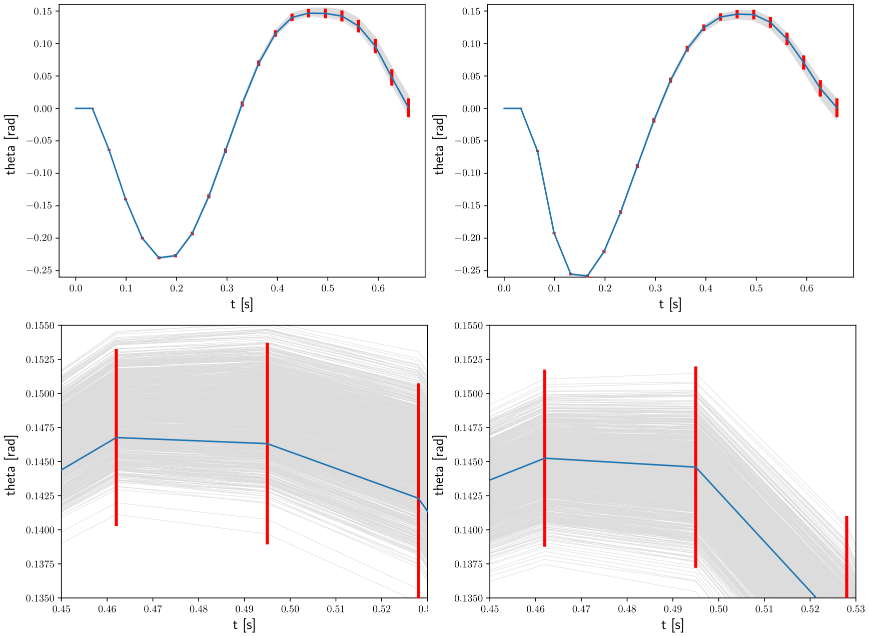


Figure 6.7: Simulated trajectories of x_2 of the cartpole example over 1000 samples with $\Delta = 0.5$ for the left column and with $\Delta = 0.02$ for the right column. The bottom row enlarges the top row figures around the area where the chance constraints effect is observed. The red line shows the 99.9 % confidence interval.

constraints and variables increase.

6.4.6 Discussion of Assumptions

In this section, we will provide empirical justification for the assumptions described in Section 6.2. that we made on the stochasticity of the matrices C, F and the determinism of the complementarity variable λ_{k+1}. Consider the cartpole system. Figure 6.10 plots simulated trajectories of the cartpole system for different realizations of the uncertain parameters. In performing the simulations, we simply sample the uncertainty in the spring constants and forward simulate the DLCS using the computed optimal controls obtained from our proposed optimization in (6.15). The subplots in 6.10a-6.10b plot the pole angle x_2 and the reaction force λ_2 for low value of uncertainty variance of 10^{-6} in the spring constants. The remaining plots 6.10c-6.10d and 6.10e-6.10f plot the same trajectories for larger uncertainty variances of 10^{-4} and $5 \cdot 10^{-4}$ respectively. In this specific scenario, x_1 and λ_1 trajectories are not shown

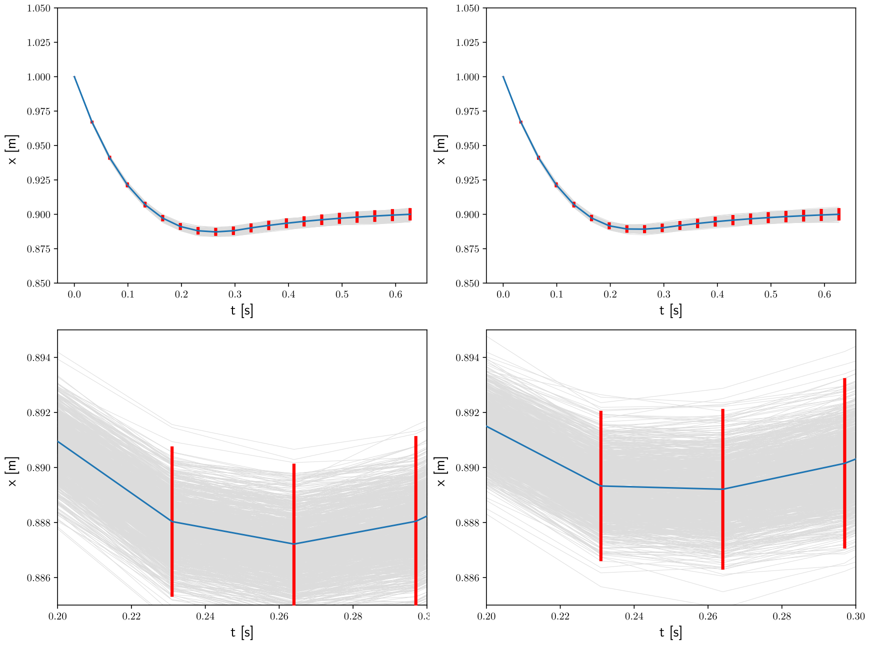


Figure 6.8: Simulated trajectories of x of sliding box with friction example over 1000 samples with $\Delta = 0.5$ for the left column and with $\Delta = 0.002$ for the right column. The bottom row enlarges the top row figures around the area where the chance constraints effect is observed. The red line shows the 99 % confidence interval.

since they do not show significant variation in the simulations. From these plots, we can observe that the simulated state trajectories x_2 (as light grey lines for sampling of uncertainty) are largely concentrated around the optimal trajectory of x_2 (blue line) computed from our proposed optimization approach (6.15). The main objective of the chance constraints is to control the state trajectories within prescribed limits. Our assumptions yield controls that precisely control the states as desired. The contact forces λ_2, in contrast to our assumption, are not necessarily deterministic. However, our approach is able to capture the uncertainty propagation in the states and effectively contain the variance as desired. We have observed a similar behavior in the other systems considered in this chapter. We believe a more rigorous theoretical justification can be provided and we will investigate this in a future work.

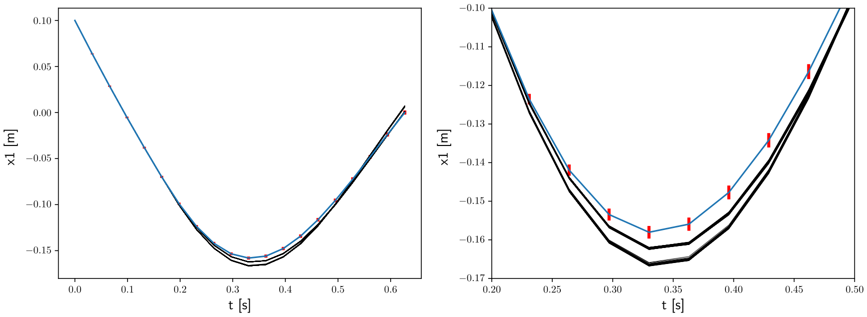
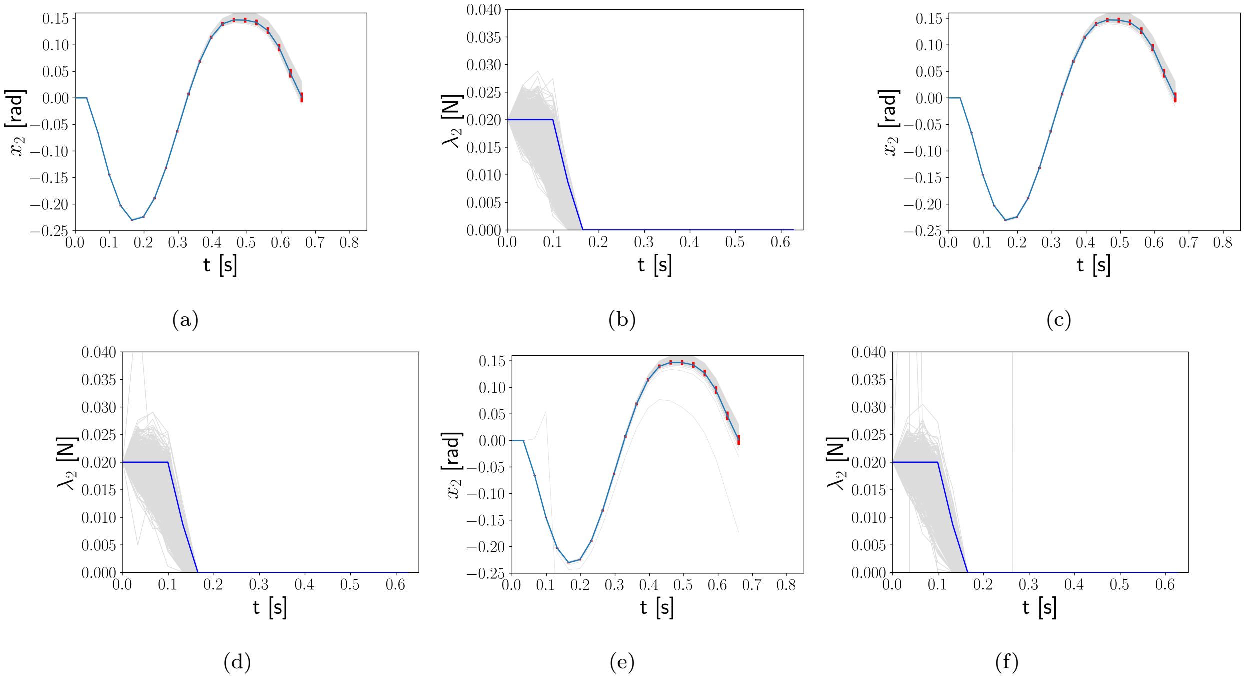


Figure 6.9: Simulated trajectories of x_1 over 1000 samples with $\Delta = 0.2$ for dual manipulation. The right figure shows the enlarged figure of the left figure. The red line shows the 99 % confidence interval.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 6.10: Trajectories for the stochastic cartpole system obtained by using 1000 samples for the uncertain parameters while the control input is set to that computed using our proposed optimization (6.15) where the uncertainty values correspond to those used in Section 6.4.2.1. (a): simulated trajectories of x_2 with uncertain $\frac{1}{k_1}, \frac{1}{k_2}$ for which standard deviations are 10^{-6}. (b): the simulated force trajectories of λ_2 corresponding to (a). (c): simulated trajectories of x_2 with uncertain $\frac{1}{k_1}, \frac{1}{k_2}$ which standard deviations are 10^{-4}. (d): the simulated force trajectories of λ_2 corresponding to (c). (e): simulated trajectories of x_2 with uncertain $\frac{1}{k_1}, \frac{1}{k_2}$ which standard deviations are $5 * 10^{-4}$. (f): the simulated force trajectories of λ_2 corresponding to (e). We rollout dynamics with the control input u which was computed (6.15), the blue lines show the optimal x_2, λ_2 and the rollouts are shown in grey.

Table 6.4: Computation Time of our method. $n_C, n_I, n_{\text{constraints}}$ show the number of continuous variables, the number of integer variables, and the number of total constraints, respectively, for each problem.

	n_C	n_I	$n_{\text{constraints}}$	runtime [s]
Cartpole with softwalls (Ours)	348	80	512	0.023
Sliding box (Ours)	220	120	624	1.95
Dual manipulation (Ours)	720	300	1632	8.29
Planar pushing (Ours)	148	40	270	0.009

Table 6.5: Computation Time of other methods. $n_{\text{variables}}$ and $n_{\text{constraints}}$ show the total number of variables and the number of total constraints, respectively, for each problem.

	$n_{\text{variables}}$	$n_{\text{constraints}}$	runtime [s]
Cartpole with softwalls (ERM)	416	400	0.319
Cartpole with softwalls (CCC)	1178	1160	2.42
Sliding box (ERM)	437	420	0.281
Sliding box (CCC)	1328	1310	2.90
Dual manipulation (ERM)	737	720	0.701
Dual manipulation (CCC)	1468	1450	8.01

6.5 SMPC for Planar Pushing

In this section, we verify our proposed SMPC algorithm for contact-rich system can track a reference trajectory with probabilistic guarantees and outperform a deterministic MPC method. We demonstrate the proposed method for a stochastic pusher-slider system.

6.5.1 Planar Pushing

The frictional interaction between the pusher and slider leads to a linear complementarity system which we describe next. The pusher interacts with the slider by exerting forces in the normal and tangential directions denoted by $f_{\vec{n}}$, $f_{\vec{t}}$ (as shown in Figure 6.11) as well as a torque τ about the center of the mass of the object. Assuming quasi-static interaction, the limit surface [146] defines an invertible relationship between applied wrench $\mathbf{w}$ and the twist of the slider $\mathbf{t}$. The applied wrench $\mathbf{w}$ causes the object to move in a perpendicular direction to the limit surface $\mathbf{H}(\mathbf{w})$. Consequently, the object twist in body frame is given by $\mathbf{t} = \nabla\mathbf{H}(\mathbf{w})$, where the applied wrench $\mathbf{w} = [f_{\overrightarrow{n}}, f_{\overrightarrow{t}}, \tau]$ could be written as $\mathbf{w} = \mathbf{J}^T(\overrightarrow{n} f_{\overrightarrow{n}} + \overrightarrow{t} f_{\overrightarrow{t}})$. For the contact configuration shown in Figure 6.11, the normal and tangential unit vectors are given by $\overrightarrow{n} = [1 \quad 0]^T$ and $\overrightarrow{t} = [0 \quad 1]^T$. The Jacobian $\mathbf{J}$ is given by $\mathbf{J} = \begin{bmatrix} 1 & 0 & -p_y \\ 0 & 1 & p_x \end{bmatrix}$.

The dynamics of the pusher-slider system is given by

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = \begin{bmatrix} \mathbf{R}\mathbf{t} & \dot{p}_y \end{bmatrix}^\top \tag{6.21}$$

where $\mathbf{R}$ is the rotation matrix. The twist $\mathbf{t}$ can be obtained using an approximate limit surface [146] for quasi-static pushing. Since the wrench applied on the system depends of the point of contact of pusher and slider, the state of the system is given by $\mathbf{x} = [x \quad y \quad \theta \quad p_y]^\top$ and the input is given by $\mathbf{u} = [f_{\overrightarrow{n}} \quad f_{\overrightarrow{t}} \quad \dot{p}_y]^\top$. The elements of the input vector must follow the laws of coulomb friction which can be expressed as complementarity conditions as follows:

$$\begin{aligned} 0 \leq \dot{p}_{y+} \perp \mu_p f_{\overrightarrow{n}} - f_{\overrightarrow{t}} \geq 0 \\ 0 \leq \dot{p}_{y-} \perp \mu_p f_{\overrightarrow{n}} + f_{\overrightarrow{t}} \geq 0 \end{aligned} \tag{6.22}$$

where $\dot{p}_y = \dot{p}_{y+} - \dot{p}_{y-}$ and the μ_p is the coefficient of friction between the pusher and the slider.

The world frame and the body frame of reference are denoted by F_w and F_b respectively. The uncertainty in the friction cone is approximately represented by the shaded region in the friction cone. We use a modified version of (6.17) to solve SMPC for planar pusher-slider system. We use $0 \leq u^e_{k,i} + u^*_{k,i} \leq Mz_{k,i,0}$ instead of (6.17e) because we have stochastic complementarity constraints on u, not on λ. We use the following hyper parameters: $\mu = 0.3, m = 1.0$. We use $dt = 0.1$ to discretize the dynamics and set $N = 10, M = 20$. We add uncertainty in μ and dynamics for which the standard deviations are 10^{-4} and 4×10^{-3}, respectively. We consider the following chance constraints: $\Pr(x^* - 0.1 \leq x_{1,k} \leq x^* + 0.1) \geq 1 - \frac{\Delta}{4N}$, $\Pr(x_{2,k} \leq 0.27) \geq 1 - \frac{\Delta}{4N}$. The initial and the terminal state are $x_s = [0, 0, 0]^\top, x_g = [0, 0.2, \pi]^\top$. Also, we formulate our Deterministic MPC (DMPC) which

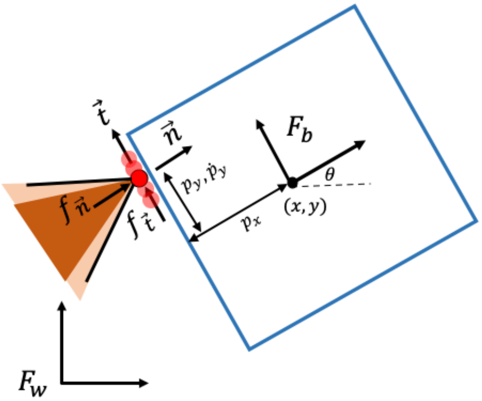


Figure 6.11: A schematic of a planar pusher-slider system. State of the system is $[x, y, \theta, p_y]^\top$ assuming that the pusher only comes in contact with the left edge as shown in the figure.

Table 6.6: Comparison of obtained Δ and MSE with different Δ from the simulation of "pushing with slipping" over 100 samples.

	No MPC	DMPC	$\Delta = 0.5$	$\Delta = 0.01$
Obtained Δ	0.31	0.19	0.10	0.00
MSE	0.00295	0.00244	0.00208	0.00216

uses the constraints $x_k \in \mathcal{X}$ instead of $\Pr(x_k \in \mathcal{X})$. We also evaluate the no-mpc case which implements the reference control sequence in open-loop.

6.5.2 Results

We evaluate the performance of the controllers with respect to:

1. the safety in terms of the chance constraints by counting the number of failures, and
2. the Mean Squared tracking Error (MSE) from the reference trajectory.

Note that a failure is defined as a constraint violation. Similar to the previous section, we use Monte Carlo simulations to evaluate the different controllers.

The obtained Δ and the MSE are shown in Table 6.6. Our proposed SMPC achieves the best performance with respect to both metrics followed by DMPC method. Since our

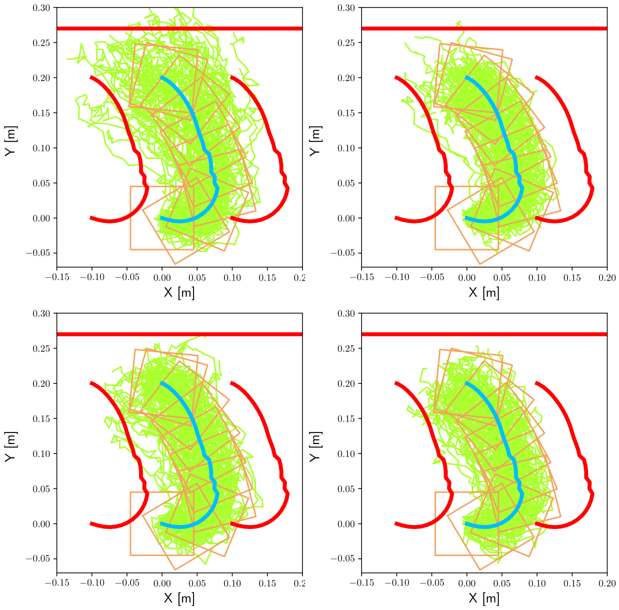


Figure 6.12: Results of SNMPC. Top left: no mpc (open loop), top right: DMPC, bottom left: $\Delta = 0.5$, bottom right: $\Delta = 0.01$. The blue curve shows the reference trajectory of the center of the box and the green lines show the simulated trajectories. The red curves show the bounds.

proposed SMPC considers stochastic complementarity constraints as well as uncertainty propagation, our method achieves lowest tracking error as well as constraint violation probability. We can see the trade-off between $\Delta = 0.5$ and $\Delta = 0.01$ where $\Delta = 0.5$ shows the higher Δ but the lower MSE. The reason why the DMPC shows higher MSE is that the DMPC may diverge from the reference trajectory since it ignores uncertainty propagation.

Fig. 6.12 illustrates the trajectories of MPC with different parameters. We can confirm that our proposed SMPC can track the reference trajectories while satisfying the chance constraints. Also, the average runtime for our SMPC to compute a solution was 0.0029 s during runtime.

6.6 Discussion and Conclusion

The hybrid dynamics of contact-rich interaction as well as uncertainty associated with contact parameters make efficient design of model-based controllers for manipulation challenging. We believe that understanding stochastic and robust optimization and control methods for contact-rich systems is important. However, this topic remains relatively unexplored in literature. One of the key reasons is the difficulty in handling stochastic complementarity constraints and its effect on uncertainty propagation for planning. This poses unique challenges for formulation of computationally feasible algorithms for robust planning of SDLCS.

In this chapter, we presented a robust trajectory optimization technique for contact-rich systems. We presented a formulation for chance constrained optimization for SDLCS which is solved using MIQPCC. This chapter makes an assumption of deterministic complementarity variables for computational tractability. We show that despite this assumption, we are able to compute controllers that are robust to the underlying stochastic system. We compared our proposed approach against other recent techniques for robust optimization for stochastic complementarity systems. We showed that our formulation outperforms these baseline techniques. We show that the proposed chance constrained optimization can be used to design stochastic MPC controllers for contact-rich system. The proposed SMPC was demonstrated for a stochastic planar pushing system.

In this chapter, we have several assumptions. First, we assume that the underlying distribution of random variables is known, which can be quite challenging to get in practice. Thus, we present robust optimization framework in Chapter 7 which does not need to know the distribution of random variables.

Another assumption in this chapter is that we assume that contact forces λ is not a random variable, which is not true. We also assume that the underlying distribution of the random variables follows the Gaussian distribution. Furthermore, we only consider open-loop controllers where the variance keeps increasing over time. To decrease the variance, feedback control is necessary. In Chapter 8, we present our optimization problem that is able to design feedforward and feedback controller simultaneously with chance constraints under non-Gaussian random variables.

CHAPTER 7

Robust Pivoting Manipulation using Contact Implicit Bilevel Optimization

In Chapter 6, we present chance-constrained optimization for contact-rich systems, which can design robust open-loop controller for general contact-rich systems including manipulation and locomotion. In contrast, in this chapter, we present robust optimization considering the underlying structure of the contact dynamics for manipulation with extrinsic contacts. Since we consider the specific manipulation problem, we can design a more robust open-loop controller compared to the work in Chapter 6. We first motivate this work more by describing how friction plays a key role in introducing stability for manipulation. Next, we show some analysis of stability margin considering friction under several different uncertain physical parameters such as coefficients of friction, mass, CoM location, and the contact location. We then propose our robust optimization formulation which is able to design optimal control sequences while improving the worst-case stability margin along the manipulation. The proposed algorithm was evaluated for the pivoting using several different objects in the hardware experiments.

This chapter has been partially adapted from the following papers:

- **Y. Shirai**, D. Jha, and A. Raghunathan, "Robust Pivoting Manipulation using Contact Implicit Bilevel Optimization", (under review for IEEE Transactions on Robotics).
- **Y. Shirai**, D. Jha, A. Raghunathan, and D. Romeres, "Robust Pivoting: Exploiting Frictional Stability Using Bilevel Optimization", in *Proc. 2022 IEEE Int. Conf. Robot. Auto.*, pp. 14-21, 2022.

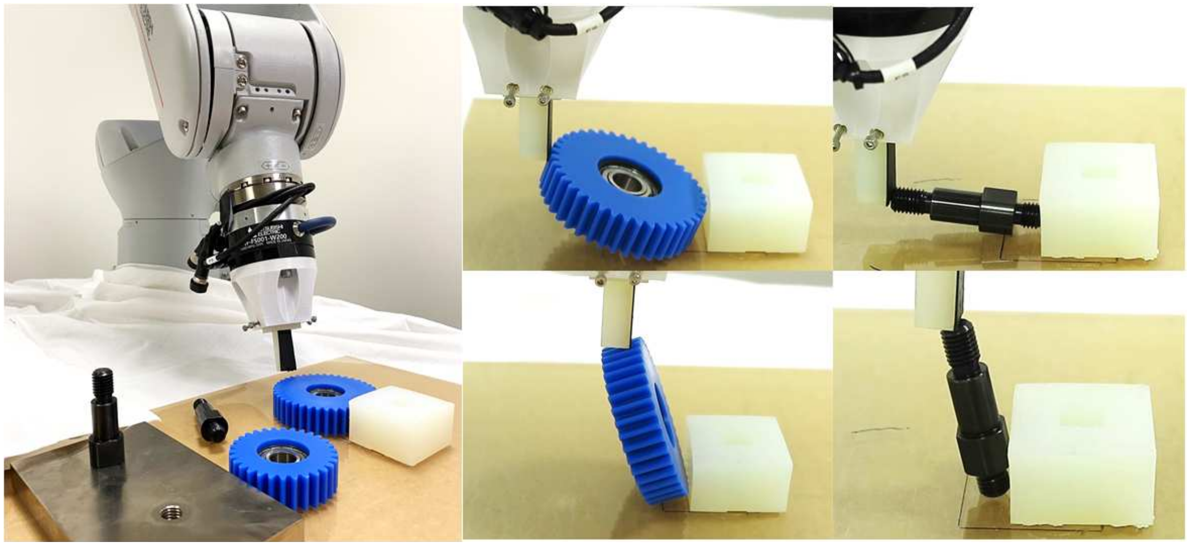


Figure 7.1: We consider the problem of reorienting parts for assembly using pivoting manipulation primitive. Such reorientation could possibly be required when the parts being assembled are too big to grasp in the initial pose (such as the gears) or the parts to be inserted during assembly are not in the desired pose (such as the pegs). The figure shows some instances during the implementation of our controller to reorient a gear and a peg.

7.1 Overview

Contacts are central to most manipulation tasks as they provide additional dexterity to robots to interact with their environment [4]. It is desirable that a robot should be able to interact with unknown objects in unknown environments during operation and thus achieve generalizable manipulation. Robust planning for frictional interaction with objects with uncertain physical properties could be challenging as the mechanical stability of the object depends on these physical properties. Inspired by this problem, we consider the task of robust pivoting manipulation in this chapter. The pivoting task considered in this chapter requires that the slipping contact be maintained at the two external contact points which presents unique challenges for robust planning. We are interested in ensuring mechanical stability via friction to compensate for uncertainty in the physical properties (e.g., mass, Center of Mass (CoM) location, coefficient of friction, contact location.) of the objects during manipulation. We present a formulation and an optimization technique that can solve robust manipulation trajectories for the proposed pivoting manipulation.

Robust planning (and control) for frictional interaction is challenging due to the hybrid nature of underlying frictional dynamics. Consequently, a lot of classical robust planning and control techniques are not applicable to these systems in the presence of uncertainties [2, 16, 147]. While concepts of stability margin or Lyapunov stability have been well studied in the context of nonlinear dynamical system controller design [148], such notions have not been explored in contact-rich manipulation problems. This can be mostly attributed to the fact that a controller has to reason about the mechanical stability constraints of the frictional interaction to ensure stability. Mechanical stability closely depends on the contact configuration during manipulation, and thus a planner (or controller) has to ensure that the desired contact configuration is either maintained during the task or it can maintain stability even if the contact sequence is perturbed. Analysis of such systems is difficult in the presence of friction as it leads to differential inclusion system (see [138]). One of the key insights we present in this chapter is that friction provides mechanical stability margin during a contact-rich task. We call the mechanical stability provided by friction as *Frictional Stability*. This *frictional stability* can be exploited during optimization to allow stability of manipulation in the presence of uncertainty. We show the effect of several different parameters on the stability of the manipulation using the proposed approach. In particular, we consider the effect of contact modes and point of contact between the robot & object on the stability of the manipulation. We believe that our proposed ideas could also be used for designing feedback controllers to correct contact trajectories based on estimates of contact states.

We study pivoting manipulation where the object being manipulated has to maintain slipping contact with two external surfaces (see Fig. 7.2). A robot can use this manipulation to reorient parts on a planar surface to allow grasping or assist in assembly by manipulating objects to a desired pose (see Fig. 7.1). Note that this manipulation is challenging as it requires controlled slipping (as opposed to sticking contact [60, 53, 134]), and thus it is imperative to consider robustness of the control trajectories. Ensuring robustness for slipping contact is challenging due to the equality constraints for the friction forces compared to inequality constraints for sticking contact. To ensure mechanical stability of the two-point pivoting in the presence of uncertainty, we derive a sufficient condition for stability which allows us to compute a margin of stability. This margin is then used in a bilevel optimization routine, CIBO (Contact Implicit Bilevel Optimization). Our proposed CIBO designs an optimal control trajectory while maximizing the worst-case margin along the entire trajectory for manipulation. Through numerical simulations as well as physical experiments, we verify that CIBO is able to achieve more robustness compared to the basic trajectory optimization.

Contributions. This chapter has the following contributions.

1. We present analysis of mechanical stability of pivoting manipulation with uncertainty in mass, CoM location, contact location, and coefficient of friction.

2. We present a robust contact-implicit bilevel optimization (CIBO) technique which can be used to optimize the mechanical stability margin to compute robust trajectories for pivoting manipulation. For objects with non-convex shapes, we present a formulation with mode-based optimization.

The proposed method is demonstrated for reorienting parts using a 6 DoF manipulator (see Fig. 7.1.

7.2 Mechanics of Pivoting

In this section, we explain quasi-static stability of two-point pivoting in a plane. Before explaining the details, we present our assumptions in this work. The following assumptions are used in the model for the pivoting manipulation task presented in this chapter:

1. The object is rigid.

2. We consider quasi-static equilibrium of the object.

3. The external contact surfaces are perfectly flat.

4. The dimensions and pose of the object is perfectly known.

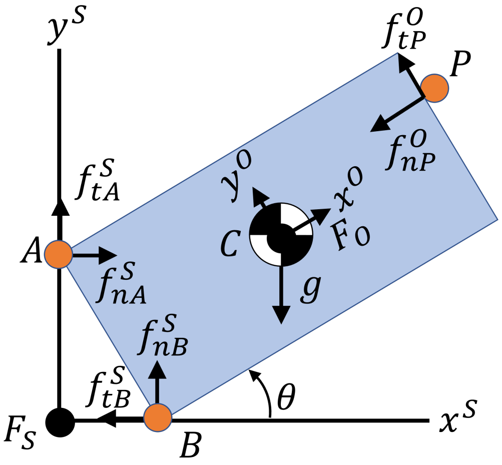


Figure 7.2: A schematic showing the free-body diagram of a rigid body during pivoting manipulation when the relative angle between F_W and F_S is zero. Point P is the contact point with a manipulator. The black circle represents the origin of each frame. The object experiences four forces corresponding to two friction forces from external contact points A and B, one control input f_P from the manipulator at point P, and gravity at point C.

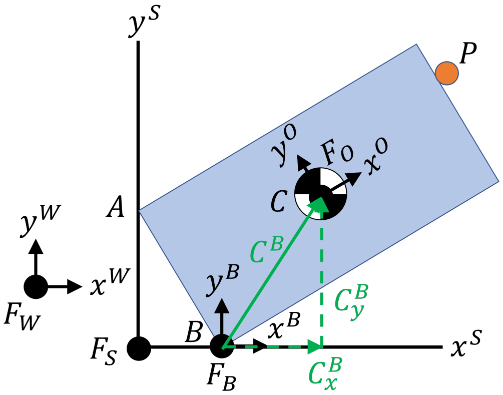


Figure 7.3: A schematic showing the frame definition of a rigid body during pivoting manipulation. F_W, F_S, F_O, and F_B are the world frame, slope frame, object frame, and frame at contact location B, respectively. Gravity is defined in F_W where the gravity is parallel to y-axis of F_W. Pivoting manipulation happens with extrinsic contact A and B defined in F_S. F_O is fixed with CoM of an object. F_B is in parallel to F_S with offset B^S_x along x-axis of F_S. We also show an example of i^Σ_x and i^Σ_x in Table 7.1. In this example, C^B_x and C^B_y are illustrated.

5. The object makes point contacts.

7.2.1 Mechanics of Pivoting with External Contacts

We consider pivoting where the object maintains slipping contact with two external surfaces (see Fig. 7.2). A free body diagram showing the quasi-static equilibrium of the object is shown in Fig. 7.2. The definitions of frames and variables are summarized in Fig. 7.3 and Table 7.1, respectively. In the later sections, we present trajectory optimization formulation where we consider decision variables at time step k (e.g., $f_{k,ni}$). In this section, we remove k to represent variables for simplicity.

The quasi-static equilibrium conditions for the object in F_B when the relative angle between F_W and F_S is zero (see Fig. 7.2) can be represented by the following equations.

$$f_{nA}^B + f_{tB}^B + f_{xP}^B = 0, \tag{7.1a}$$

$$f_{tA}^B + f_{nB}^B + mg + f_{yP}^B = 0, \tag{7.1b}$$

$$A_x^B f_{tA}^B - A_y^B f_{nA}^B + C_x^B mg + P_x^B f_{yP}^B - P_y^B f_{xP}^B = 0 \tag{7.1c}$$

Note that because we define F_B as parallel to F_S, all force variables in F_B and F_S are the same. We consider Coulomb friction law which results in friction cone constraints as follows:

$$|f_{tA}^B| \leq \mu_A f_{nA}^B, |f_{tB}^B| \leq \mu_B f_{nB}^B, \quad f_{nA}^B, f_{nB}^B \geq 0, \tag{7.2}$$

To describe sticking-slipping complementarity constraints, we have the following complementarity constraints at point A, B:

$$0 \leq \dot{A}_{y+}^B \perp \mu_A f_{nA}^B - f_{tA}^B \geq 0, \tag{7.3a}$$

$$0 \leq \dot{A}_{y-}^B \perp \mu_A f_{nA}^B + f_{tA}^B \geq 0, \tag{7.3b}$$

$$0 \leq \dot{B}_{x+}^B \perp \mu_B f_{nB}^B - f_{tB}^B \geq 0, \tag{7.3c}$$

$$0 \leq \dot{B}_{x-}^B \perp \mu_B f_{nB}^B + f_{tB}^B \geq 0 \tag{7.3d}$$

where the slipping velocities follows $\dot{A}_y^B = \dot{A}_{y+}^B - \dot{A}_{y-}^B, \dot{B}_x^B = \dot{B}_{x+}^B - \dot{B}_{x-}^B$. $\dot{A}_{y+}^B, \dot{A}_{y-}^B$ represent the slipping velocity at A along positive and negative directions for y-axis in F_B, respectively.

Table 7.1: Notation of variables for analysis of frictional stability margin. In Σ column, we indicate the frame of variables. We use the following indices for defining variables in this table: $j \in \{A, B, C, P\}$ for representing the location of frames, $i \in \{A, B, P\}$ for representing contact location, and $\Sigma \in \{W, S, O, B\}$ for representing a frame.

Name	Description	Size	Σ
F_Σ	Σ frame.		
f_{nj}^{Σ}	normal force at j in frame F_Σ	$\mathbb{R}^1$	Σ
f_{tj}^{Σ}	friction force at j in frame F_Σ	$\mathbb{R}^1$	Σ
f_{xj}^{Σ}	force at j along x-axis in frame F_Σ	$\mathbb{R}^1$	Σ
f_{yj}^{Σ}	force at j along y-axis in frame F_Σ	$\mathbb{R}^1$	Σ
m	mass	$\mathbb{R}^1$	
g	gravity acceleration	$\mathbb{R}^1$	W
l	length of an object	$\mathbb{R}^1$	
w	width of an object	$\mathbb{R}^1$	
μ_i	coefficient of friction at i	$\mathbb{R}^1$	
i_x^{Σ}	contact location at i along x-axis in frame F_Σ	$\mathbb{R}^1$	Σ
i_y^{Σ}	contact location at i along y-axis in frame F_Σ	$\mathbb{R}^1$	Σ
$\dot{i}_x^{\Sigma}$	slipping velocity at i along x-axis in frame F_Σ	$\mathbb{R}^1$	Σ
$\dot{i}_y^{\Sigma}$	slipping velocity at i along y-axis in frame F_Σ	$\mathbb{R}^1$	Σ
θ	angle of an object	$\mathbb{R}^1$	S
ϕ	relative angle of frame from $\{F_W\}$ to $\{F_S\}$	$\mathbb{R}^1$	W

$\dot{B}^B_{x+}, \dot{B}^B_{x-}$ represent the slipping velocity at B along positive and negative directions for x-axis in F_B, respectively. The notation $0 \leq a \perp b \geq 0$ means the complementarity constraints $a \geq 0, b \geq 0, ab = 0$. Since we consider slipping contact during pivoting, we have "equality" constraints in friction cone constraints at points A, B:

$$f^B_{tA} = \mu_A f^B_{nA}, f^B_{tB} = -\mu_B f^B_{nB} \tag{7.4}$$

To realize stable pivoting, actively controlling position of point P is important. Thus, we consider the following complementarity constraints that represent the relation between the slipping velocity $\dot{P}_y$ at point P in F_O and friction cone constraint at point P:

$$0 \leq \dot{P}^O_{y+} \perp \mu_p f^O_{nP} - f^O_{tP} \geq 0 \tag{7.5a}$$

$$0 \leq \dot{P}^O_{y-} \perp \mu_p f^O_{nP} + f^O_{tP} \geq 0 \tag{7.5b}$$

where $\dot{P}^O_y = \dot{P}^O_{y+} - \dot{P}^O_{y-}$.

7.3 Robust Pivoting Formulation

In this section, we present a generic formulation for robust pivoting manipulation. In particular, we use the quasi-static equilibrium conditions (7.1) in the presence of disturbances to formulate the robust planning problem. In particular, using sufficiency for stability of the object during manipulation we can estimate the bound of disturbance that can be tolerated during manipulation. Since this bound would depend on the pose of the object, we reason about the margin throughout the manipulation trajectory during the optimization problem formulation. We present the general idea in the following paragraph.

In the most general case, we assume that there is an external force F^B_{ext} and moment M^B_{ext} acting on the object during manipulation. Let us assume that the x and y component of the external force in F_B are represented as $F^B_{ext,x}$ and $F^B_{ext,y}$ respectively. Then the quasi-static equilibrium conditions (7.1) can be rewritten as follows:

$$f_{nA}^B + f_{tB}^B + f_{xP}^B + F_{ext,x}^B = 0, \tag{7.6a}$$

$$f_{tA}^B + f_{nB}^B + mg + f_{yP}^B + F_{ext,y}^B = 0, \tag{7.6b}$$

$$A_x^B f_{tA}^B - A_y^B f_{nA}^B + C_x^B mg + P_x^B f_{yP}^B - P_y^B f_{xP}^B + M_{ext}^B = 0 \tag{7.6c}$$

Note that F_{ext}^B and M_{ext}^B may not be independent of each other. They are related via the the point of application of force F_{ext}^B in the quasi-static equilibrium conditions (7.6). These equations may not be satisfied for all possible values of F_{ext}^B and M_{ext}^B. Since the contact forces can be readjusted in (7.6), the quasi-static equilibrium can be satisfied for a certain range of F_{ext}^B and M_{ext}^B. A generic analysis for estimating this margin or bound for which these disturbances can be compensated by contact forces is a bit involved as such a bound is dependent on the point and angle of application of the external force F_{ext}^B. In the following sections, we present some specific cases which can be analyzed by making some simplifying assumptions on these disturbances. For brevity, we omit superscript B of variables in the following sections because we consider quasi-static equilibrium in F_B unless we consider quasi-static equilibrium in a different frame (see Sec 7.3.5).

7.3.1 Frictional Stability Margin

The robust quasi-static equilibrium conditions shown in (7.6) can be used to explain the concept of stability margin. The stability margin is given by the magnitude of the external force F_{ext}^B and moment M_{ext}^B which can be satisfied in (7.6) in any stable configuration of the object. This margin would depend on the contact force between the object and the environment as well as the control force used by the manipulator during the task. This provides the intuition that one can design a control trajectory such that the stability margin can be maximized.

We briefly provide some physical intuition about frictional stability for a few specific cases. First suppose that uncertainty exists in mass of a body. In the case when the actual mass is lower than estimated, the friction force at point A would increase while the friction force at point B would decrease, compared to the nominal case. In contrast, suppose if the actual mass of the body is heavier than that of what we estimate, then the body can tumble along point B in the clockwise direction. In this case, we can imagine that the friction force at point A would decrease while the friction force at point B would increase. However, as long as the friction forces are non-zero, the object can stay in contact with the external environment. Similar arguments could be made for uncertainty in CoM location. The key point to note that the friction forces can re-distribute at the two contact locations and thus provide a margin of stability to compensate for uncertain gravitational forces and moments. We call this margin as *frictional stability*.

In the following sections, we present the mathematical formulation of *frictional stability* for cases when the mass, CoM location, friction coefficients, or finger contact location are not known perfectly.

7.3.2 Stability Margin for Uncertain Mass

For simplicity, we denote ϵ as uncertain weight with respect to the estimated weight. Also, to emphasize that we consider the system under uncertainty, we put superscript ϵ for each friction force variable. Thus, the quasi-static equilibrium conditions in (7.1) can be rewritten as:

$$f^{\epsilon}_{nA} + f^{\epsilon}_{tB} + f_{xP} = 0, \tag{7.7a}$$

$$f^{\epsilon}_{tA} + f^{\epsilon}_{nB} + (mg + \epsilon) + f_{yP} = 0, \tag{7.7b}$$

$$A_x f^{\epsilon}_{tA} - A_y f^{\epsilon}_{nA} + C_x(mg + \epsilon) + P_x f_{yP} = P_y f_{xP} \tag{7.7c}$$

Then, using (7.4) and (7.7c), we obtain:

$$f^{\epsilon}_{nA} = \frac{-C_x\,(mg + \epsilon) - P_x f_{yP} + P_y f_{xP}}{\mu_A A_x - A_y} \tag{7.8}$$

To ensure that the body maintains contact with the external surfaces, we would like to enforce that the body experience non-zero normal forces at the both contacts. To realize this, we have $f^{\epsilon}_{nA} \geq 0, f^{\epsilon}_{nB} \geq 0$ as conditions that the system needs to satisfy. Consequently, by simplifying (7.8), we get the following:

$$\epsilon \geq \frac{P_y f_{xP} - P_x f_{yP} - C_x mg}{C_x}, \text{ if } C_x > 0, \tag{7.9a}$$

$$\epsilon \leq \frac{P_y f_{xP} - P_x f_{yP} - C_x mg}{C_x}, \text{ if } C_x < 0 \tag{7.9b}$$

Note that the upper-bound of ϵ means that the friction forces can exist even when we make the mass of the body lighter up to $\frac{\epsilon}{g}$. The lower-bound of ϵ means that the friction forces can exist even when we make the mass of the body heavier up to $\frac{\epsilon}{g}$. (7.9) provides some useful insights. (7.9) gives either upper- or lower-bound of ϵ for f^{ϵ}_{nA} according to the sign of C_x (the moment arm of gravity). This is because the uncertain mass would generate an additional moment along with point B in the clock-wise direction if $C_x > 0$ and in the counter clock-wise direction if $C_x < 0$. If $C_x = 0$, we have an unbounded range for ϵ, meaning that the body would not lose contact at point A no matter how much uncertainty exists in the mass.

(7.9) can be reformulated as an inequality constraint:

$$C_x(\epsilon - \epsilon_A) \geq 0 \tag{7.10}$$

where $\epsilon_A = \frac{P_y f_{xP} - P_x f_{yP} - C_x mg}{C_x}$.

We can derive condition for ϵ based on $f^{\epsilon}_{nB} \geq 0$ from (7.4), (7.7a), and (7.7b):

$$\epsilon \leq \mu_A f_{xP} - f_{yP} - mg \tag{7.11}$$

We only have upper-bound on ϵ based on $f^{\epsilon}_{nB} \geq 0$, meaning that the contact at point B cannot be guaranteed if the actual mass is lighter than $\mu_A f_{xP} - f_{yP} - mg$.

7.3.3 Stability Margin for Uncertain CoM Location

We denote d^O_x, d^O_y as residual CoM locations with respect to the estimated CoM location in F_O coordinate, respectively. Thus, the residual CoM location in F_W, d^W_x, d^W_y, are represented by $d^W_x = d\cos(\theta + \theta_d), d^W_y = d\sin(\theta + \theta_d)$, where $d = \sqrt{(d^O_x)^2 + (d^O_y)^2}$, $\theta_d = \arctan\frac{d^O_y}{d^O_x}$. For notation simplicity, we use r to represent d^W_x. In this chapter, we put superscript r for each friction force variable. The quasi-static equilibrium conditions in (7.1) can be rewritten as follows:

$$f_{nA}^r + f_{tB}^r + f_{xP} = 0, \tag{7.12a}$$

$$f_{tA}^r + f_{nB}^r + mg + f_{yP} = 0, \tag{7.12b}$$

$$A_x f_{tA}^r - A_y f_{nA}^r + (C_x + r)mg + P_x f_{yP} = P_y f_{xP} \tag{7.12c}$$

Then, using (7.4) in (7.12), we obtain:

$$r \leq \frac{P_y f_{xP} - P_x f_{yP}}{mg} - C_x, \tag{7.13a}$$

$$r \geq -\frac{\frac{\mu_A A_x - A_y}{1+\mu_A}(-f_{xP} - f_{yP} - mg) - P_y f_{xP} + P_x f_{yP}}{mg} - C_x \tag{7.13b}$$

where (7.13a), (7.13b) are obtained based on $f_{nA}^r \geq 0, f_{nB}^r \geq 0$, respectively. (7.13) means that the object would lose contact at A if the actual CoM location is more to the right than our expected CoM location while the object would lose the contact at B if the actual CoM location is more to the left.

7.3.4 Stability Margin for Stochastic Friction

In this section, we present modeling and analysis of pivoting manipulation in the presence of stochastic friction coefficients. In particular, we consider stochastic friction at the two different contact points A and B. We do not consider stochastic friction at the contact point between the robot and the manipulator since that leads to stochastic complementarity constraints (please see [44, 147] for detailed analysis on stochastic complementarity constraints). We make the assumption that the friction coefficients at A and B are partially known. In particular, we assume that the friction coefficients for contact at A could be represented as $\mu_A = \hat{\mu}_A + \tilde{\mu}_A$ where $\tilde{\mu}_A$ is the uncertain stochastic variable. Similarly, the friction coefficient at B could be represented as $\mu_B = \hat{\mu}_B + \tilde{\mu}_B$ where $\tilde{\mu}_B$ is the uncertain stochastic variable. Note that we do not need to need to know any information regarding the probabilistic distribution (e.g., probability density function of Gaussian distribution, beta distribution.) of the unknown part. We can rewrite (7.6) for this case as follows. We put superscript μ for each friction variable:

$$f^{\mu}_{nA} + \hat{f}^{\mu}_{tB} + f_{xP} + \epsilon_B = 0, \tag{7.14a}$$

$$\hat{f}^{\mu}_{tA} + f^{\mu}_{nB} + mg + f_{yP} + \epsilon_A = 0, \tag{7.14b}$$

$$A_x\hat{f}^{\mu}_{tA} + A_x\epsilon_A - A_y f^{\mu}_{nA} + C_x mg + P_x f_{yP} - P_y f_{xP} = 0 \tag{7.14c}$$

where, $f^{\mu}_{tA} = \hat{f}^{\mu}_{tA} + f^{\mu}_{nA}\tilde{\mu}_A$ and $f^{\mu}_{tB} = \hat{f}^{\mu}_{tB} + f^{\mu}_{nB}\tilde{\mu}_B$. The above equations are obtained by representing $f_{nA}\tilde{\mu}_A$ as ϵ_A for contact at A and similarly, ϵ_B for the contact at B. Thus, ϵ_A and ϵ_B are the uncertain contact forces for the contacts at A and B. The robust formulation that we consider in this chapter considers the worst-case effect of these uncertainties on the stability of the object during manipulation. Thus, we try to maximize the bound of these variables ϵ_A and ϵ_B using our proposed bilevel optimization. It is noted that ϵ_A and ϵ_B are the stability margin for this particular case of stochastic friction.

To ensure that the body maintains contact, we impose $f^{\mu}_{nA} \geq 0, f^{\mu}_{nB} \geq 0$, so that we get the following inequalities for ϵ_A, ϵ_B:

$$-\mu_A f_{xP} + \epsilon_A + mg + f_{yP} \leq \mu_A\epsilon_B \tag{7.15a}$$

$$\epsilon_B \leq -\mu_B(\epsilon_A + mg + f_{yP}) - f_{xP} \tag{7.15b}$$

To ensure slipping contact even in the presence of uncertainties, we need to satisfy friction cone constraints specified earlier in (7.2), (7.4). Using these constraints, we can find the upper and lower bound for the variables ϵ_A and ϵ_B:

$$(\hat{\mu}_A + \tilde{\mu}_A)f^{\mu}_{nA} = \hat{f}^{\mu}_{tA} + \tilde{\mu}_A f^{\mu}_{nA} \tag{7.16a}$$

$$(\hat{\mu}_B + \tilde{\mu}_B)f^{\mu}_{nB} = -\hat{f}^{\mu}_{tB} - \tilde{\mu}_B f^{\mu}_{nB} \tag{7.16b}$$

To get a lower bound for the variables ϵ_A and ϵ_B, we make a assumption regarding the uncertainty for the friction coefficients at A and B. We assume that the unknown part is bounded above by the known part, i.e., $\tilde{\mu}_i \leq \hat{\mu}_i, \forall i = A, B$. Note that this is not a restrictive assumption. What this implies is that the above parameter has bounded uncertainty. For simplicity, we assume that uncertainty is bounded by the known part of the parameter. For example, if the friction coefficient is modeled as a stochastic random variable, then we assume that we know the mean of the friction parameter and the standard deviation is bounded by some multiple of mean (note that this bound is just for simplification and one can assume any practical bound for uncertainty). Consequently, we can derive the following relations:

$$-\hat{\mu}_A f_{nA}^{\mu} \leq \epsilon_A \leq \hat{\mu}_A f_{nA}^{\mu} \tag{7.17a}$$

$$-\hat{\mu}_B f_{nB}^{\mu} \leq \epsilon_B \leq \hat{\mu}_B f_{nB}^{\mu} \tag{7.17b}$$

Thus, we get constraints (7.15) and (7.17) for the stability margin by considering the stability and the friction cone constraints in the presence of uncertain friction coefficients. These constraints are used to estimate the stability margin during the proposed bilevel optimization.

7.3.5 Stability Margin for Finger Contact Location

We consider the stability margin d of finger contact location on an object due to imperfect stiffness controller from robotic manipulators, as illustrated in Fig. 7.4. We can formulate the following quasi-static equilibrium in F_O. We put superscript d for each extrinsic friction variable:

$$f_{xA}^{O,d} + f_{xB}^{O,d} + mg\sin\theta + f_{nP}^{O} = 0, \tag{7.18a}$$

$$f_{yA}^{O,d} + f_{yB}^{O,d} + mg\cos\theta + f_{tP}^{O} = 0 \tag{7.18b}$$

$$\sum_{i\in\{A,B\}} \left(i_x^O f_{yi}^{O,d} - i_y^O f_{xi}^{O,d}\right) + P_x^O f_{tP}^O - (P_y^O + d) f_{nP}^O = 0 \tag{7.18c}$$

Note that $-A_x^O = -B_x^O = P_x^O = \frac{l}{2}, A_y^O = -B_y^O = \frac{w}{2}$. Using this relation, we can simplify (7.18). In particular, we use $f_{xA}^{O,d} \geq 0, f_{xB}^{O,d} \geq 0, f_{nP}^O \geq 0$ and thus we can get the following bound for d:

$$\underline{d} \leq d \leq \bar{d}, \tag{7.19a}$$

$$\underline{d} = -A_x \frac{mg\cos\theta + 2f_{tP}}{f_{nP}} - A_y \frac{mg\sin\theta + f_{nP}}{f_{nP}} - P_y^O, \tag{7.19b}$$

$$\bar{d} = -A_x \frac{mg\cos\theta + 2f_{tP}}{f_{nP}} + A_y \frac{mg\sin\theta + f_{nP}}{f_{nP}} - P_y^O \tag{7.19c}$$

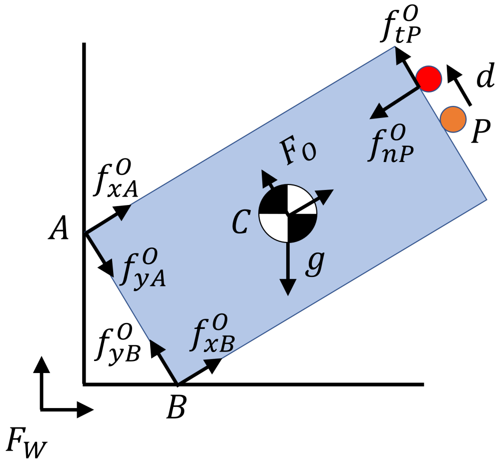


Figure 7.4: A schematic showing the free-body diagram of a rigid body during pivoting manipulation. We consider the stability margin of finger location due to imperfect control of stiffness controller in a robotic manipulator.

When $f_{nP}^{O} \to 0$, the equation suggests that $\bar{d}$ tends to infinity and $\underline{d}$ tends to negative infinity. As $f_{nP}^{O} = 0$ implies no force at point P, the finger's placement becomes inconsequential as it does not affect the quasi-static equilibrium of the object.

We can consider that uncertainty in finger contact location and uncertainty in the geometry of an object have a similar influence on the manipulation. This is because the relative pose of the object with respect to the robot changes for both cases, resulting in the potential contact mode changes.

7.3.6 Stability Margin for Uncertain Mass on a Slope

We consider the case where we tilt the two external walls by the angle of ϕ. Our discussion in Sec. 7.3.2 still holds. The only difference arises from gravity terms. Hence, the quasi-static equilibrium conditions in F_B can be rewritten as:

$$f_{nA}^{\epsilon} + f_{tB}^{\epsilon} + f_{xP} + (mg + \epsilon)\sin\phi = 0, \tag{7.20a}$$

$$f_{tA}^{\epsilon} + f_{nB}^{\epsilon} + f_{yP} + (mg + \epsilon)\cos\phi = 0, \tag{7.20b}$$

$$A_x f_{tA}^{\epsilon} - A_y f_{nA}^{\epsilon} + (C_x \cos\phi - C_y \sin\phi)(mg + \epsilon) + P_x f_{yP} - P_y f_{xP} = 0 \tag{7.20c}$$

Following the same logic in Sec. 7.3.2, we can get the following bound for the stability margin ϵ under uncertain mass when the object is on a slope:

$$\epsilon \geq \frac{P_y f_{xP} - P_x f_{yP} - (C_x \cos\phi - C_y \sin\phi) mg}{C_x \cos\phi - C_y \sin\phi}, \text{ if } C_x \cos\phi > C_y \sin\phi \tag{7.21a}$$

$$\epsilon \leq \frac{P_y f_{xP} - P_x f_{yP} - (C_x \cos\phi - C_y \sin\phi) mg}{C_x \cos\phi - C_y \sin\phi}, \text{ if } C_x \cos\phi < C_y \sin\phi \tag{7.21b}$$

As a result, (7.21a) and (7.21b) result in the following inequality constraint:

$$(C_x \cos\phi - C_y \sin\phi)(\epsilon - \epsilon_A) \geq 0 \tag{7.22}$$

where $\epsilon_A = \frac{P_y f_{xP} - P_x f_{yP} - (C_x \cos\phi - C_y \sin\phi) mg}{C_x \cos\phi - C_y \sin\phi}$. We also derive the bound on ϵ using $f_{nB}^{\epsilon} \geq 0$, (7.21a), and (7.21b):

$$(\mu_A \sin\phi - \cos\phi)\,\epsilon \geq f_{yP} - \mu_A f_{xP} \tag{7.23}$$

Note that the sign of $\mu_A \sin\phi - \cos\phi$ can change depending on the angle of slope. In this chapter, we choose ϕ such that the sign of $\mu_A \sin\phi - \cos\phi$ does not change during manipulation.

The discussion in this section for manipulation under uncertain mass on a slope can be easily extended with other uncertain parameters such as CoM location, friction, and finger contact location.

7.3.7 Pivoting with Patch Contact between the object and the manipulator

In the previous sections, we considered point contact between the manipulator and the object. This could be potentially restrictive. Moreover, this may not be a realistic assumption when a robot is interacting with objects. In this section, we present a slightly modified formulation by considering patch contact between the object and the manipulator. We would like to analyze and understand how patch contact would compare against a point contact model for stability during pivoting manipulation. Fig. 7.5 shows the simplest patch contact model during the pivoting task we consider in this chapter. Using this model, we can write the

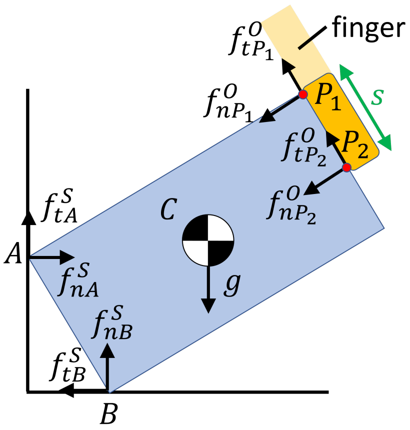


Figure 7.5: A schematic showing the free-body diagram of a rigid body during pivoting manipulation with patch contact. We approximate patch contact as two point contacts P_1 and P_2 with the same force distribution. We assume that P_1 always lies on the vertex of the object for this simplistic patch contact model. s is the distance between point contact P_1 and P_2 along y-axis of F_O.

following quasi-static equilibrium:

$$f_{nA} + f_{tB} + f_{xP_1} + f_{xP_2} = 0, \tag{7.24a}$$

$$f_{tA} + f_{nB} + mg + f_{yP_1} + f_{yP_2} = 0, \tag{7.24b}$$

$$A_x f_{tA} - A_y f_{nA} + C_x mg + \sum_{i=1}^{2} \left(P_{i_x} f_{yP_i} - P_{i_y} f_{xP_i} \right) = 0 \tag{7.24c}$$

where P_{i_x}, P_{i_y} represent x and y coordinate of P_1 and P_2 in F_O, respectively. In this work, we assume that patch contact as two point contacts P_1 and P_2 as the same force distribution, which indicates that $f_{xP_1} = f_{xP_2}, f_{yP_1} = f_{yP_2}$. s is the distance between point contact P_1 and P_2 and s is a decision variable, meaning that location of P_2 is a decision variable and can change over time. In this work, we assume that P_1 does not move over time, which simplifies the model of patch contact.

Using the above quasi-static equilibrium conditions with $f_{nA} \geq 0, f_{nB} \geq 0$, we can solve and find the upper and the lower bound of stability margin under the various uncertainties described earlier in the previous subsections. We will present some results in the later section using this formulation and compare against the point contact formulation.

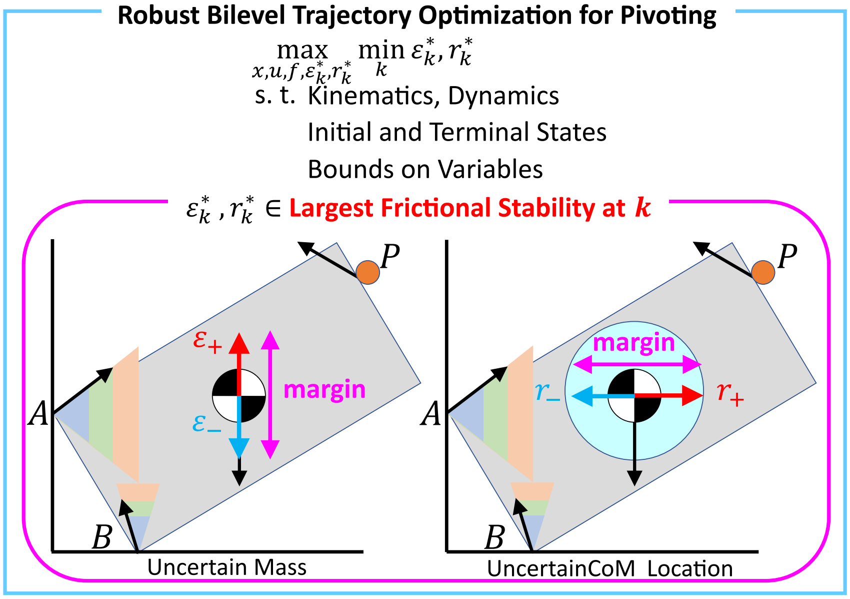

Robust Bilevel Trajectory Optimization for Pivoting
$\max_{x,u,f,\varepsilon_k^*,r_k^*} \min_k \varepsilon_k^*, r_k^*$
s. t. Kinematics, Dynamics
Initial and Terminal States
Bounds on Variables
$\varepsilon_k^*, r_k^* \in$ Largest Frictional Stability at k
P
A
B
ε_+
ε_-
margin
Uncertain Mass
P
A
B
margin
r_-
r_+
UncertainCoM Location

Figure 7.6: Conceptual schematic of our proposed frictional stability and robust trajectory optimization for pivoting. Due to slipping contact, friction forces at points A, B lie on the edge of friction cone. Given the nominal trajectory of state and control inputs, friction forces can account for uncertain physical parameters to satisfy quasi-static equilibrium. We define the range of disturbances that can be compensated by contacts as frictional stability. The above figure shows the case of uncertain mass and CoM location.

7.4 Robust Trajectory Optimization

Using the notion of *frictional stability* introduced in the previous section, we describe our proposed contact implicit bilevel optimization (CIBO) method for robust optimization of manipulation trajectories. The proposed method explicitly considers frictional stability under uncertain physical parameters. It is noted that the proposed method considers robustness under slipping contact which results in equality for friction cone constraints (see Fig. 7.6). After describing the formulation for convex objects, we also describe how to extend the proposed CIBO to consider objects with non-convex geometry. Our proposed method is also presented as a schematic in Fig. 7.6. As shown in Fig. 7.6, the proposed CIBO considers frictional stability margin along the entire trajectory for manipulation and then maximizes the minimum margin in the proposed framework. This is also explained in Fig. 7.7, where we show that we estimate the bound of stability margin in the lower level optimization and maximize the minimum margin in the upper level optimization. Before introducing our proposed bilevel optimization, we present a baseline contact-implicit TO which can be formulated as an MPCC.

7.4.1 Contact-Implicit Trajectory Optimization for Pivoting

The purpose of our optimal control is to find optimal control input sequences under constraints for pivoting manipulation. In particular, we consider the objective function for achieving the minimum motion of objects under kinematics constrains, quasi-static equilibrium, friction cone constraints, and sticking-slipping complementarity constraints as follows:

$$\min_{x,u,f} \sum_{k=1}^{N} (x_k - x_g)^\top Q (x_k - x_g) + \sum_{k=0}^{N-1} u_k^\top R u_k \tag{7.25a}$$

$$\text{s. t. } i_{k,x}, i_{k,y} \in FK(\theta_k, P^O_{k,y}), (7.1), (7.4), (7.5), \tag{7.25b}$$

$$x_0 = x_s, x_N = x_g, x_k \in \mathcal{X}, u_k \in \mathcal{U}, 0 \leq f_{k,ni} \leq f_u \tag{7.25c}$$

where $x_k = [\theta_k, P^O_{k,y}, \dot{\theta}_k, \dot{P}^O_{k,y}]^\top$, $u_k = [f_{k,nP}, f_{k,tP}]^\top$, $f_k = [f_{k,nA}, f_{k,nB}]^\top$, $Q = Q^\top \geq 0, R = R^\top > 0$. The input of (7.25) consists of physical parameters such as mass, length, and width of the object and the optimization parameters such as Q and R. The output of (7.25) consists of trajectories of $x_k, u_k, f_k, \forall k \in \{0, 1, \dots, N\}$. We use explicit Euler to discretize the dynamics with sample time Δ. The function FK represents forward kinematics to specify each contact point i and CoM location. $\mathcal{X}$ and $\mathcal{U}$ are convex polytopes, consisting of a finite number of linear inequality constraints. f_u is an upper-bound of normal force at each contact point. Note that we impose (7.1), (7.4) at each time step k. x_s, x_g are the states at $k = 0, k = N$, respectively.

7.4.2 Robust Bilevel Contact-Implicit Trajectory Optimization

In this section, we present our formulation where we incorporate frictional stability in trajectory optimization to obtain robustness. In particular, we first focus on discussing the

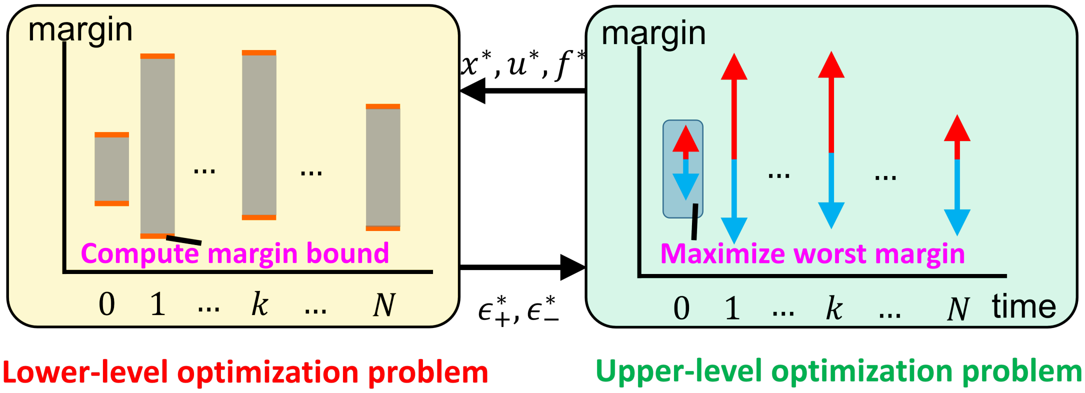

margin
Compute margin bound
0
1
...
k
...
N
x^*, u^*, f^*
$\epsilon_+^*, \epsilon_-^*$
margin
Maximize worst margin
0
1
...
k
...
N
time
Lower-level optimization problem
Upper-level optimization problem

Figure 7.7: This figure illustrates the idea of the proposed contact implicit bilevel optimization, CIBO. Given the trajectory of x, u, f, the stability margin over the trajectory can be computed as shown in lower-level optimization problem. Then, given the computed stability margin over the trajectory ϵ, the upper-level optimization problem maximizes the worst-case stability margin over the trajectory by optimizing the trajectory of x, u, f. Our CIBO simultaneously optimizes the lower-level optimization problem and the upper-level optimization problem. In the right plot, red and blue arrows represent the stability margin along positive and negative directions, respectively. Our CIBO optimizes the stability margin for each direction.

optimization problem with uncertain mass, CoM location, and finger contact location. We later discuss the optimization problem of uncertain coefficient of friction in Sec 7.4.3.

An important point to note is that the optimization problem would be ill-posed if we naively add (7.7), (7.12), and/or (7.19) to (7.25) since there is no u to satisfy all uncertainty realization in equality constraints [149]. Therefore, our strategy is that we plan to find an optimal nominal trajectory that can ensure external contacts under uncertain physical parameters. In other words, we aim at maximizing the worst-case stability margin over the trajectory given the maximal frictional stability at each time-step k (also shown in Fig. 7.6). Thus, we maximize the following objective function:

$$\min_k \epsilon_{k,+}^* - \max_k -\epsilon_{k,-}^* \tag{7.26}$$

where $\epsilon_{k,+}^*, \epsilon_{k,-}^*$ are non-negative variables. Note that $\epsilon_{k,+}^*, \epsilon_{k,-}^*$ are the largest uncertainty in the positive and negative direction, respectively, at instant k given x, u, f, which results in non-zero contact forces (i.e., stability margin, see also Fig. 7.6). (7.26) calculates the smallest stability margin over time-horizons by subtracting the stability margin along the positive direction from that along the negative direction. Hence, we formulate a bilevel optimization problem which consists of two lower-level optimization problems as follows (see also Fig. 7.7):

$$\max_{x,u,f,\epsilon_+^*,\epsilon_-^*} (\min_k \epsilon_{k,+}^* - \max_k -\epsilon_{k,-}^*) \tag{7.27a}$$

$$\text{s. t.} \quad (7.25b), (7.25c), \tag{7.27b}$$

$$\epsilon_{k,+}^* \in \underset{\epsilon_{k,+}}{\operatorname{argmax}}\{\epsilon_{k,+} : A_k\epsilon_{k,+} \leq b_k, \epsilon_{k,+} \geq 0\}, \tag{7.27c}$$

$$\epsilon_{k,-}^* \in \underset{\epsilon_{k,-}}{\operatorname{argmax}}\{\epsilon_{k,-} : -A_k\epsilon_{k,-} \leq b_k, \epsilon_{k,-} \geq 0\} \tag{7.27d}$$

where $A_k \in \mathbb{R}^{2\times1}, b_k \in \mathbb{R}^{2\times1}$ represent inequality constraints in (7.10) and (7.11) or (7.22) and (7.23) if the object is on a slope. $A_k\epsilon_{k,+} \leq b_k, \epsilon_{k,+} \geq 0$, and $-A_k\epsilon_{k,-} \leq b_k, \epsilon_{k,-} \geq 0$ represent the lower-level constraints for each lower-level optimization problem while (7.25b), (7.25c) represent the upper-level constraints. ϵ_+, ϵ_- are the lower-level objective functions while $\min_k \epsilon_{k,+}^* - \max_k -\epsilon_{k,-}^*$ is the upper-level objective function. $\epsilon_{k,+}, \epsilon_{k,-}$ are the lower-level decision variables of each lower-level optimization problem while $x, u, f, \epsilon_+^*, \epsilon_-^*$ are the upper-level decision variables.

(7.27) considers the largest one-side frictional stability margin along positive and negative direction at k. Therefore, by solving these two lower-level optimization problems, we are able to obtain the maximum frictional stability margin along positive and negative direction. The advantage of (7.27) is that since the lower-level optimization problem are formulated as two linear programming problems, we can efficiently solve the entire bilevel optimization problem using the Karush-Kuhn-Tucker (KKT) condition as follows:

$$w_{k,+,j}, w_{k,-,j} \geq 0, C_k\epsilon_{k,+} \leq d_k, E_k\epsilon_{k,-} \leq d_k, \tag{7.28a}$$

$$w_{k,+,j}(C_k\epsilon_{k,+} - d_k)_j = 0, \tag{7.28b}$$

$$w_{k,-,j}(E_k\epsilon_{k,-} - d_k)_j = 0, \tag{7.28c}$$

$$\nabla(-\epsilon_{k,+}) + \sum_{j=1}^{3} w_{k,+,j}\nabla(C_k\epsilon_{k,+} - d_k)_j = 0, \tag{7.28d}$$

$$\nabla(-\epsilon_{k,-}) + \sum_{j=1}^{3} w_{k,-,j}\nabla(E_k\epsilon_{k,-} - d_k)_j = 0 \tag{7.28e}$$

where $C_k = [A_k^\top, -1]^\top \in \mathbb{R}^{3\times1}, d_k = [b_k^\top, 0]^\top \in \mathbb{R}^{3\times1}, E_k = [-A_k^\top, -1]^\top \in \mathbb{R}^{3\times1}$. $w_{k,+,j}$ is Lagrange multiplier associated with $(C_k\epsilon_{k,+} \leq d_k)_j$, where $(C_k\epsilon_{k,+} \leq d_k)_j$ represents the j-th inequality constraints in $C_k\epsilon_{k,+} \leq d_k$. $w_{k,-,j}$ is Lagrange multiplier associated with $(E_k\epsilon_{k,-} \leq d_k)_j$. Using the KKT condition and epigraph trick, we eventually obtain a single-level large-scale nonlinear programming problem with complementarity constraints:

$$\max_{x,u,f,\epsilon_+^*,\epsilon_-^*} (t_+ + \alpha t_-) \tag{7.29a}$$

$$\text{s. t.} \quad (7.25b), (7.25c), (7.28), \tag{7.29b}$$

$$t_+ \leq \epsilon_{k,+}, t_- \leq \epsilon_{k,-}, \forall k \tag{7.29c}$$

where α is a weighting scalar. Note that we derive (7.29) for the case with an uncertain mass parameter but this formulation can be easily converted to the case where uncertainty exists in CoM location by replacing A_k, b_k in (7.27) with (7.13). Similarly, we can consider uncertainty in finger contact location by replacing A_k, b_k in (7.27) with (7.19). Therefore, by solving tractable (7.29), we can efficiently generate robust trajectories that are robust against uncertain mass, CoM location, and contact location parameters.

Remark 1: If we consider the case where uncertainty exists in both mass and CoM location simultaneously, we would have a nonlinear coupling term $(C_x+r)(mg+\epsilon)$ in quasi-static equilibrium of moment. This makes the lower-level optimization non-convex optimization, making it extremely challenging to solve during bilevel optimization.

7.4.3 Robust Bilevel Contact-Implicit Trajectory Optimization under Frictional Uncertainty

We consider the case where the system has uncertainty in the friction coefficients at A and B as discussed in Sec 7.3.4. In order to design a robust open-loop controller for the system, we can use the similar formulation presented in Sec 7.4.2. The proposed formulation aims at maximizing the stability margin from stochastic friction. In particular, to avoid non-convex optimization as the lower-level optimization problem, we consider the stability margin along positive and negative direction for both ϵ_A and ϵ_B, as we discuss in Sec 7.4.2. By borrowing the optimization problem (7.27), the proposed formulation can be seen as follows. For simplicity, we abbreviate subscript k.

$$\max_{x,u,f,\epsilon^*_{A,+},\epsilon^*_{A,-},\epsilon^*_{B,+},\epsilon^*_{B,-}} \sum_{c\in\mathcal{C}}(\min_k \epsilon^*_{c,+} - \max_k -\epsilon^*_{c,-}) \tag{7.30a}$$

$$\text{s. t.} \quad (7.25\text{b}), (7.25\text{c}), \tag{7.30b}$$

$$\epsilon^*_A \in [-\epsilon^*_{A,-}, \epsilon^*_{A,+}], \epsilon^*_B \in [-\epsilon^*_{B,-}, \epsilon^*_{B,+}], \tag{7.30c}$$

$$\epsilon^*_{A,+} \in \underset{\epsilon_{A,+}}{\operatorname{argmax}}\{\epsilon_{A,+} : g(x,u,f,\epsilon_{A,+},\epsilon^*_B) \le 0, \quad \epsilon_{A,+} \ge 0\}, \tag{7.30d}$$

$$\epsilon^*_{A,-} \in \underset{\epsilon_{A,-}}{\operatorname{argmax}}\{\epsilon_{A,-} : g(x,u,f,-\epsilon_{A,-},\epsilon^*_B) \le 0, \quad \epsilon_{A,-} \ge 0,\}, \tag{7.30e}$$

$$\epsilon^*_{B,+} \in \underset{\epsilon_{B,+}}{\operatorname{argmax}}\{\epsilon_{B,+} : g(x,u,f,\epsilon_{B,+},\epsilon^*_A) \le 0, \quad \epsilon_{B,+} \ge 0,\}, \tag{7.30f}$$

$$\epsilon^*_{B,-} \in \underset{\epsilon_{B,-}}{\operatorname{argmax}}\{\epsilon_{B,-} : g(x,u,f,-\epsilon_{B,-},\epsilon^*_A) \le 0, \quad \epsilon_{B,-} \ge 0\}, \tag{7.30g}$$

where g summarizes the constraints for each lower-level optimization problem and $\mathcal{C} = \{A, B\}$. For each lower-level optimization problem, we consider that another uncertain friction is in the range of optimal stability margin. For instance, (7.30d) is one of the four lower-level optimization problems which aims at maximizing the stability margin under

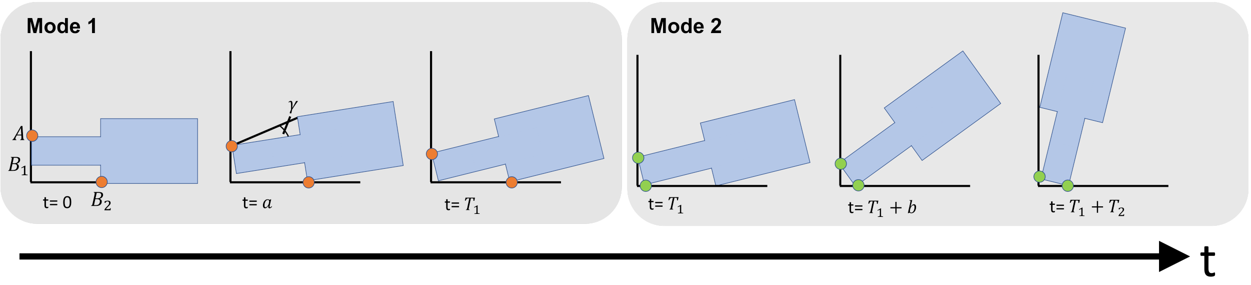


Figure 7.8: A schematic of pivoting for a non-convex shape object where contact set changes over time. During mode 1, the peg rotates with contact at A and B_2. During mode 2, the peg rotates with contact at A and B_1. γ represents one of the kinematic features of peg, which is used to discuss the result in Sec 7.9.

stochastic friction forces at A, given stochastic friction force at B, ϵ_B^*. (7.30c) ensures that ϵ_B^* needs to be within the range of stability margin computed from other two lower-level optimization problems (7.30f) and (7.30g).

The resulting optimization introduces many complementarity constraints through the KKT condition because of four lower-level optimization problems, but the resulting computation is still tractable. We discuss computational results in Sec 7.5.10.

7.4.4 Robust Bilevel Optimization over Mode Sequences for Non-Convex Objects

The method introduced in the previous subsections assumes convex geometry of the object being manipulated and can not be applied to objects with non-convex geometry (such as pegs as shown in Fig. 7.1). This is because non-convex objects could result in different contact formations between the object and the environment and it is not trivial to identify a feasible contact sequence. In [131], the proposed optimization (7.29) was solved sequentially for pegs with non-convex geometry. As illustrated in Fig. 7.8, we first solve the optimization for a particular contact set (i.e., mode 1 in Fig. 7.8) and then solve the optimization for another contact set (i.e., mode 2 in Fig. 7.8) given the solution obtained from the first optimization. While this method works, it requires extensive domain knowledge. We observed that the second stage optimization can result in infeasible solutions given the solution from the first stage optimization. Thus, we had to carefully specify the parameters of optimization and, in particular, the initial state and terminal state constraints. Such a hierarchical approach has difficulty in finding a feasible solution once the object becomes more complicated.

To overcome these issues, in general, complementarity constraints can be used to model the change of contact. However, introducing complementarity constraints inside the lower-level optimization makes the lower-level optimization non-convex optimization. Hence, the KKT condition is not a necessary and sufficient condition for optimality but rather a necessary condition. Thus, it is not guaranteed to find globally optimal safety margins over the trajectory.

In this work, we propose another approach to deal with the non-convexity of the object. Inspired by [16], we formulate the optimization that optimizes the trajectory given mode sequences instead of optimizing mode sequences. It is worth noting that our framework still optimizes the trajectory over the time duration of each mode given the sequence of the mode. Our goal is that the optimization has a larger feasible space so that less domain knowledge is required.

Using the formulation presented in [16], we present a mode-based formulation for non-convex shaped objects. See [16] for more details regarding mode-based optimization. For simplicity of exposition, we only present the formulation considering two modes. But one can easily extend this to problems with multiple modes. For each contact mode, the system has the different constraints. For brevity, we abbreviate the subscript k:

$$i_x, i_y \in FK_m(\theta_k, P^O_{k,y}), \forall i \in \{A, B_m\} \tag{7.31a}$$

$$g_m(f_{nA}, f_{tA}, f_{nB_1}, f_{tB_1}, f_{nP}, f_{tP}, P^O_y) \text{ if } m = 1 \tag{7.31b}$$

$$g_m(f_{nA}, f_{tA}, f_{nB_2}, f_{tB_2}, f_{nP}, f_{tP}, P^O_y) \text{ if } m = 2 \tag{7.31c}$$

$$f_{tA} = \mu_A f_{nA}, f_{tB_1} = -\mu_{B_1} f_{nB_1}, f_{tB_2} = -\mu_{B_2} f_{nB_2} \tag{7.31d}$$

$$(7.5), x_k \in \mathcal{X}, u_k \in \mathcal{U}, 0 \leq f_{k,ni} \leq f_u \tag{7.31e}$$

where $m \in \{1, 2\}$ to represent each contact mode. g_m represents the quasi-static model of pivoting manipulation for mode m. It is worth noting that since we decompose the optimization problem into the two mode optimization problem, complementarity constraints are encoded for each mode.

What the optimization problem needs to perform is that for each mode, it only considers the associated constraints and does not consider constraints associated with different contact mode. For example, during mode 1, the optimization should consider only constraints associated with mode 1 and should not consider constraints such as (7.31c). Another thing the optimization needs to consider is that it needs to scale $\dot{\theta}, \dot{P}_y^O$ since we would like to optimize over the time duration. To achieve that, we employ the scaled time variables as discussed in [16]. As a result, we recast the quasi-static model by introducing a new state variable with a scaled time, $\tilde{x}_k = \left[\theta_k, P^O_{k,y}, \frac{\dot{\theta}_k}{T}, \frac{\dot{P}^O_{k,y}}{T}\right]^\top$ where $T = T_1$ during mode 1 and $T = T_2$ during mode 2.

For two contact modes, we can remodel our optimization (7.25) as follows:

$$\min_{\tilde{x},u,f} \sum_{k=0}^{N-1} (\tilde{x}_k - x_g)^\top Q(\tilde{x}_k - x_g) + u_k^\top R u_k + \sum_{l=1}^{2} T_l \tag{7.32a}$$

$$\text{s. t. } h_1(\tilde{x}_k, u_k, f_k) \leq 0, \text{for } k\Delta \leq 1 \tag{7.32b}$$

$$h_2(\tilde{x}_k, u_k, f_k) \leq 0, \text{for } k\Delta > 1 \tag{7.32c}$$

where $\tilde{x}_k = \left[\theta_k, P^O_{k,y}, \frac{\dot{\theta}_k}{T_1}, \frac{\dot{P}^O_{k,y}}{T_1}\right]^\top$ for $k\Delta \leq 1$ and $\tilde{x}_k = \left[\theta_k, P^O_{k,y}, \frac{\dot{\theta}_k}{T_2}, \frac{\dot{P}^O_{k,y}}{T_2}\right]^\top$ for $k\Delta > 1$. We use h_1 and h_2 to represent all constraints for each mode. Given (7.32), we can obtain bilevel optimization formulation for non-convex shape objects by following the logic in Sec 7.4.2.

7.4.5 Robust Bilevel Contact-Implicit Trajectory Optimization with Patch Contact

The formulation for robust CIBO is similar to the point contact case except that the underlying equilibrium conditions are different. The quasi-static equilibrium conditions for the patch contact case were earlier presented in (7.24). Using these equations and the analysis presented in Sections 7.3.2 through 7.3.4, it is straightforward to compute the constraints for the corresponding robust CIBO similar to (7.27). More explicitly, this can be achieved by

Table 7.2: Parameters of objects. m, l, w represent the mass, length, and the width of the object, respectively. For pegs, the first element in l, w are l_1, w_1 and the second element in l, w are l_2, w_2, respectively, shown in Fig. 7.20. For pegs, since they are made of the same material and they make contact on the same environment, we can assume $\mu_B = \mu_{B_1} = \mu_{B_2}$.

	m [g]	l [mm]	w [mm]	μ_A, μ_B, μ_P
gear 1	140	84	20	0.3, 0.3, 0.8
gear 2	100	121	9.5	0.3, 0.3, 0.8
gear 3	280	84	20	0.3, 0.3, 0.8
peg 1	45	36, 40	20, 28	0.3, 0.3, 0.8
peg 2	85	28, 40	10, 11	0.3, 0.3, 0.8
peg 3	85	28, 40	10, 27.5	0.3, 0.3, 0.8

Table 7.3: Worst-case stability margin over the control horizon obtained from optimization for gear 1. Note that the stability margin for the solution of the benchmark optimization is analytically calculated.

	ϵ_+^*, ϵ_-^* [N]	r_+^*, r_-^* [mm]
Benchmark optimization (7.25)	0.10, 0.66	1.5, 0.85
Ours (7.29) with mass uncertainty	0.34, 0.50	N/A
Ours (7.29) with CoM uncertainty	N/A	3.43, 2.70

computing the appropriate constraints of the type $A_k \epsilon_{k,+} \leq b_k$ and $-A_k \epsilon_{k,-} \leq b_k$ using (7.24) and the frictional stability margin discussion in Sec 7.3.7.

7.5 Experimental Results

In this section, we verify the performance of our proposed approach for pivoting. Through the experiments we present in this section, we evaluate the efficacy of the proposed planner in several different settings and the computational requirement of the method. We also present results of implementation of the proposed planner on a robotic system using a 6 DoF manipulator arm and compare it against a baseline trajectory optimization method.

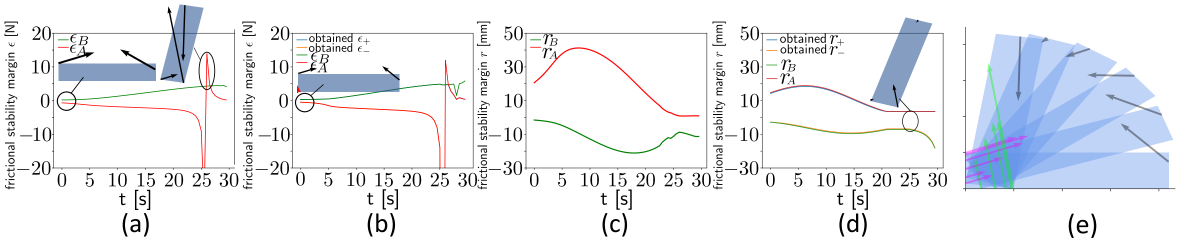


Figure 7.9: Trajectory of frictional stability margin. ϵ_A, ϵ_B are bounds of ϵ from (7.10), (7.11). r_A, r_B are bounds of r from (7.13). $\epsilon_+, \epsilon_-, r_+, r_i$ are solutions obtained from CIBO. (a), (b): Trajectory of frictional stability of gear 1 based on uncertain mass obtained from baseline optimization, our CIBO, respectively. (c), (d): Trajectory of frictional stability of gear 1 based on uncertain CoM location obtained from baseline optimization, CIBO, respectively. (e): Snapshots of pivoting motion for gear 1 obtained from CIBO considering uncertain CoM location.

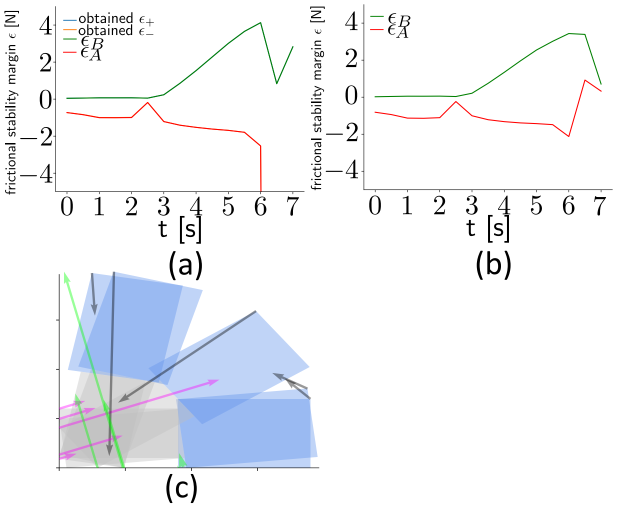


Figure 7.10: (a), (b): Trajectory of frictional stability margin of peg 1 based on uncertain mass obtained from CIBO, baseline optimization, respectively. Note that here we solve CIBO sequentially for each mode (i.e., hierarchical planning), instead of using the proposed mode-sequence-based optimization. (c): Snapshots of pivoting motion for peg 1, obtained from CIBO considering uncertain mass.

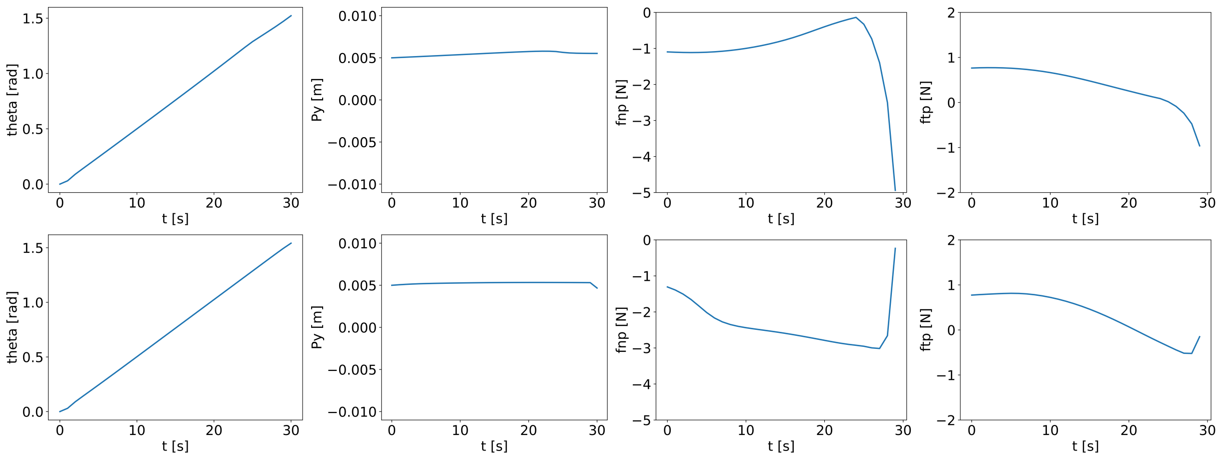


Figure 7.11: We show the time history of object angle, finger position, and contact forces from a manipulator during pivoting of gear 1. The top row shows the result using CIBO (7.29) considering CoM uncertainty and the bottom one shows the result using (7.25) (i.e., it does not consider robustness criteria in the formulation explicitly.). The top row results and the bottom row results are used in visualizing the stability margin in Fig. 7.9 (d), (c), respectively.

Table 7.4: Obtained worst stability margins over the time horizons from optimization for peg 1. Note that the stability margin for the solution of the benchmark optimization is analytically calculated.

	ϵ^*_+, ϵ^*_- [N]	r^*_+, r^*_- [mm]
Benchmark optimization (7.25)	0.035, 0.018	31, 0
Ours (7.29) with mass uncertainty	0.050, 0.021	N/A
Ours (7.29) with CoM uncertainty	N/A	38, 0

7.5.1 Experiment Setup

We implement our method in Python using IPOPT solver [10] with PYROBOCOP wrapper [16]. We use HSL MA86 [150] as a linear solver for IPOPT. The optimization problem is implemented on a computer with Intel i7-12800K.

We demonstrate our algorithm on several different objects, as detailed in Table 7.2. During optimization, we set $Q = \text{diag}(0.1, 0)$, $R = \text{diag}(0.01, 0.01)$. We use $\alpha = 1$ when we run (7.29). We set $x_s = [0, \frac{w}{4}]^\top, \theta_g = \frac{\pi}{2}$. Note that we only enforce terminal constraints for convex shape objects. For non-convex shape objects, we do not enforce terminal constraints since the peg cannot achieve $\theta_N = \frac{\pi}{2}$ unless we consider another contact mode (see Fig. 7.8). In PYROBOCOP wrapper, we did warm-start for the state at $k = 0, N$ by setting initial and terminal states as initial guesses. We did not explicitly conduct a warm-start for other decision variables and we set them to 0.

We use a Mitsubishi Electric Factory Automation (MELFA) RV-5AS-D Assista 6 DoF arm (see Fig. 7.1) for the experiments. The robot has a pose repeatability of ±0.03mm. The robot is equipped with Mitsubishi Electric F/T sensor 1F-FS001-W200 (see Fig. 7.1). To implement the computed force trajectory during manipulation, we use the default stiffness controller for the robot. By selecting an appropriate stiffness matrix [151], we design a reference trajectory that would result in the desired interaction force required for manipulation [152, 153]. Note that this trajectory is implemented in open-loop and we do not design a controller to ensure that the computed force trajectory is precisely tracked during execution.

Explain MPC setting: Devesh could you work on this? The object states are tracked using AprilTag[154]. The robot states are tracked using the robot's joint encoders. The contact states at contact A, B, P in Fig. 7.2 are estimated using the object state, the robot state, and the known geometry of the object.

7.5.2 Results of Bilevel Optimization for Uncertain Mass and CoM Parameters

Fig. 7.9 shows the trajectory of frictional stability margin of gear 1 obtained from the proposed robust CIBO considering uncertain mass and uncertain CoM location, and the benchmark optimization. Overall, CIBO could generate more robust trajectories. For example, at $t = 0$ s, f_{nB} in (a) is almost zero so that the stability margin obtained from (7.11) is almost zero. In contrast, CIBO could realize non-zero f_{nB} as shown as a red arrow in (b). In (d), to increase the stability margin, the finger position P_y^O moves on the face of gear 1 so that the controller can increase the stability margin more than the benchmark optimization. This would not happen if we do not consider complementarity constraints (7.5). Also, our obtained $\epsilon_+, \epsilon_-, r_+, r_-$ follows bounds of stability margin. It means that CIBO can successfully design a controller that maximizes the worst stability margin given the best stability margin for each time-step.

Fig. 7.11 shows that both the benchmark and CIBO actually change the finger position P_y^O by considering complementarity constraints (7.5). In fact, we observed that at $t = 25$ s, P_y^O in both results moves to the negative value to maintain the stability of the object. In practice, we are unable to find any feasible solutions with fixed P_y^O, instead of using (7.5). Thus, (7.5) is critically important to find a feasible solution.

Next, we discuss how much CIBO improves the worst-case stability margin. The trajectories of f_{nP} in Fig. 7.11 show that the magnitude of f_{nP} from CIBO increase at $t = 25$ s to improve the worst-case stability margin. On the other hand, f_{nP} from the benchmark optimization does not increase at $t = 25$ s. Hence, we verify that by increasing normal force, the robot could successfully robustify the pivoting manipulation. This result can be also understood in Fig. 7.9 (c) and (d) where the stability margin in (d) at $t = 25$ s is larger than that in (c), as discussed above.

Table 7.3 and Table 7.4 summarize the computed stability margin from Fig. 7.9. In Table 7.3, for the case where CIBO considers uncertainty of mass, we observe that the value of ϵ_-^* from CIBO is smaller than that from the benchmark optimization although the sum of the stability margin $\epsilon_+^* + \epsilon_-^*$ from CIBO is greater than that from the benchmark optimization. This result means that CIBO can actually improve the worst-case performance by sacrificing the general performance of the controller. Regarding the case where we consider the uncertain CoM location, CIBO outperforms the benchmark trajectory optimization in both r_+^*, r_-^*. For peg 1, the bilevel optimizer without using mode sequence-based optimization (i.e., hierarchical optimization) finds trajectories that have larger stability margins for both uncertain mass and CoM location as shown in Table 7.4. The trajectory of stability margin obtained from CIBO considering mass uncertainty is illustrated in Fig. 7.10. We discuss the results using CIBO with mode-sequence based optimization in Sec 7.5.7.

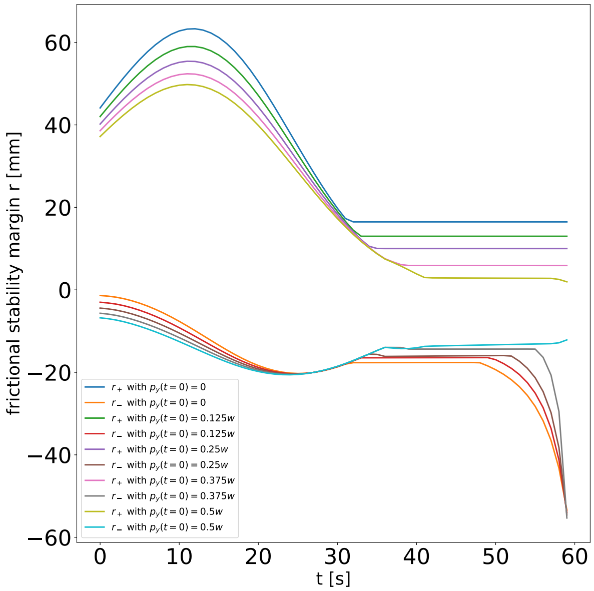


Figure 7.12: Time history of frictional stability margin considering CoM location with different initial manipulator position $P_y^O(t=0)$.

7.5.3 Results of Bilevel Optimization with Different Manipulator Initial State

We believe that the efficiency of the optimization depends on the initial location of the manipulator finger. This is because the stability margin depends on the manipulation finger location, which is partially governed by its location at $t = 0$. Thus we present some results by randomizing over the manipulator finger location at $t = 0$. We sample initial state of finger position $P_y^O(t = 0)$ from a discrete uniform distribution with the range of $P_y^O(t = 0) \in [-0.5w, -0.375w, -0.25w, -0.125w, \ldots, 0.5w]$. Then we run CIBO considering CoM location uncertainty.

Fig. 7.12 illustrates the time history of stability margin with different $P_y^O(t = 0)$. CIBO is not able to find feasible solutions with $P_y^O(t = 0) < 0$. It makes sense since there may not be enough moment for the desired motion if $P_y^O(t = 0) < 0$.

Fig. 7.12 shows that different $P_y^O(t = 0)$ leads to different stability margin over the time horizon. Table 7.5 summarizes the worst-case stability margin over the trajectory obtained from Fig. 7.12. Table 7.5 also shows that the worst-case stability margin is different with different $P_y^O(t = 0)$. Finding a good $P_y^O(t = 0)$ is not trivial and it requires domain

Table 7.5: Computed worst-case stability margin considering uncertain CoM location with different P_y^O at $t = 0$ over the control horizon obtained from optimization for gear 1.

	r_+^*, r_-^* [mm]
Ours with $P_y^O(t = 0) = 0$	16.47, 1.36
Ours with $P_y^O(t = 0) = 0.125w$	12.99, 2.98
Ours with $P_y^O(t = 0) = 0.25w$	10.00, 4.41
Ours with $P_y^O(t = 0) = 0.375w$	5.94, 5.67
Ours with $P_y^O(t = 0) = 0.5w$	1.94, 6.77

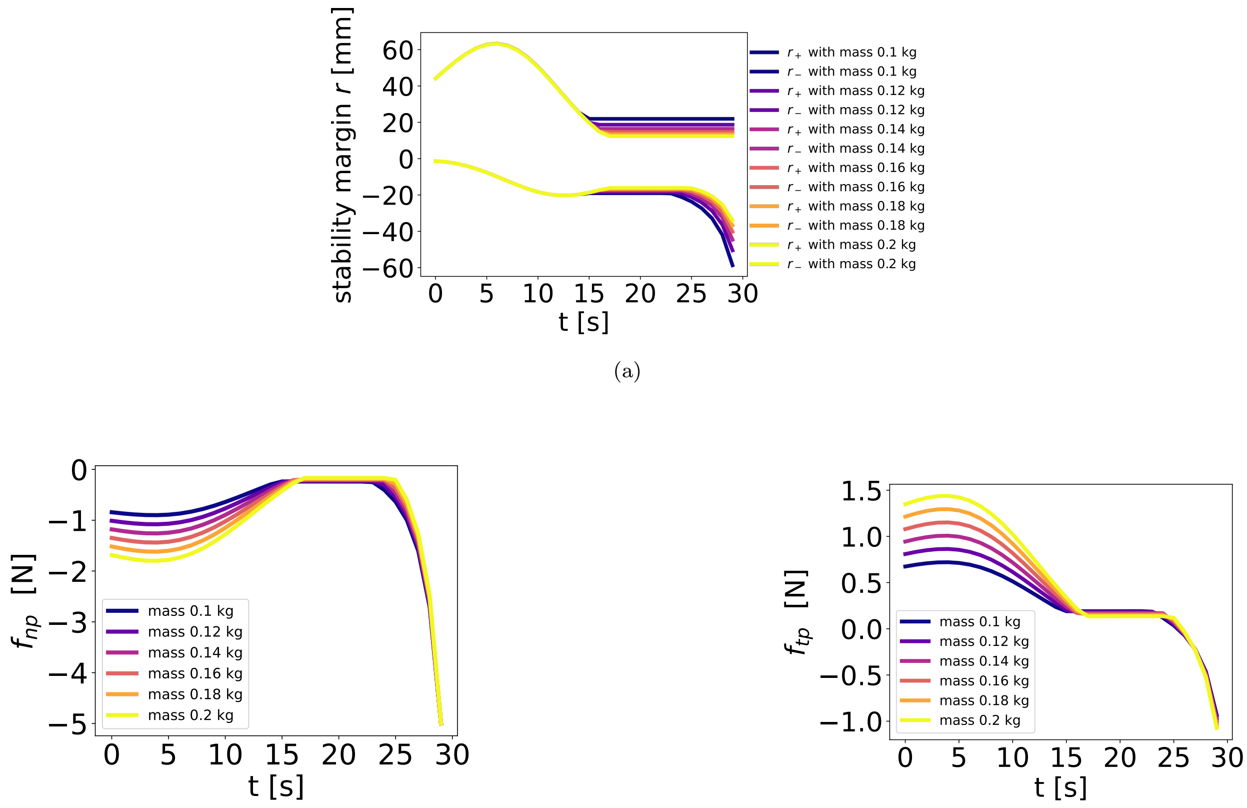
knowledge. Thus, ideally, we should formulate CIBO where $P_y^O(t = 0)$ is also a decision variable so that the solver can optimize the trajectory over $P_y^O(t = 0)$ as well.

Since CIBO is non-convex optimization, it is still possible that a feasible solution exists for $P_y^O(t = 0) < 0$. However, we can at least argue that it is much more difficult to find a feasible solution with $P_y^O(t = 0) < 0$ than that with $P_y^O(t = 0) \geq 0$.

7.5.4 Results of Bilevel Optimization for Uncertain CoM parameters with Different Mass and Friction of Object

We first study how stability margin with uncertain CoM location changes with different mass parameters. We sample the mass of the object from a discrete uniform distribution with range of $m \in [0.1, 0.12, 0.14, 0.16, 0.18, 0.2]$ kg. Then we run CIBO considering CoM location uncertainty.

Fig. 7.13 shows the time history of stability margin and contact forces over the time horizon. For this analysis, the projection of CoM lies on the contact B (i.e., $C_x^B = 0.$) at $t = 15$ s. At $t \in [0, 15]$ s (i.e., $C_x^B > 0$), the robot has to execute the contact forces to support the object against gravity. In fact, Fig. 7.13b and Fig. 7.13c show that the contact forces increase as mass increases. Since other parameters of the system are the same, the CIBO designs the trajectory whose stability margin is the same with different mass by changing the contact forces from the robot. At $t \in [15, 30]$ s, the upper-bound of stability margin r_+



(a)

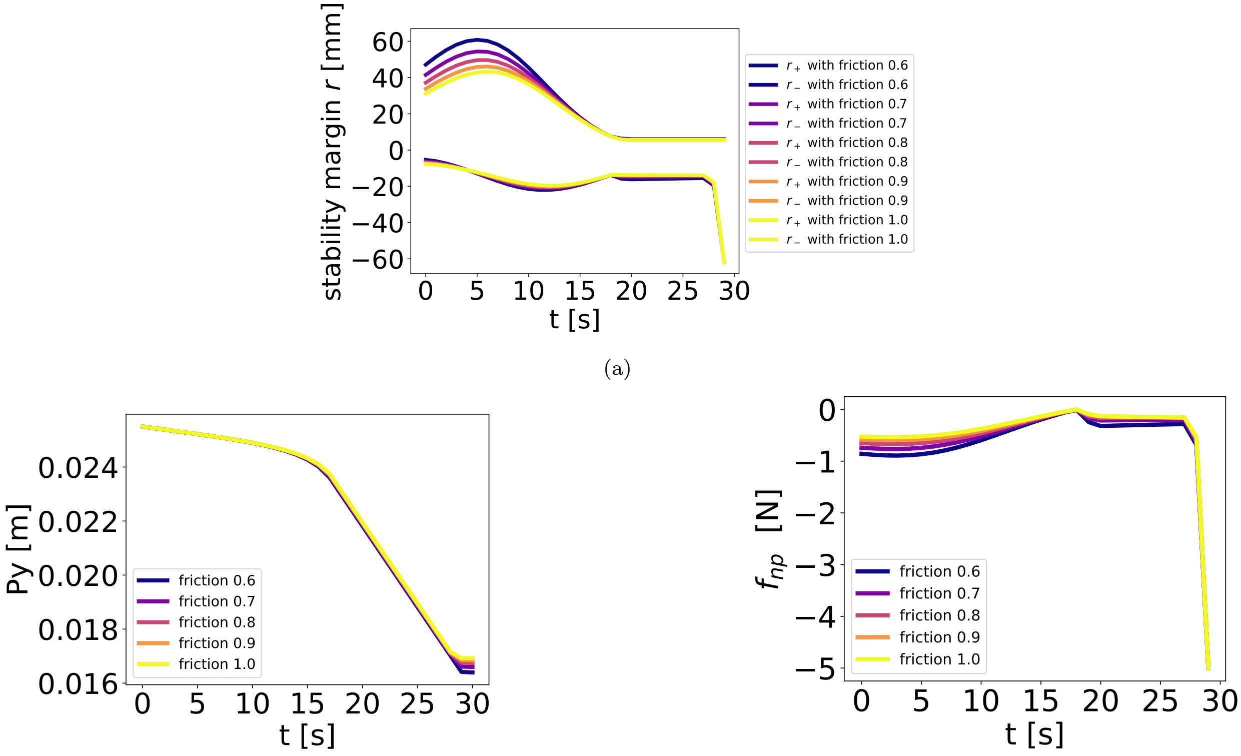
(b)

(c)

Figure 7.13: (a): Time history of stability margin considering CoM location with different mass. The trajectory with the same color means that the same mass is used in the CIBO. The trajectories where $r > 0$ are the trajectories of r_+ and the the trajectories where $r < 0$ are the trajectories of r_-. (b): Time history of f^O_{nP}. (c): Time history of f^O_{tP}.

shows the larger value with the lighter object, and the lower-bound of stability margin r_- also shows the larger value with the lighter mass of the object. This makes sense because as the object becomes lighter, the system allows for a longer moment arm r in quasi-static equilibrium.

Second, we study how stability margin with uncertain CoM location changes with different coefficients of friction between the object and the robot finger (i.e., μ_P at contact P in Fig. 7.2). We sample the friction of the object from a discrete uniform distribution with a range of $\mu_P \in [0.6, 0.7, 0.8, 0.9, 1.0]$. Then we run CIBO considering CoM location uncertainty.

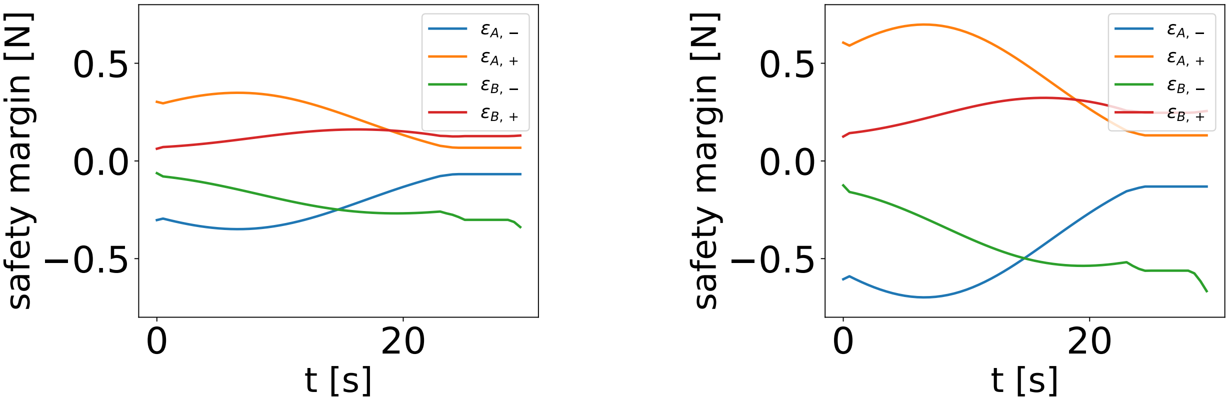


(b)

(c)

Figure 7.14: (a): Time history of stability margin considering CoM location with different friction at P. The trajectory with the same color means that the same mass is used in the CIBO. The trajectories where $r > 0$ are the trajectories of r_+ and the the trajectories where $r < 0$ are the trajectories of r_-. (b): Time history of P^O_y. (c): Time history of f^O_{nP}.

Fig. 7.14 shows the time history of stability margin, finger contact location P^O_y, and the contact normal force f^O_{nP} over the time horizon. We observe that the different friction leads to different trajectories of the stability margin. In particular, we observe that the CIBO considering the lower μ_P results in a larger r_+. As Fig. 7.14b, the finger keeps moving during the manipulation to complete the pivoting. It means that the complementarity constraints at P (7.5) are always equality constraints like (7.4), $f^O_{tP} = \mu_P f^O_{nP}$. With the small μ_P, the robot can execute the large f^O_{nP} with the small f^O_{tP}, which is beneficial at $t \in [0, 18]$ s to avoid losing the contact A, before the projection of CoM lies on the contact B.



(a)

(b)

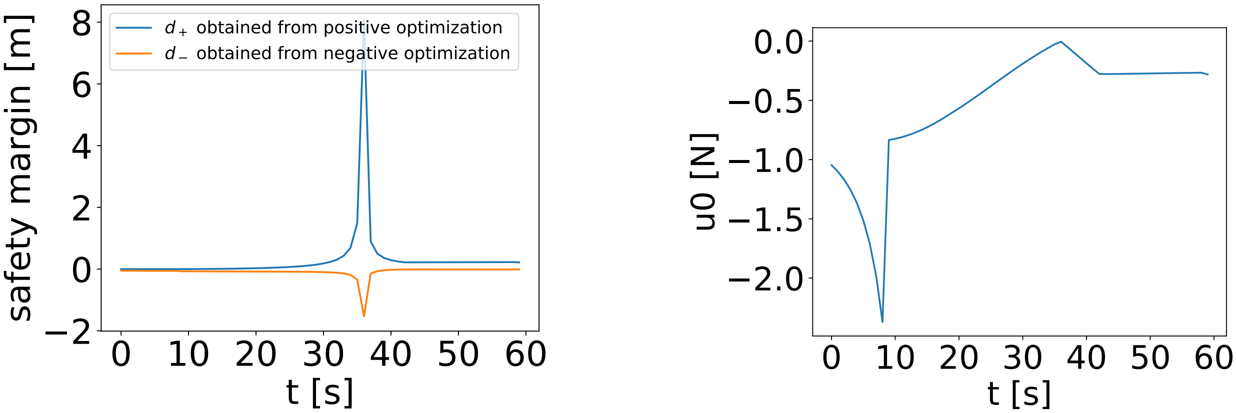
Figure 7.15: Trajectory of frictional stability margin of (a) gear 1 and (b) gear 3, based on uncertain friction obtained from CIBO (7.30), respectively.

7.5.5 Results of Bilevel Optimization for Uncertain Friction Parameters

Fig. 7.15 shows the time history of frictional stability margin of gear 1 and gear 3 using (7.30). CIBO could successfully design an optimal open-loop trajectory by improving the worst-case performance of stability margin. We observe that Fig. 7.15 (b) shows a larger stability margin compared to (a). This result makes sense since in (b), we consider gear 3 whose weight is heavier than the weight of gear 1 and thus we get stability margins which are bigger than those obtained for (a).

7.5.6 Results of Bilevel Optimization for Uncertain Finger Contact Location

In this section, we present results for pivoting manipulation under uncertain finger contact location. Fig. 7.16 shows the time history of the stability margin of gear 2 using (7.29). Our CIBO could successfully design a controller for an uncertain contact location. Also, Fig. 7.16 shows that stability margin has a quite large value at $t = 37$ s. At $t = 37$ s, the controller makes the finger move with zero normal force, resulting in a large stability margin as we explain in Sec 7.3.5.



(a)

(b)

Figure 7.16: We consider CIBO with uncertain finger contact location. (a): Time history of frictional stability margin. (b) Time history of normal force at the finger.

7.5.7 Results of Bilevel Optimization over Mode Sequences for Non-Convex Objects

In this section, we present results for objects with non-convex geometry using the mode-based optimization presented in Section 7.4.4. Fig. 7.17 shows the time history of states, control inputs, and frictional stability margins for pegs whose geometry are non-convex and the contact sets change over time. First of all, we can observe that CIBO in (7.32) could successfully optimize the stability margin over trajectory while it optimizes the time duration of each mode. We observe that $\frac{T_1}{T_1+T_2}$ (i.e., the ratio of mode 1 over the horizon) of peg 2 is much smaller than that of peg 3 since γ (see Fig. 7.8 for the definition of γ) of peg 2 is smaller than that of peg 3 and thus, it spends less time in mode 1. Fig. 7.17 shows that f_{tP} of peg 3 dramatically changes at $t = T_1$ s while that of peg 1 does not. In contrast, the shape of peg 2 has smaller γ (i.e., less non-convex shape) and it can be regarded as a rectangle shape. Thus, the effect of contact mode is less, leading to a smaller change of f_{tP} at $t = T_1$ s.

In order to show that we can find solutions much more effortlessly using (7.32) compared to two-stage optimization (that was earlier used in [131]), we sample 20 different p_y at $t = 0$ s and count the number of times the benchmark two-stage optimization problem and the

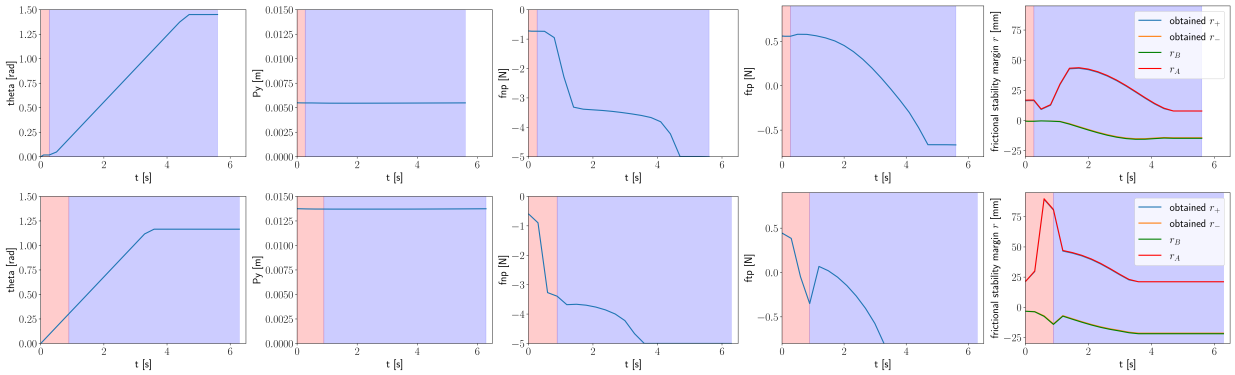


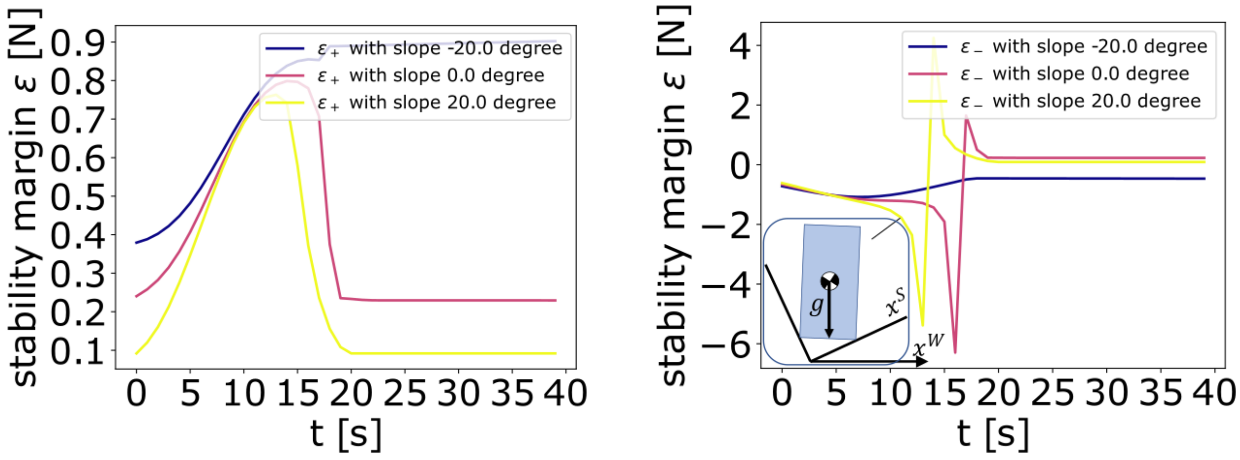
Figure 7.17: We show the time history of object angle, finger position, contact forces from a manipulator, and frictional stability margins. The top row shows the result with peg 2 and the bottom one shows the result with peg 3. The pink and blue shade regions represent that the system follows mode 1 and mode 2, respectively.

proposed optimization problem over the mode sequences (7.32) can find feasible solution. We observed that the benchmark two-stage optimization problem found feasible solutions only 2 times while the mode-based optimization using (7.32) was successfully able to find feasible 18 out of 20 times. Therefore, we verify that our proposed optimization problem enables to find solutions much more effortlessly. The benchmark method requires careful selection of parameters to ensure feasibility (as was explained in [131]).

7.5.8 Results of Bilevel Optimization for Uncertain Mass on a Slope

We present results of objects with uncertain mass with varying angles of slope discussed in Sec 7.3.6. We consider gear 2 with $\phi = [-20^\circ, 0^\circ, 20^\circ]$ as an angle of slope.

Fig. 7.18a and Fig. 7.18b shows the time history of the stability margin ϵ_+ and ϵ_-, respectively. Fig. 7.18a shows that the smaller ϕ is, the larger ϵ_+ is during the manipulation. ϵ_+ under mass uncertainty considers if contact B is losing as we discuss in (7.11). Fig. 7.18a means that contact B can more easily lose contact as *phi* increases. This makes sense because the larger the angle of slope ϕ is, the larger the moment which makes the object rotate along the counter-clockwise direction, resulting in the loss of contact at B. Similarly, ϵ_- under mass uncertainty considers if contact A is losing as we discuss in (7.9). Fig. 7.18b means



(a)

(b)

Figure 7.18: We consider CIBO with uncertain mass on varying angles of slope. (a): Time history of stability margin, ϵ_+. (b) Time history of stability margin, ϵ_-. The case where the object is on the slope whose angle of slope is 20° is illustrated in Fig. 7.18b.

that contact A can more easily lose contact as ϕ decreases at $t = \in [0, 15]$ s. This makes sense because the smaller the angle of slope ϕ is, the larger the moment which makes the object rotate along the clockwise direction, resulting in the loss of contact at A.

7.5.9 Results of Bilevel Optimization for Patch Contact

Table 7.9 shows the computed stability margin considering patch contact shows the greater margins for both positive and negative directions. Hence, we verify that our optimization can still work with patch contact and design the robust controller for maximizing the worst-case stability margin. Intuitively, this result makes sense since the contact area increases and the pivoting system has a larger physically feasible space, resulting in a greater stability margin.

Fig. 7.19 illustrates the time history of frictional stability margin of gear 2 from CIBO with considering point contact and with considering patch contact. Both CIBO with point contact and patch contact have the smallest (i.e., worst-case) stability margin at $t = 0$. However, CIBO with patch contact shows a greater margin at $t = 0$, as we discuss above. In addition, over the trajectory, CIBO with patch contact shows a greater margin than that

Table 7.6: Average Solving Time (AST) comparison between benchmark optimization (7.25) and CIBO under mass uncertainty using (7.29) with gear 2.

N	AST (s) of (7.25)	AST (s) of (7.29)
30	0.21	0.38
60	0.50	0.68
120	1.01	1.24

with point contact. Thus, we quantitatively verify that using patch contact is beneficial over the trajectory even though the optimization aims at maximizing the worst-case margin, not the stability margin over the trajectory. It is noted that we are not able to obtain better margins using patch contact due to the non-convexity of the underlying optimization problem.

7.5.10 Computation Results

Table 7.6 compares the computation time between benchmark optimization (7.25) and CIBO under mass uncertainty using (7.29) for gear 2. Overall, (7.29) is not so computationally demanding compared to (7.25). However, as you can see in Table 7.7 and Table 7.8, once the optimization problem has too many complementarity constraints because of the KKT condition, we clearly observe that the computational time increases.

Table 7.7 and Table 7.8 shows the computational results for CIBO considering frictional uncertainty (7.30) and bilevel optimization over mode sequences (7.32), respectively.

In general, the computational time for CIBO is larger than the benchmark optimization as CIBO has larger number of complementarity constraints. In the future, we will try to work on better warm-starting strategies so that we might be able to accelerate the optimization.

Table 7.7: NLP specification for CIBO under frictional uncertainty using (7.30) with gear 1.

N	# of Variables	# of Constraints	Average Solving Time (s)
30	2339	2280	1.9
60	4679	4560	10.6
120	9359	9130	30.9

Table 7.8: NLP specification for CIBO over mode sequences considering uncertain CoM location using (7.32) with peg 3.

N	# of Variables	# of Constraints	Average Solving Time (s)
30	1648	1590	3.68
60	3298	3180	61.6
120	6598	6360	73.0

7.5.11 Hardware Experiments

We implement our controller using a 6 DoF manipulator to demonstrate the efficacy of our proposed method. In particular, we perform a set of experiments to compare our method against a baseline method using gear 1. To evaluate robustness for objects with unknown mass, we solve the optimization with mass different from the true mass of the object and implement the obtained trajectory on the object. We implement trajectories obtained from the two different optimization techniques using 4 different mass values, $m = \{100, 110, 140, 170\}$ g. Then, we implement the obtained trajectory on the object with known mass. Note that the actual mass of gear 1 is 140 g. We test the trajectories over 10 trials for the two different methods.

We observe that our proposed bilevel optimization is able to achieve 100 % success rates for all 4 mass values while benchmark optimization cannot realize stable pivoting for all 4 mass values over 10 trials. Note that the benchmark trajectory optimization also generates trajectories with non-zero frictional stability margin but they failed to pivot the object. The reason would be that the system has a number of uncertainties such as incorrect coefficient of

Table 7.9: Computed worst-case stability margin considering uncertain CoM location over the control horizon obtained from optimization for gear 2.

	r^*_+, r^*_- [mm]
Ours with point contact	5.27, 1.31
Ours with patch contact	6.81, 8.82

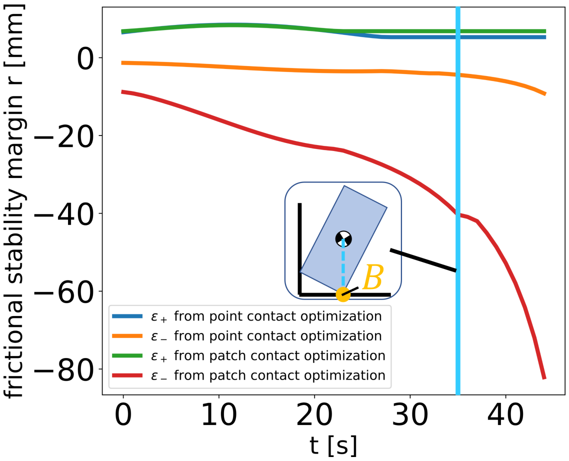


Figure 7.19: Trajectory of frictional stability margin of gear 2 based on uncertain CoM obtained from CIBO using point contact model and patch contact model, respectively. The vertical blue line represents the moment when the projection of CoM lies on the contact B.

friction, sensor noise in the F/T sensor (for implementing the force controller), etc. which are not considered in the model. We believe that these uncertainties make the objects unstable leading to the failure of pivoting. In contrast, even though CIBO also does not consider these uncertainties, it generates more robust trajectories and we believe that this additional robustness could account for the unknown uncertainty in the real hardware. We also observe that the trajectories generated by benchmark optimization can successfully realize pivoting if the manipulator uses patch contact during manipulation (thus getting more stability).

We perform hardware experiments with additional objects to evaluate the generalization of the proposed planning method. All the objects used in the hardware experiments are shown in Fig. 7.21. A video describing the hardware experiments with all the object can be found at this link `https://www.youtube.com/watch?v=ojlZDaGytSY`. Fig. 7.20 shows the

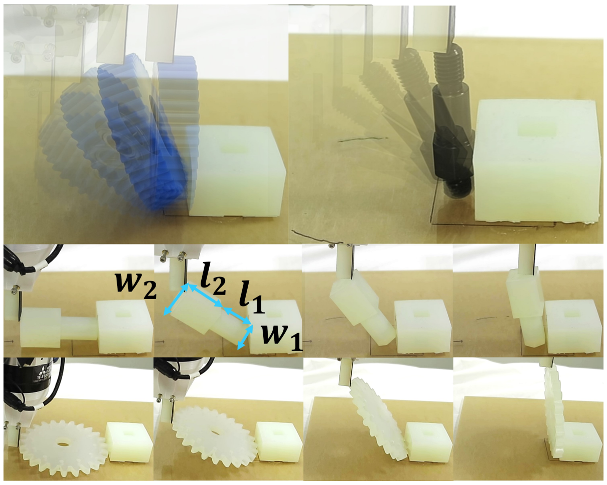


Figure 7.20: Snapshots of hardware experiments. We show snapshots of the white peg and gear (instead of overlaid images) for clarity.

snapshots of hardware experiments for the 4 objects detailed in Table 7.2. We observe that our bilevel optimization can successfully pivot all the objects during hardware experiments (see Fig. 7.20 and the videos). This shows that we can use the proposed method with objects with different size and shape.

7.6 Discussion and Future Work

Generalizable manipulation through contact requires that robots be able to incorporate and account for uncertainties during planning. However, designing the robust controller for achieving such manipulation remains an open problem and remains largely unexplored. This chapter presents *frictional stability*-aware optimization, a strategy that exploits friction for robust planning of pivoting manipulation. By considering a variety sources of uncertainty such as mass, CoM location, finger contact location, and friction coefficients, we discussed the stability margin for pivoting manipulation with slipping contact. We presented CIBO, which solves novel bilevel optimization for pivoting manipulation while optimizing the worst-case stability margin of pivoting manipulation for (non-convex) objects. The proposed method was evaluated in simulation using several test settings. We showed that our proposed bilevel optimization method is able to design trajectories which are robust to larger uncertainties

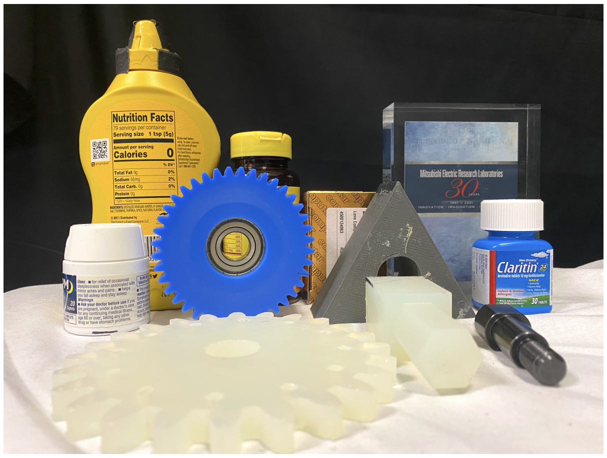


Figure 7.21: The different objects used in hardware evaluation of the proposed method. Please check the hardware experiments results in the video at this link https://www.youtube.com/watch?v=ojlZDaGytSY.

compared to a baseline trajectory optimization method. The proposed method was also demonstrated on a physical robotic system by implementing the computed trajectories in an open-loop fashion.

Although this chapter focuses on pivoting manipulation as a demonstration of our framework, our work can be generalized to other manipulation primitives such as pivoting with one-point contact, pushing, and grasping. This is because our stability margin analysis and CIBO are derived from quasi-static equilibrium (7.1) and the corresponding friction cone constraints (7.2). These conditions are very common across most manipulation problems, and thus our framework can be applicable to the aforementioned manipulation primitives as long as they satisfy (7.1) and (7.2).

The focus of this work is robust planning for pivoting in the presence of uncertainties. However, once the uncertainty of the system is *too* large (e.g., the mass of the object used in CIBO and the actual mass of the object is so different), the robot might not be able to complete the pivoting and the mission fails. Therefore, closed-loop control will be beneficial to recover from a failure. Thus, we present our closed-loop controllers in the next chapters.

CHAPTER 8

Covariance Steering for Uncertain Contact-Rich Systems

We only present our contributions in open-loop control so far. However, in reality, uncertainty always exists and thus (robust) open-loop control might not be enough for the robot to complete its desired task. In Chapter 8 and Chapter 9, we present our contributions in closed-loop control.

The most closely related work in this chapter is the work presented in Chapter 6. In Chapter 6, we present robust open-loop controller for contact-rich systems where we consider the change of contact modes explicitly in the proposed optimization problem. However, we make several assumptions, which are 1) the underlying distribution of the system follows Gaussian distribution, 2) contact force variables are regarded as deterministic variables, which should be regarded as random variables, and 3) we only consider open-loop control and do not consider the closed-loop control. These assumptions limit the application of the proposed algorithm in Chapter 6.

In this chapter, we relax these assumptions. We propose novel chance-constrained particle-filter-based optimization algorithm to approximate the distribution of SDLCS, which is not Gaussian. In particular, we propose NLP-based approach, not MIP-based approach, which dramatically decreases the computation time. Our controller is able to regulate both robots's states and contact states simultaneously. We demonstrate our controller for several contact-rich systems.

This chapter has been partially adapted from the following paper:

- **Y. Shirai**, D. Jha, and A. Raghunathan, "Covariance steering for uncertain contact-rich systems", in *Proc. 2023 IEEE Int. Conf. Robot. Auto.*, pp. 7923-7929, 2023.

8.1 Overview

Even though complementarity systems are well studied, stochastic complementarity systems are not well understood. The state and complementarity variables are implicitly related via the complementarity constraints – uncertainty in one leads to stochastic evolution of other. This makes uncertainty propagation challenging. Furthermore, multiplicity of solutions to the complementarity variables also makes it difficult to characterize the stochastic evolution. In this chapter, we present an approximate treatment of stochastic complementarity systems using particles. We present the design and evaluation of a contact-aware stochastic controller for covariance control of the underlying uncertain system. An important-particle algorithm is presented for an efficient solution to the resulting stochastic optimization problem.

Chance-constrained optimization (CCO) has been extensively studied in the control of uncertain systems [155, 36, 156, 157, 158, 26]. It allows us to plan using the uncertainty in the model by propagating the uncertainty which can be then used to design a controller for desired performance constraints of the system. However, in practice, the CCO techniques, based on the analytical form of chance constraints, impose restrictive assumptions of Gaussian uncertainty and linear constraints. Further, state uncertainty increases with time and thus finding a controller for satisfying tighter state constraints could be infeasible over a long planning horizon. This is often the case in control of nonlinear systems with large uncertainty. This problem is aggravated for contact-rich systems due to the presence of discontinuities in system dynamics.

To circumvent these challenges, we consider particle-based method for uncertainty propagation and explicit covariance control of our contact-rich system during optimization.

Contributions.

1. We present a novel formulation of covariance steering for complementarity systems using feedforward and feedback controller design.
2. An important-particle algorithm is proposed for numerical efficiency and we evaluate the proposed method on several examples.

8.2 Problem Formulation

In this section, we describe preliminaries of the method proposed in the current work.

8.2.1 Stochastic Discrete-time Linear Complementarity Systems

In this work, we consider the Stochastic Discrete-time Linear Complementarity Systems (SDLCS):

$$x_{k+1} = A_k(\xi)x_k + B_k u_k + C_k(\xi)\lambda_{k+1} + g_k(\xi) + w_k(\xi) \tag{8.1a}$$

$$0 \leq \lambda_{k+1} \perp D_k(\xi)x_k + E_k u_k + F_k(\xi)\lambda_{k+1} + h_k(\xi) + l_k(\xi) \geq 0 \tag{8.1b}$$

where k is the time-step index, $x_k \in \mathbb{R}^{n_x}$ is the state, $u_k \in \mathbb{R}^{n_u}$ is the control input, and $\lambda_k \in \mathbb{R}^{n_c}$ is the algebraic variable (e.g., contact forces). We define $x = [x_1, \ldots, x_T], u = [u_0, \ldots, u_{T-1}], \lambda = [\lambda_1, \ldots, \lambda_T]$. The parameter $\xi \sim \Xi$ is the uncertain parameter with distribution Ξ. In addition, $A_k(\xi) \in \mathbb{R}^{n_x \times n_x}$, $B_k \in \mathbb{R}^{n_x \times n_u}$, $C_k(\xi) \in \mathbb{R}^{n_x \times n_c}$, $g_k(\xi) \in \mathbb{R}^{n_x}$, $D_k(\xi) \in \mathbb{R}^{n_c \times n_x}$, $E_k \in \mathbb{R}^{n_c \times n_u}$, $F_k(\xi) \in \mathbb{R}^{n_c \times n_c}$, and $h_k(\xi) \in \mathbb{R}^{n_c}$ are all dependent on the uncertain parameter ξ. For simplicity, we abbreviate ξ from these matrices for the discussion in the following sections. The notation $0 \leq a \perp b \geq 0$ denotes the complementarity constraints $a \geq 0, b \geq 0, ab = 0$. The initial state of the system $x_0(\xi)$ is also assumed to be uncertain. $\|x\|_Q^2$ means a quadratic term with a weighting matrix Q.

In the following, we make the assumption that $F_k(\xi)$ is a P-matrix [139] for all k and ξ. Under this assumption, there is an unique solution λ_{k+1} to (8.1b) for each ξ and any u_k, x_k. From this it is easy to infer that there exists an unique trajectory x and λ for any realization of uncertainty $\xi \sim \Xi$ and controls u from every initial condition $x_0(\xi)$. In other words, we can define functions $\mathbf{x} : \Xi \times \mathbb{R}^{n_u(T-1)} \rightarrow \mathbb{R}^{n_x T}$ and $\boldsymbol{\lambda} : \Xi \times \mathbb{R}^{n_u T}$ that provides the unique trajectory given a realization of uncertainty, and the controls trajectory. Note that we do not show explicit dependence on initial condition due to the dependence of x_0 on the uncertain parameter ξ.

8.2.2 Stochastic Control for Contact-Rich Systems

In this work, we aim at finding a robust controller that satisfies chance constraints over SDLCS. To realize this, the following optimization problem can be formulated:

$$\min_{u} \sum_{k=1}^{T} \|\mathbb{E}_{\xi\sim\Xi}\left[\mathbf{x}_k(\xi, u)\right] - x_d\|_Q^2 + \sum_{k=0}^{T-1} \|u_k\|_R^2 \tag{8.2a}$$

$$\text{s.t. } u_k \in \mathcal{U} \tag{8.2b}$$

$$\Pr_{\xi\sim\Xi}\left(\mathbf{x}(\xi, u) \in \mathcal{X}\right) \geq \Delta \tag{8.2c}$$

where $Q = Q^\top$ is positive semidefinite, $R = R^\top$ is positive definite, $\mathcal{U}$ is a convex polytope consisting of a finite number of linear inequality constraints. x_d is the target state at $t = T$. The set $\mathcal{X}$ represents a convex safe region where the entire state trajectory has to lie in. We assume that $\mathcal{X} = \{x \in \mathbb{R}^{n_x T} \mid g_i(x) \leq 0 \,\forall\, i = 1, \ldots, n_g\}$. Pr denotes the probability of an event and Δ is the user-defined minimum safety probability, where the probability of satisfying constraints is at least greater than Δ.

We propose to obtain an approximate solution to (8.2) using the Sample Average Approximation (SAA) introduced in [159, 160]. We explain more details in Sec 8.3.

8.3 Covariance Steering for Contact-Rich Systems

This section presents our proposed framework of stochastic optimal control for contact-rich systems. Our framework approximates the distribution of the state and algebraic variables using particles. Under the assumption that $\bar{F}$ is P-matrix, our method can capture stochastic evolution of SDLCS such that we can formally guarantee the violation of states and design a closed-loop controller for SDLCS (i.e., covariance steering for SDLCS).

We first present our open- and closed-loop controller formulation for SDLCS using particles and then present a computationally beneficial approach based on the active-point method [161] to accelerate the resulting optimization.

8.3.1 Particle-based Control for Contact-Rich Systems

We propose to solve (8.2) approximately using SAA by sampling the uncertainty. In particular, we obtain N realizations of the uncertainty $\Xi^N = \{\xi^1, \ldots, \xi^N\}$ by sampling the distribution Ξ. In other words, we approximate the distribution Ξ using a finite-dimensional distribution Ξ^N which follows an uniform distribution on the samples. Accordingly, the SAA for (8.2) is given as

$$\min_{u} \sum_{k=1}^{T} \left\| \mathbb{E}_{\xi \sim \Xi^N} \left[\mathbf{x}_k(\xi, u) \right] - x_d \right\|_Q^2 + \sum_{k=0}^{T-1} \|u_k\|_R^2 \tag{8.3a}$$

$$\text{s.t. } u_k \in \mathcal{U} \tag{8.3b}$$

$$\mathrm{Pr}_{\xi \sim \Xi^N} \left(\mathbf{x}(\xi, u) \in \mathcal{X} \right) \geq \Delta. \tag{8.3c}$$

Note that the distribution Ξ has been replaced with the finite-dimensional Ξ^N in the above to simplify the computation of the expectation in the objective and chance constraint. However, there still remains the implicit function $\mathbf{x}(\xi, u)$ which requires us to simulate the SDLCS for every realization of $\xi \in \Xi^N$. We opt to remove this difficulty by replacing the implicit functions with the corresponding trajectories x^i, λ^i for each $\xi^i \in \Xi^N$.

Our proposed computational formulation using N particles is given by:

$$\min_{x^i, u, \lambda^i} \sum_{k=1}^{T} \left\| \frac{1}{N} \sum_{i=1}^{N} x_k^i - x_d \right\|_Q^2 + \sum_{k=0}^{T-1} \|u_k\|_R^2 \tag{8.4a}$$

$$\text{s.t. } x_{k+1}^i = A_k^i x_k^i + B_k u_k + C_k^i \lambda_{k+1}^i + g_k^i + w_k^i \tag{8.4b}$$

$$0 \leq \lambda_{k+1}^i \perp D_k^i x_k^i + E_k u_k + F_k^i \lambda_{k+1}^i + h_k^i + l_k^i \geq 0 \tag{8.4c}$$

$$x_0^i = x_0(\xi^i) \tag{8.4d}$$

$$u_k \in \mathcal{U} \tag{8.4e}$$

$$\frac{1}{N} \sum_{i=1}^{N} \mathbb{I} \left(x^i \in \mathcal{X} \right) \geq \Delta \tag{8.4f}$$

where $\mathbb{I}(\cdot)$ is an indicator function returning 1 when the conditions in the operand are satisfied and 0 otherwise. We denote θ_k^i as $\theta_k^i = [A_k^i, C_k^i, g_k^i, D_k^i, F_k^i, h_k^i, w_k^i, v_k^i]$. Note that x^i, λ^i

represent the state and algebraic variable trajectory, respectively, propagated from a particular set of particles x_0^i, θ_k^i. Using N trajectories obtained from N particles, we approximate mean of random variables as $\mathbb{E}_{\xi\sim\Xi}[\mathbf{x}_k(\xi,u)] \approx \frac{1}{N}\sum_{i=1}^{N} x_k^i, \mathbb{E}_{\xi\in\Xi}[\boldsymbol{\lambda}_k(\xi,u)] \approx \frac{1}{N}\sum_{i=1}^{N}\lambda_k^i$. In (8.4), we approximate (8.2a) using the mean variable as shown in (8.4a). Chance constraints (8.2c) can be also approximated as (8.4f) using N realization trajectories, which can be formulated as integer constraints (see [156]).

In this work, we consider the following controllers:

$$\textbf{feedforward} : u_k = v_k \tag{8.5a}$$

$$\textbf{feedback} : u_k = v_k + K_k(x_k - \bar{x}_k) + L_k(\lambda_k - \bar{\lambda}_k) \tag{8.5b}$$

where K_k, L_k are feedback gains to control covariance. For brevity, we use $\bar{x}_k = \frac{1}{N}\sum_{i=1}^{N} x_k^i$ and $\bar{\lambda}_k = \frac{1}{N}\sum_{i=1}^{N}\lambda_k^i$. We emphasize that controlling both states and contact variables is critical for contact-rich systems and thus we also introduce $L_k(\lambda_k - \bar{\lambda}_k)$ to (8.5b) to stabilize the system. Here, we focus on discussing feedback controller (8.5b) for (8.4). The optimization formulation for covariance steering of SDLCS using particles would be:

$$\min_{x^i,v,K,L,\lambda^i} \sum_{k=1}^{T} ||\bar{x}_k - x_d||_Q^2 + \sum_{k=0}^{T-1} \|u_k\|_R^2 \tag{8.6a}$$

$$\text{s. t. } x_{k+1}^i = (A_k^i + B_kK_k)x_k^i + B_kv_k + (C_k^i + B_kL_k)\lambda_{k+1}^i + \bar{g}_k^i - B_kK_k\bar{x}_k - B_kL_k\bar{\lambda}_{k+1} + w_k^i \tag{8.6b}$$

$$\begin{aligned} &0 \le \lambda_{k+1}^i \perp (D_k^i + E_kK_k)x_k^i + E_kv_k + (F_k^i + E_kL_k)\lambda_{k+1}^i \\ &+ h_k^i - E_kK_k\bar{x}_k - E_kL_k\bar{\lambda}_{k+1} + l_k^i \ge 0 \end{aligned} \tag{8.6c}$$

$$(8.4d), (8.4e), (8.4f) \tag{8.6d}$$

To solve (8.6), we need to take care of, (8.6b), (8.6c) and (8.4f). One method is mixed-integer programming. It is possible that binary variables can be used to deal with integer constraints (8.4f) using Big-M formulation. Also, bilinear terms in (8.6b) and (8.6c) can be approximated using McCormick envelopes, leading to additional binary variables. As a result, a number of binary variables are introduced and we observed that it is almost impossible to obtain a single feasible solution. Instead, in this work, we use NLP which can solve (8.6b) as nonlinear constraints and (8.6c) as complementarity constraints. We describe how we solve (8.4f) using NLP through complementarity constraints in Sec 8.3.2.

8.3.2 Bilevel Optimization for Particle-based Control

To solve (8.6) using NLP, we need to solve integer constraints (8.4f) in NLP fashion. To achieve this, we propose the following bilevel optimization problem.

$$\min_{x^i, v, K, L, \lambda^i, t^i, z^*} \sum_{k=1}^{T} \|\bar{x}_k - x_d\|_Q^2 + \sum_{k=0}^{T-1} \|u_k\|_R^2 \tag{8.7a}$$

$$\text{s. t. } (8.6b), (8.6c), (8.4e) \tag{8.7b}$$

$$\forall j = 1, \ldots, n_g,\ g_j(x) \le t^i, \tag{8.7c}$$

$$\frac{1}{N} \sum_{i=1}^{N} z^{i,*} \ge \Delta \tag{8.7d}$$

$$\forall i = 1, \ldots, N,\ z^{i,*} = \operatorname*{argmin}_{z^i} t^i z^i | 0 \le z^i \le 1 \tag{8.7e}$$

We introduce time-invariant parameter $t^i \in \mathbb{R}^1$ for each set of trajectory realization i. If $x^i \in \mathcal{X}$, $t^i \ge -\epsilon$ with $\epsilon \ge 0$. In contrast, if $x \notin \mathcal{X}$, $t^i \ge 0$. This condition is encoded in (8.7c). We have in total N lower-level optimization problems (8.7e), where each optimization is formulated as linear programming. $z^i \in \mathbb{R}^1$ is the decision variable used in i -th lower-level optimization problem.

The purpose of (8.7e) is to count the number of trajectory realizations that are inside $\mathcal{X}$. The optimal solution of (8.7e) can be as follows:

$$z^i = \begin{cases} 1, & t^i < 0 \\ [0, 1], & t^i = 0 \\ 0, & t^i > 0 \end{cases} \tag{8.8}$$

If $t^i < 0$, (8.7c) argues that $x^i \in \mathcal{X}$ and thus we count this i-th trajectory propagated from i-th particles as one. If $t^i = 0$, (8.7c) argues $x^i \in \mathcal{X}$ (x^i lies on the boundary of $\mathcal{X}$) and thus we count this i-th trajectory propagated from i-th particles as one. If $t^i > 0$, then x^i is not within $\mathcal{X}$, and thus we count it as zero. Then (8.7d) considers the approximated chance constraints.

Since the upper-level optimization decision variable t^i can be influenced by other upper-level decision variables, we need to solve these two problems simultaneously, leading to a bilevel optimization problem. Since the lower-level optimization problems are formulated as N linear programming problems, we can efficiently solve the entire bilevel optimization problem using the Karush-Kuhn-Tucker (KKT) condition as follows:

$$\min_{x^i,v,K,L,\lambda^i,t^i,z^{i,*},w^i_+,w^i_-} (8.7a) \tag{8.9a}$$

$$\text{s. t. } (8.7b), (8.7c), (8.7d) \tag{8.9b}$$

$$\forall i = 1, \ldots, N,\ 0 \leq z^{i,*} \leq 1, w^i_+, w^i_- \geq 0 \tag{8.9c}$$

$$w^i_+(z^{i,*} - 1) = 0, w^i_-(z^{i,*}) = 0, \tag{8.9d}$$

$$t^i + w^i_+ - w^i_- = 0 \tag{8.9e}$$

where w^i_+, w^i_- are Lagrange multipliers associated with $z^i - 1 \leq 0$, $-z^i \leq 0$, respectively. In conclusion, we obtain a single-level nonlinear programming problem with complementarity constraints, which can be efficiently solved using an off-the-shelf solver such as IPOPT [10].

8.3.3 Important-particle Method for Particle-based Control

One limitation of our method in Sec 8.3.2 is that the computation can be demanding with many particles to capture the evolution of uncertainty. In this section, we present an approximate algorithm which samples important particles which might be most informative for constraint violation. To decrease the computational burden, we employ an important-particle method (see Algorithm 5) which starts from a relatively small number of particles and keeps adding particles if the chance constraints are not satisfied due to the lack of the accurate approximation of variables. Since we start from a small number of particles, it is possible that our optimization could quickly find a feasible solution which works over testing data set. However, in the case when the problem is infeasible for some particles, we add the particles which experience maximum constraint violation to our set. Thus, we call our

Algorithm 5 ImportantParticle$(\text{Param}, \alpha, \beta, \gamma, \eta)$

```
$j = 0$, $\theta = \gamma$, $\Delta_\alpha = 0$
while $j \leq$ MAX-ITER and $(\Delta - \Delta_\alpha)^2 \geq \Delta_{\text{th}}$ and $\Delta > \Delta_\alpha$; do
    Run (8.9) with $N = \theta$
    if The obtained solution from (8.9) is feasible then
        Run MC simulation with $\alpha$ particles and calculate $\Delta_\alpha$.
        Choose the $\eta$ worst particles that violate chance constraints.
    else
        Choose the random $\eta$ particles.
    $\theta = \theta + \eta$
Run MC simulation with $\beta$ particles and calculate $\Delta_\beta$.
return $x^{i,*}, v^*, K^*, L^*, \lambda^{i,*}, t^{i,*}, z^{i,*}, w^i_+, w^i_-, \Delta_\beta$
```

proposed method "important-particle" method– the worst particles specify the boundary of feasible sets.

The pseudocode of our important-particle method for covariance steering is shown in Algorithm 5. Param is the collection of parameters such as Q, R. α, β represent the number of particles for training and testing the controller, respectively. γ is the number of initial particles our method uses during its first iteration. η is the number of particles our methods adds to (8.9) for each iteration.

As shown in Algorithm 5, our method keeps adding more particles unless either it runs more than MAX-ITER or converges to user-defined Δ given threshold Δ_{th}. For each iteration, we run (8.9). If the obtained solution is feasible, we do Monte Carlo simulation (MC simulation) over the training data set with α particles and calculate the empirical safe probability Δ_α. If this Δ_α is close to or greater than Δ, we terminate the while loop and run the obtained controller over the testing data set with β particles. Otherwise, we choose the η worst particles based on how much they violate the chance constraints and add them to θ. If we obtain the infeasible solution or the "restoration phase failed" solution in IPOPT, we randomly choose the η particles.

8.4 Results

In this section, we present numerical results for our proposed approach and compare them against some baselines. In particular, we would like to highlight and understand the following questions:

1. Does uncertainty in complementarity constraints lead to uncertainty in state trajectory?
2. How does the proposed controller perform of variance of states for SDLCS?

We implement our method using IPOPT [10] with PYROBOCOP [16]. The optimization problem is implemented on a computer with Intel i7-12700K processor. We set $\alpha = 250, \beta = 1000$ for Algorithm 5. For γ and η in Algorithm 5, we use the different values for different applications as shown in Table 8.2 and Table 8.3. When we run (8.9) alone without using Algorithm 5, we use 1000 samples to calculate the empirical probability of failure to evaluate the satisfaction of chance constraints.

Here we explain how we simulate trajectories (i.e., perform MC simulation for SDLCS, see [162] for more details). We propagate the dynamics by finding the roots of the complementarity system with sampled parameters given the control sequence obtained from optimization. We run each case for 1000 trials with different sampled parameters to estimate the probability of failure. Note that, unlike the continuous-domain dynamics, we cannot rollout the dynamics for SDLCS with the given control sequences since we do not have the access to λ_{k+1}.

8.4.1 Uncertainty Propagation for SDLCS

We show uncertainty evolution for SDLCS. We demonstrate this for a cartpole system with softwalls (see [19] for more details). Here we consider both k_1 and k_2 follows uniform distributions where upper bound of uniform distribution for k_1 and k_2 is 14, 12, respectively, and the lower bound is 5 for both k_1 and k_2. In this experiment, we do not run any controller:

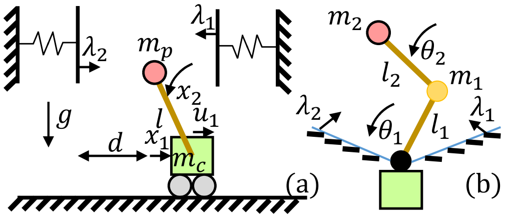


Figure 8.1: (a): cartpole with softwalls. (b): acrobot with soft joints.

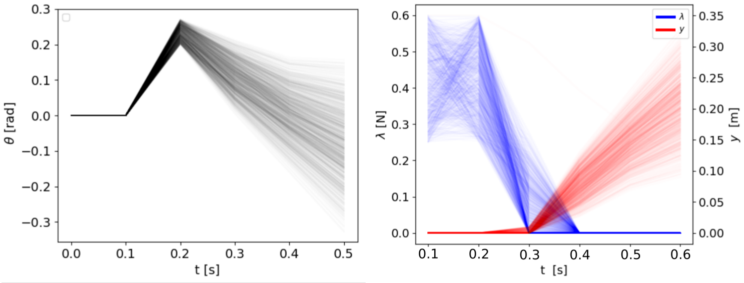


Figure 8.2: Uncertainty propagation for cartpole system. Here only uncertainty arises from stiffness parameters k_1, k_2.

we simply propagate SDLCS given uncertain parameters in order to show how the SDLCS behaves.

Fig. 8.2 shows the evolution of uncertainty for the aforementioned system. At $t = 0$ s, there is no uncertainty for state $\theta_{t=0}$. However, because we provide uncertainty with k_1 and k_2, $\lambda_{t=0.1}$ has uncertainty. This is again because given realization of uncertain parameters, complementarity constraints give a realization of λ and y, resulting in uncertainty in λ and y. This stochastic $\lambda_{t=0.1}$ brings uncertainty in $\theta_{t=0.1}$ based on (8.1). As shown in Fig. 8.2, both state and complementarity variables are stochastic. This can not be captured in approximations like Expected Residual Minimization (ERM) [2].

8.4.2 Cartpole with Softwalls

We demonstrate our open- and closed-loop controllers for cartpole with softwalls system. x is the cart position and θ is the pole angle. u_1 is the control and λ_1, λ_2 are the reaction forces at from the wall 1, 2, respectively. We have the following deterministic physical parameters. $g = 9.81$ is the gravitational acceleration, $m_p = 0.1, m_c = 1.0$ are the mass of the pole, cart,

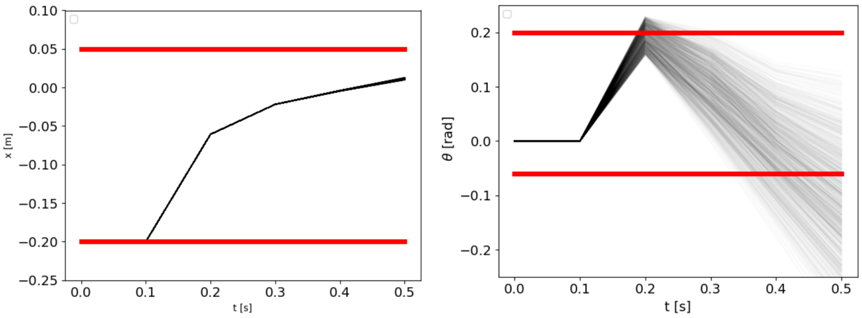


Figure 8.3: Simulated trajectories for cartpole system using ERM-based controller. $\Delta = 0.2$ and $\Delta_{\text{test}} = 0.083$. Red lines show boundaries specified in chance constraints.

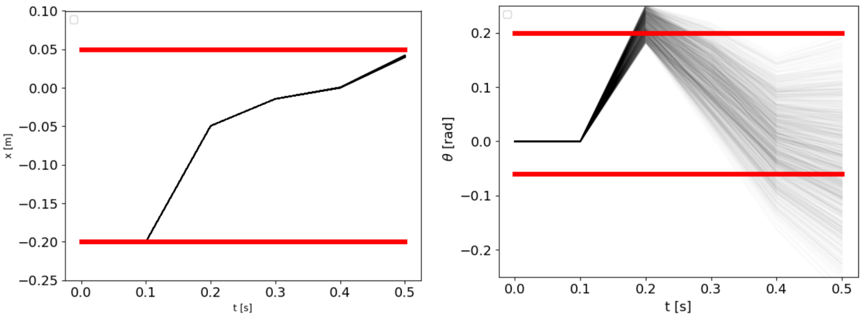


Figure 8.4: Simulated trajectories for cartpole system using our open-loop controller. $\Delta = 0.2$ and $\Delta_{\text{test}} = 0.190$ where Δ is input of optimization and Δ_{test} is the empirically obtained success rate from MC simulation. Red lines show boundaries specified in chance constraints.

respectively. $l = 0.5$ is the length of the pole and $d = 0.15$ is the distance from the origin of the coordinate to the walls. We assume that the uncertainty arises from the k_1, k_2 and use the same distribution in Sec 8.4.1. We set $dt = 0.1$ for the explicit Euler integration and $T = 6$.

The results using ERM and our controller for the open-loop trajectory are shown in Fig. 8.3, Fig. 8.4. We observed that the proposed open-loop controller shows the better satisfaction of chance constraints compared to the ERM-based method. This is because our method explicitly considers propagation of uncertainty for SDLCS while the ERM-based method is unable to consider. Also, we observe that the gap between the commanded Δ used in our optimization and Δ_{test} obtained from MC simulation over testing dataset is smaller the gap between the commanded Δ used in ERM method and Δ_{test} obtained from

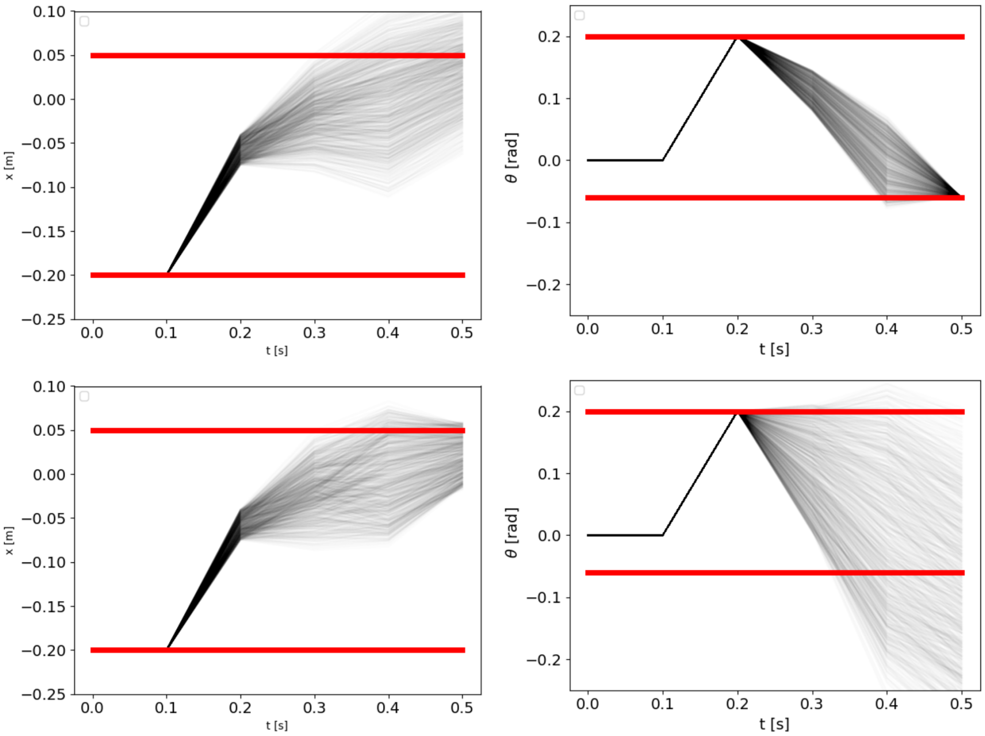


Figure 8.5: Simulated trajectories for cartpole system using our closed-loop controller. Top: $\Delta = 0.6$ and $\Delta_{\text{test}} = 0.510$, bottom: $\Delta = 0.2$ and $\Delta_{\text{test}} = 0.188$, where Δ is input of optimization and Δ_{test} is the empirically obtained success rate from MC simulation. Red lines show boundaries specified in chance constraints.

MC simulation over testing dataset. Again this is because our method could capture the evolution of uncertainty for SDLCS. However, even our open-loop controller does not show the much better performance than the ERM. To show the higher Δ_{test}, we need to input the higher Δ as an input of optimization. It is quite difficult especially for long-horizon planning problems since uncertainty keeps evolving, which can be observed from both Fig. 8.3 and Fig. 8.4.

Next, we discuss the difference among our proposed contact-aware closed-loop, the non-contact-aware (i.e., $L_k = 0, \forall k$ in (8.5b)) closed loop, and the open-loop controllers. We observed that in Table 8.1, (8.9) for open-loop controller with high Δ was unable to find feasible solutions but (8.9) for closed-loop controller could find feasible solutions. Since the closed-loop controller can change feedback gains to satisfy chance constraints, it could find feasible solutions with high Δ. Also, Table 8.1 shows that the contact-aware closed-loop

Table 8.1: Comparison of feasibility for cartpole system among open-, non-contact-aware closed, and contact-aware-closed controllers with different Δ. $\circ$ and $\times$ show if optimization finds a feasible solution or not, respectively.

Δ	0.8	0.7	0.6	0.4	0.2
Open-loop	$\times$	$\times$	$\times$	$\circ$	$\circ$
Non-contact-aware closed-loop	$\times$	$\times$	$\circ$	$\circ$	$\circ$
Contact-aware closed-loop	$\circ$	$\circ$	$\circ$	$\circ$	$\circ$

controller could find the feasible solution with high $\Delta = 0.8, 0.7$ but the non-contact-aware controller (i.e., $L_k = 0, \forall k$ in (8.5b)) could not. For SDLCS, introducing feedback to both states and forces is important to realize the robust motion. The MC simulation results using our contact-aware closed-loop controller are shown in Fig. 8.5. In contrast to Fig. 8.3 and Fig. 8.4, the closed-loop controller could bound the distribution of the states because it controls covariance.

We discuss computational results. Firstly, we observe that our important-particle method converges and the gap between Δ_{train} and Δ_{test} is small once it finishes its third time iteration. It means that our important-particle method could successfully find feasible trajectories with relative small number of particles. Secondly, in Table 8.2 the important-particle method shows the higher Δ_{train} as the number of particles used in optimization increases. The proposed important-particle method shows better convergence (in total 208 s to have $\Delta_{\text{train}} \geq 0.49$) than the naive method (620 s with 50 particles to have $\Delta_{\text{test}} \geq 0.49$) since our important-particle method keeps choosing the worst-case particles which break chance constraints.

8.4.3 Acrobot with Soft Joints

We also demonstrate our controller for acrobot with soft joints system (see [19] for more details). θ_1 is the first joint angle and θ_2 is the second joint angle. u_1 is the control at the second joint and λ_1, λ_2 are the reaction forces at from the wall 1, 2, respectively. We have

Table 8.2: Comparison of safe probability and runtime for cartpole system between important-particle method (top) with $\gamma = 10, \eta = 10$ and naive method (bottom) with $\Delta = 0.6$ for designing the closed-loop controller. T represents runtime for each iteration and n_p is the number of particles.

iter	1	2	3
Δ_{train}	0.2708	0.09	0.592
Δ_{test}	N/A	N/A	0.588
T [s]	25	35	148
n_p	10	20	30

Case	1	2	3	4
Δ_{test}	0.277	0.376	0.451	0.499
T [s]	25	26	55	620
n_p	10	20	30	50

the following deterministic physical parameters.$g = 9.81$ is the gravitational acceleration, $m_1 = 0.5, m_2 = 1.0$ are the mass of the pole, cart, respectively. $l_1 = 0.5$ is the length of the rod from the first to the second joint. $d = 0.2$ is the angle limit of θ_1. We consider the stochastic physical parameters k and l_2 where k is the stiffness of the walls and l_2 is the length of the second rod. We assume that k follows uniform distribution where the upper bound and the lower bound of the distribution is 1.6 and 0.6, respectively. We assume that l_2 follows a truncated Gaussian distribution where we set the mean to 1.0, variance to 0.01, the upper bound of the interval is 1.3, and the lower bound of the interval is 0.7, respectively. We set $dt = 0.04$ for the explicit Euler integration and $T = 15$.

The open- and closed-loop trajectories are shown in Fig. 8.6. We observed that both controller could satisfy chance constraints over the testing dataset and the closed-loop controller shows the better performance. Table 8.3 shows that the important-particle method shows the higher $\Delta_{\text{test}} = 0.771$ than the naive method with the same number of particles used in optimization.

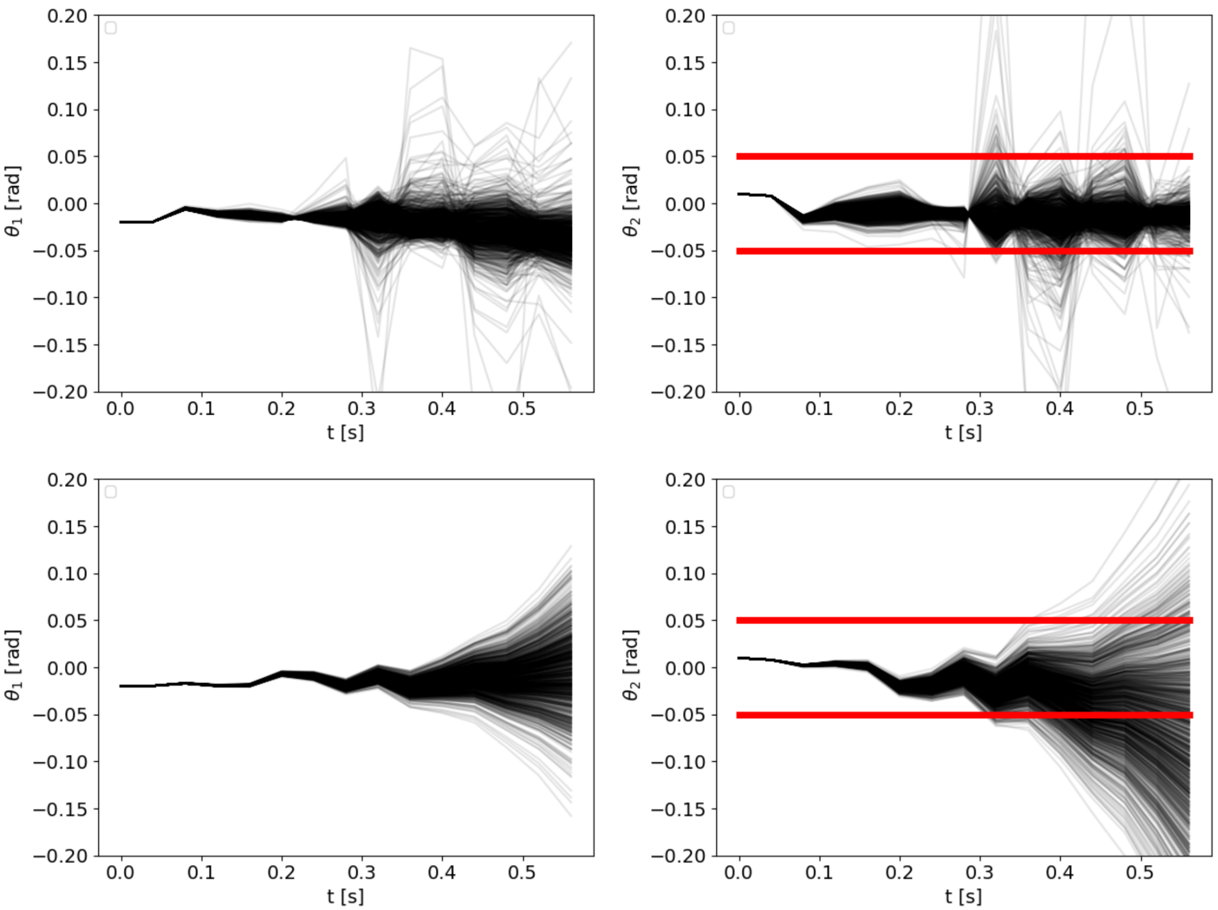


Figure 8.6: Simulated trajectories for acrobot using our open- and closed-loop controllers. Top: closed-loop controller with $\Delta = 0.8$ and $\Delta_{\text{test}} = 0.771$, bottom: open-loop controller with $\Delta = 0.4$ and $\Delta_{\text{test}} = 0.366$. Red lines show boundaries specified in chance constraints. The reader should note that open-loop controller solution was infeasible for $\Delta = 0.8$, and thus we show results for $\Delta = 0.4$.

8.5 Discussion

Stochastic complementarity systems are not well understood in literature. This chapter presents a study of SDLCS to perform covariance steering using particles. Under the assumption of uniqueness of trajectory ($\bar{F}$ is P-matrix) for complementarity systems, the proposed method is able to compute covariance controller for SDLCS. We presented an important-particle method to alleviate computational complexity of the resulting optimization problem. It is shown that our work could design open- and closed-loop controllers with chance constraints by appropriately considering the evolution of uncertainty for SDLCS.

Table 8.3: Comparison of safe probability and runtime for acrobot system between important-particle method (top) with $\gamma = 4, \eta = 4$ and naive method (bottom) with $\Delta = 0.8$. T represents runtime for each iteration and n_p is the number of particles.

iter	1	2	3	4	5	6	7
Δ_{train}	0.426	0.485	0.562	0.625	0.363	0.593	0.763
Δ_{test}	N/A	N/A	N/A	N/A	N/A	N/A	0.771
T [s]	31	97	557	887	698	2450	779
n_p	4	8	12	16	20	24	28

Case	1	2	3	4	5	6	7
Δ_{test}	0.009	0.103	0.159	0.541	0.670	0.553	0.539
T [s]	31	15	229	260	944	3993	901
n_p	4	8	12	16	20	24	28

In the future, we would like to study more general manipulation systems by relaxing the assumption on uniqueness of trajectory for SDLCS. Another limitation of this work is that the computation is still demanding. Thus, we would like to employ distributed optimization techniques such as ADMM [29, 163].

CHAPTER 9

Closed-Loop Tactile Control for Tool Manipulation

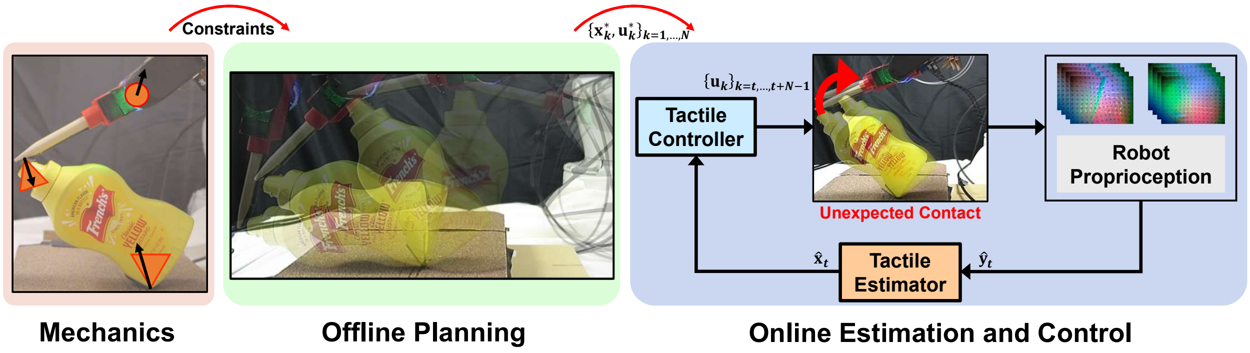
In this chapter, we present closed-loop control of a complex manipulation task where a robot uses a tool to interact with objects. Manipulation using a tool leads to complex kinematics and contact constraints that need to be satisfied to generate feasible manipulation trajectories. We first present an open-loop controller design using Non-Linear Programming (NLP) that satisfies these constraints. In order to design a closed-loop controller, we present a pose estimator of objects and tools using tactile sensors. Using our tactile estimator, we design a closed-loop controller based on Model Predictive Control (MPC). The proposed algorithm is verified using a 6 DoF manipulator on tasks using a variety of objects and tools. We verify that our closed-loop controller can successfully perform tool manipulation under several unexpected contacts. All hardware experiment videos can be found at https://youtu.be/VsClK04qDhk

This chapter has been partially adapted from the following paper:

- **Y. Shirai**, D. Jha, A. Raghunathan, and D. Hong, "Tactile Tool Manipulation", in *Proc. 2023 IEEE Int. Conf. Robot. Auto.*, pp. 12597-12603, 2023.

9.1 Overview

Using contacts efficiently can provide additional dexterity to robots while performing complex manipulation tasks [4, 164, 163, 165]. However, most robotic systems avoid making contact with their environment. This is mainly because contact interactions lead to complex, discontinuous dynamics and thus, planning, estimation, and control of manipulation require careful



Mechanics **Offline Planning** **Online Estimation and Control**

Figure 9.1: We present tactile tool manipulation where a robot uses an external tool to manipulate an external object. Usage of an external tool results in multiple contact formations which leads to a large number of constraints that need to be satisfied during manipulation. Using the underlying frictional mechanics, we present design of open-loop and closed-loop controllers which can successfully maintain all contact formations during manipulation. We present the design and use of a tactile estimator which makes use of tactile sensing to estimate pose of the system. The tactile estimator is used to perform closed-loop control in an MPC fashion. All hardware experiment videos could be found at https://youtu.be/VsClK04qDhk.

treatment of these constraints. As a result of these challenges, most of the classical control approaches are not applicable to control of manipulation systems [4, 166, 162, 131, 44]. However, closed-loop control of manipulation tasks is imperative for design of robust, high-performance robotic systems that can effortlessly interact with their environments.

In this chapter, we consider tool manipulation where a robot can grasp an external tool that can be used to pivot an external object in the environment (See Fig. 9.1). As could be seen in Fig. 9.2, tool manipulation leads to multiple contact formations between the robot & a tool, the tool & an object, and the object & environment. It is easy to imagine that planning for tool manipulation needs to incorporate all constraints imposed by these contact formations. This makes planning for tool manipulation extremely challenging. Furthermore, the robot can not directly observe all the relevant contact and object states during tool manipulation. This imposes additional complexity during controller design. This makes tool manipulation a challenging, albeit extremely rich system to study closed-loop controller design for manipulation.

We present design of planning, estimation, and control for tool manipulation using tactile sensing. In particular, we first present analysis of the underlying contact mechanics which allows us to plan feasible trajectories for manipulating an external object. To allow robust implementation of the planned manipulation, we design a closed-loop controller using tactile sensors co-located at the fingers of the gripper. We present design of a tactile estimator which estimates the pose of the external object during manipulation. This estimator is used to design a closed-loop controller using MPC. The proposed planner and closed-loop controller are extensively tested with several different tool-object pairs.

Contributions. This chapter has the following contributions:

1. We present design of closed-loop controller for tool manipulation using tactile sensing and NLP.
2. The proposed controller is implemented and verified on tasks using different tools and objects using a 6 DoF manipulator equipped with GelSlim tactile sensors.

9.2 Mechanics of Tool Manipulation

In this section, we explain mechanics of tool manipulation as illustrated in Fig. 9.2 and then discuss Trajectory Optimization (TO) of tool manipulation for designing open-loop trajectories. Before explaining the details, we present our assumptions in this chapter as follows:

1. The object and the tool are rigid.
2. The object and the tool always stay in quasi-static equilibrium.
3. We consider simplified quasi-static mechanics in 2D.
4. The kinematics of the tool and the friction coefficients for different contact formations are known.

The notation of variables are summarized in Table 9.1. We define the rotation matrix from

Table 9.1: **Notation of variables**. Σ column indicates the frame of variables. See Fig. 9.2 and Fig. 9.3 for graphical definition.

Name	Description	Size	Σ
$\mathbf{w}_E$	reaction wrench at point A	$\mathbb{R}^2$	W
$\mathbf{w}_O$	gravity of object at point O	$\mathbb{R}^2$	W
$\mathbf{w}_{TO}$	wrench from the tool to the object at point B	$\mathbb{R}^2$	T
$\mathbf{w}_T$	gravity of tool at point T	$\mathbb{R}^2$	W
$\mathbf{w}_G$	wrench from the gripper to the tool at patch C	$\mathbb{R}^2$	G
θ_O	orientation of object	$\mathbb{R}^1$	W
θ_T	orientation of tool	$\mathbb{R}^1$	W
θ_G	orientation of gripper	$\mathbb{R}^1$	W
θ_S	relative orientation of frame at center of grasp	$\mathbb{R}^1$	S

frame Σ_A to Σ_B as ${}^A_BR$. We denote $\mathbf{p}_i$ as a position at contact i defined in Σ_W. We denote x- and y-axis as axes in 2D plane and z-axis is perpendicular to the plane.

9.2.1 Quasi-Static Mechanics of Tool Manipulation

As is shown in Fig. 9.2, tool manipulation leads to several contact formations at A, B, and C that would need to be maintained during manipulation. Additionally, we need to consider quasi-static equilibrium of the tool and the object in the presence of these contacts. The static equilibrium of the object is described as:

$$F_O(\mathbf{w}_E, \mathbf{w}_O, {}^W_TR\mathbf{w}_{TO}) = \mathbf{0}, \tag{9.1a}$$

$$G_O(\mathbf{w}_E, \mathbf{w}_O, {}^W_TR\mathbf{w}_{TO}, \mathbf{p}_A, \mathbf{p}_B, \mathbf{p}_O) = 0 \tag{9.1b}$$

where F_O and G_O represent static equilibrium of force and moment, respectively. The static equilibrium of the tool is:

$$F_T(\mathbf{w}_T, {}^W_GR\mathbf{w}_G, {}^W_TR\mathbf{w}_{OT}) = \mathbf{0}, \tag{9.2a}$$

$$G_T(\mathbf{w}_T, {}^W_GR\mathbf{w}_G, {}^W_TR\mathbf{w}_{OT}, \mathbf{p}_B, \mathbf{p}_T, \mathbf{p}_{G1}, \mathbf{p}_{G2}) = 0 \tag{9.2b}$$

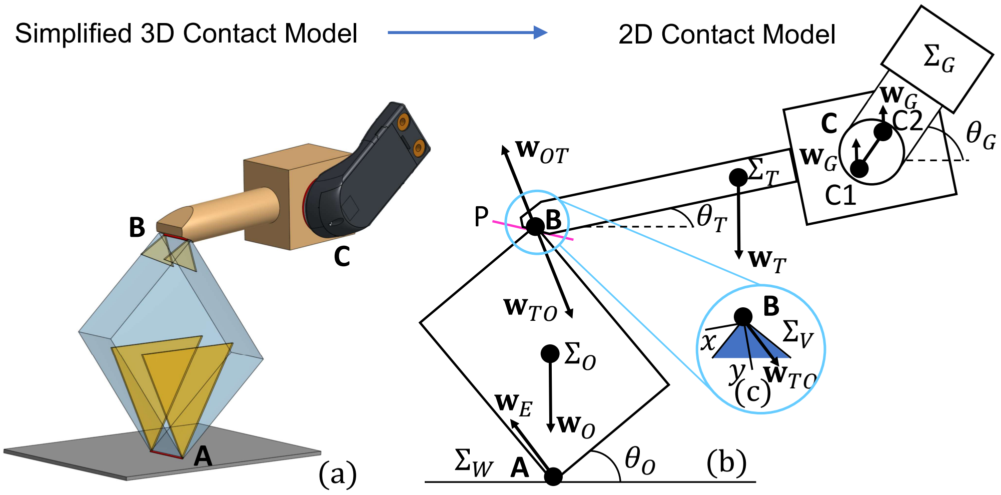

Simplified 3D Contact Model
2D Contact Model
B
C
A
(a)
$\mathbf{w}_{OT}$
P
B
θ_T
Σ_T
$\mathbf{w}_T$
Σ_G
$\mathbf{w}_G$
C2
θ_G
C
$\mathbf{w}_G$
C1
$\mathbf{w}_{TO}$
Σ_O
$\mathbf{w}_E$
$\mathbf{w}_O$
Σ_W
A
θ_O
(b)
B
Σ_V
x
y
$\mathbf{w}_{TO}$
(c)

Figure 9.2: **Mechanics of tool manipulation.** (a): A simplified 3D contact model for tool manipulation highlighting the three main contact interactions during the task. (b): Free-body diagram of a rigid body and a tool during tool manipulation in 2D. (c): Force from a tool to an object has to lie on a cone defined by the shape of the object.

Note that ${}^{T}\mathbf{w}_{TO} = -{}^{T}\mathbf{w}_{OT}$. In this work, we approximate patch contact at C as two point contacts with the same force distribution, and thus we have $\mathbf{p}_{C1}, \mathbf{p}_{C2}$ in (9.2b). In the next section, we consider contact formations at A, B, and C while making necessary simplifications for modeling.

9.2.2 Contact Model

We first discuss the contact model in 3D then we present the approximated contact model in 2D using Fig. 9.2. In a simplified 3D setting, the different contact formations could be best described as follows:

1. contact A: line contact.
2. contact B: line or patch contact.
3. contact C: patch contact.

For line contacts A and B, we need to consider generalized friction cones [167] to describe sticking line contact in 3D. However, this work considers manipulation in 2D as shown in

Fig. 9.2 (b) and thus we can argue that there is no moment to break the line contact. Thus, we can approximate line contacts as two point contacts with the same force distribution, leading to the larger coefficients of friction effectively. Also for patch contacts at contact C, we need to consider 4D limit surface [168] where we have 3D force $[f_x, f_y, f_z]$ and 1D moment m_z. However, in practice, implementing m_z is difficult, especially for position-controlled manipulators with a force controller with low bandwidth. Thus, this work approximates patch contact at C as two point contacts (see Fig. 9.2 (b)) with same force distribution. This approximation makes low-level controllers track the force trajectory easily.

For point contacts A, B, C_1, C_2, we have the following friction cone constraints:

$$-\mu_i f_y^i \leq f_x^i \leq \mu_i f_y^i, f_y^i \geq 0, \forall i = \{A, B, C_1, C_2\} \tag{9.3}$$

where μ_i is the coefficient of friction at contact $i = \{A, B, C_1, C_2\}$ and f_x^i, f_y^i are tangential and normal forces for each local coordinate. Note that we set $\mu_i = 2\mu_{i,\text{point}}, i = \{A, B\}$ where $\mu_{i,\text{point}}$ is the coefficient of friction between the environment and *point* contact A, B to take into account line contact effects.

Remark 1: Contact formation at B could be either patch or line contact. To formally discuss the change of these two different contact modes, constraints such as complementarity constraints are required, which is out of scope in this chapter. Thus, we assume that contact B always realizes line contact.

9.2.3 Contact between Tool and Object

The line contact at B introduces an important insight. As illustrated in Fig. 9.2 (b), this line contact is on a certain plane P created by a tool. The plane P is used to discuss the friction cone between the object and the tool since slipping can only occur along the plane P. Thus, by changing the orientation of the tool, the orientation of this plane also changes. This does not have an effect on local friction cone constraints (9.3) but does have an effect on the object through static equilibrium. Furthermore, different tools have different tip shapes (see Fig. 9.4). Based on kinematics of the tool, local force definition changes, which is tricky and unique to tool manipulation. In conclusion, the system has a preferred orientation of the plane P for finding a feasible trajectory.

Another unique nature of this task is that we need to explicitly consider the feasible region of a force controller. Note that the manipulator can only apply forces along the axes where its motion is constrained. This constraint needs to be explicitly enforced during optimization to generate mechanically feasible force trajectories.

Hence, like friction cone constraints, we formulate inequality constraints in vertex frame Σ_V (see Fig. 9.2 (c)) such that $\mathbf{w}_{TO}$ is constrained by the object:

$$-\rho f_y \leq f_x \leq \rho f_y, f_y \geq 0 \tag{9.4}$$

where $[f_x, f_x]^\top = {}^V_T R \mathbf{w}_{TO}$. We define Σ_V such that y-axis of Σ_V bisect the angle of vertex B. ρ can be determined by the shape of the object.

9.2.4 Trajectory Optimization for Planning

We formulate TO for tool manipulation as follows:

$$\min_{\mathbf{x},\mathbf{u},\mathbf{f}} \sum_{k=1}^{N} (\mathbf{x}_k - \mathbf{x}_g)^\top Q (\mathbf{x}_k - \mathbf{x}_g) + \sum_{k=0}^{N-1} \mathbf{u}_k^\top R \mathbf{u}_k \tag{9.5a}$$

$$\text{s.t. } (9.1), (9.2), (9.3), (9.4), \tag{9.5b}$$

$$\mathbf{x}_0 = \mathbf{x}_s, \mathbf{x}_N = \mathbf{x}_g, \mathbf{x}_k \in \mathcal{X}, \mathbf{u}_k \in \mathcal{U}, \mathbf{f}_k \in \mathcal{F} \tag{9.5c}$$

where $\mathbf{x}_k = [\theta_{O,k}, \theta_{T,k}, \theta_{G,k}]^\top$, $\mathbf{u}_k = \mathbf{w}_{G,k}$, $\mathbf{f}_k = [\mathbf{w}_{E,k}^\top, \mathbf{w}_{TO,k}^\top]^\top$, $Q = Q^\top \geq 0, R = R^\top > 0$. $\mathcal{X}$, $\mathcal{U}$, and $\mathcal{F}$ are convex polytopes, consisting of a finite number of linear inequality constraints. $\mathbf{p}_i$ can be calculated from kinematics with $\mathbf{x}_k$ since we could assume that contacts ensure sticking contacts by satisfying (9.3). Based on the solution of (9.5), we can calculate the pose and force trajectory of the end-effector and we command them during implementation. The resulting optimization in (9.5) is NLP, which can be solved using off-the-shelf solvers such as IPOPT [10].

Remark 2: For nox-convex shape objects (e.g., peg in Fig. 9.4), the origin of pivoting, $\mathbf{p}_A$, changes over the trajectory. Thus, we cannot directly apply (9.5) for the non-convex objects. Hence, we solve (9.5) hierarchically for them where we solve (9.5) with the first contact origin and then we solve (9.5) with the next contact origin and so far and so forth.

9.3 Tactile Tool Manipulation

In this section, we present design of our closed-loop controller which makes use of observations from tactile sensors and robot encoders to estimate pose of the system. Most manipulation systems are underactuated and unobservable. The tool manipulation system falls under the same umbrella. Thus, we present the design of a tactile estimator which can estimate θ_O, θ_T, $\mathbf{p}_A$, and the length of the object, r_O. Then, we present our MPC-based controller using the estimated states as inputs.

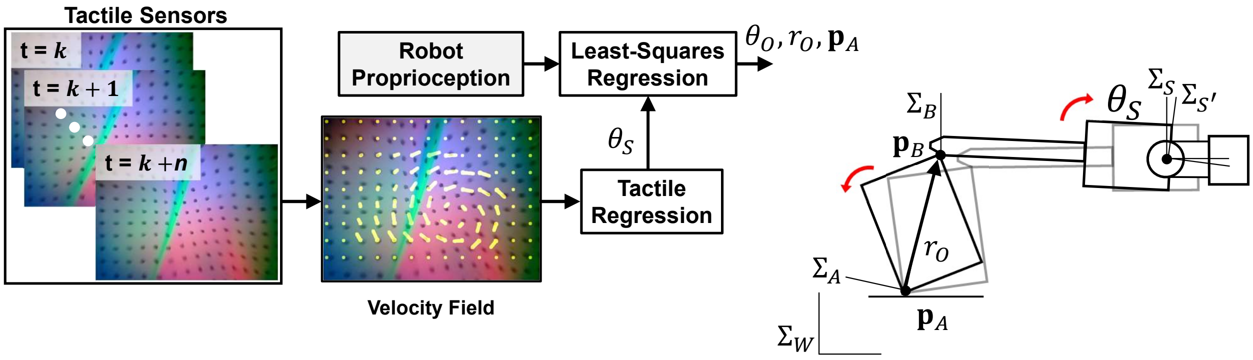
9.3.1 Tactile Stiffness Regression

We use tactile sensors to monitor and estimate the slip of the tool during manipulation. Since the tactile sensors are deformable, we need to identify their stiffness to correctly estimate the slip of objects in grasp. We employ a simple polynomial regression to estimate θ_S (see Fig. 9.3 (b)) given the velocities of all markers as illustrated in Fig. 9.3 (a).

We explain how we train the regression model. Given two images at $t = k$ and $t = k + n, n > 0$, we compute the velocity flow of the markers on the tactile sensors. We use the norm of the velocity flow as input of polynomial regression. We use Apriltag [154] to obtain the ground truth values of θ_S and train the regression algorithm. We observe a nonlinear trend in the stiffness of the sensors, i.e., the sensors become more stiff as they deform.

9.3.2 Tactile Estimator

For our estimator design, we make an assumption that contacts at A and B are sticking. This means that $\mathbf{p}_A$ does not change during pivoting. Using this knowledge, the observed


Tactile Sensors
t = k
t = k + 1
t = k + n
Velocity Field
Robot Proprioception
Least-Squares Regression
$\theta_O, r_O, \mathbf{p}_A$
θ_S
Tactile Regression
Σ_B
$\mathbf{p}_B$
θ_S
Σ_S
$\Sigma_{S'}$
r_O
Σ_A
$\mathbf{p}_A$
Σ_W

(a) (b)

Figure 9.3: **Tactile estimator.** (a): Given measurements of robot proprioception and tactile sensors, our method estimates the state of the object and the tool. (b): Schematic showing tool manipulation experiencing rotational slipping by θ_S.

$\mathbf{p}_B$ at $t = k$, denoted as $\bar{\mathbf{p}}_{B,k}$, can be represented as (see Fig. 9.3 (b)):

$$\left[\bar{\mathbf{p}}_{B,k}^\top, 1\right]^\top = {}^W_S T\, {}^S_{S'}T(\theta_{S,k})\, {}^{S'}_B T \left[\mathbf{0}^\top, 1\right]^\top \tag{9.6}$$

where $\theta_{S,k}$ is the relative rotation of frame at the center of grasp at $t = k$ (we denote this frame as $\Sigma_{S'}$) with respect to the frame at the reference center of grasp at $t = 0$ (we denote this frame as Σ_S). ${}^S_{S'}T, {}^W_S T$ can be obtained from the tactile sensor and encoders, respectively. ${}^{S'}_B T$ is obtained from the known tool kinematics. We can represent $\mathbf{p}_B$ at $t = k$, denoted as $\mathbf{p}_{B,k}$, also as follows:

$$\left[\mathbf{p}_{B,k}^\top, 1\right]^\top = {}^W_A T(\theta_{O,k}, \mathbf{p}_A)\, {}^A_B T(r_O)[\mathbf{0}^\top, 1]^\top \tag{9.7}$$

Then, using (9.6) and (9.7), with time history of measurements from $t = 0$ to $t = m$, we can do non-linear regression based on least-squares:

$$\left\{\theta_{O,k}^*\right\}_{k=0,\ldots,m}, r_O^*, \mathbf{p}_A^* = \operatorname{argmin} \sum_{k=0}^{m} \left\|\mathbf{p}_{B,k} - \bar{\mathbf{p}}_{B,k}\right\|^2 \tag{9.8}$$

Once contact at $\mathbf{p}_A, \mathbf{p}_B$ slip, the estimator is unable to estimate the state of the object anymore like [69], [72].

Remark 3: Similar to [72], our estimator is able to estimate $\theta_{O,k}^*, r_O^*, \mathbf{p}_A^*$. However, similar to [72], this requires a controller that can maintain the desired contact state during estimation. In this work, we assume that we know the object and tool kinematics during control. Thus, we only make use of $\theta^*_{O,k}$. Controller design when the object kinematics is not known fully is left as a future work.

Remark 4: As illustrated in Fig. 9.3, the tool can experience both rotational and translational slip. In practice, we observed that the deformation and high friction at the tactile sensors resulted in minimum translation slip. Thus, we ignored translational slip during manipulation. However, considering the translational slipping might improve the performance of the estimator.

9.3.3 Tactile Controller

Our online controller based on MPC is as follows:

$$\min_{\mathbf{x},\mathbf{u},\mathbf{f}} \sum_{k=t+1}^{N+t} (\theta_{O,k} - \bar{\theta}_{O,k})^2 + \sum_{k=t}^{N+t-1} \mathbf{u}_k^\top R \mathbf{u}_k \tag{9.9a}$$

$$\text{s.t. } (9.5b), (9.5c) \tag{9.9b}$$

where $\bar{\theta}_{O,k}$ represent the reference trajectory computed offline using (9.5). We observed that slipping between the tool and object kept happening if the controller tracks the tool state as well. Thus, we only consider state tracking for θ_O so that the system does not care if θ_T is tracked - it tries to find a new θ_T to track θ_O (i.e., replanning for θ_T).

9.4 Results

In this section, we perform several different experiments to answer the following questions:

1. How do the open-loop trajectories behave on a physical setup?
2. How effective is the proposed closed-loop controller for tool manipulation under different disturbances?

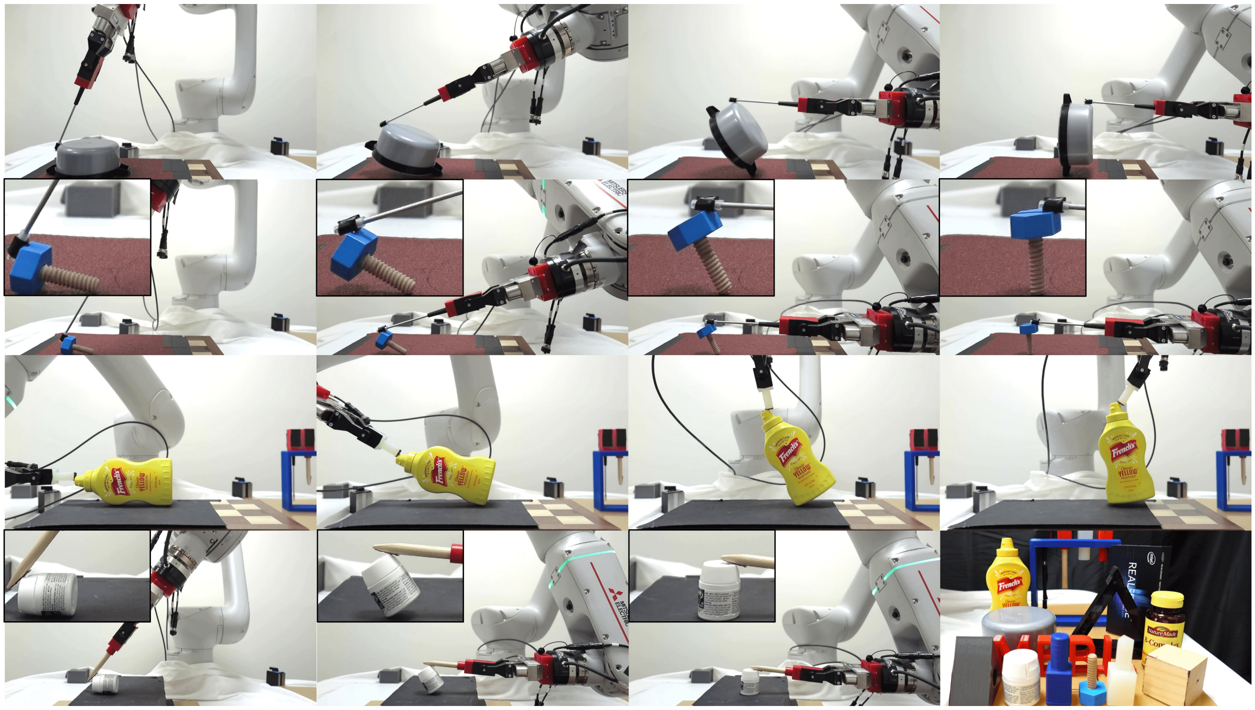


Figure 9.4: **Open-loop tool manipulation.** Our controller could successfully perform tool manipulation with different object-tool-environment pairs. The bottom right picture shows the objects and the tools we use in this chapter.

9.4.1 Experiment Setup

For the planner and controller, we use IPOPT [10] with pyrobocop [16] interface to solve TO. MPC is run with a large horizon of $N = 160$, and thus can only achieve a control rate of 2 Hz. However, the control frequency can be increased by using linearized constraints with a QP solver.

For the hardware experiments, we use a Mitsubishi Electric Assista industrial manipulator arm equipped with a WSG-32 gripper. For the closed-loop experiments, the gripper is equipped with GelSlim 3.0 [169] sensors. We use a stiffness controller to track the reference force trajectory [170, 153]. As shown in Fig. 9.4, we test our framework with 12 different objects, 4 different tools, and 5 different environments (i.e., friction surfaces). We use an Apriltag system to obtain the ground truth for pose of objects.

9.4.2 Open-Loop Controller

In this experiment, we show that our open-loop controller (9.5) generates successful trajectories for different objects, tools, and environments. Note that our framework works even for non-rectangle objects as long as the shape of the object can be approximated as a rectangle in 2D. The results are shown in Fig. 9.4. More results are shown in the supplementary video. Overall, we verified that our open-loop controller could successfully perform tool manipulation with the *carefully-tuned* parameters by the authors.

Through these experiments, we observed the following failure cases:

1. Failure at the beginning of trajectory: We found that the open-loop controller is most susceptible to failure at $t = 0$. This is because the tool needs to make contact with the object at B. It might happen that the contact force at B is too little that it can not support the moment to lift the object up or in the opposite case, it might be too strong so that the object slips at contact C. If too much contact force is applied at contact B, the tool might also rotate at the contact C.

2. Incorrect physical parameters: We observed that the open-loop controller fails with

inaccurate physical parameters such as mass.

3. Unexpected contacts: Since there is no feedback in the open-loop controller, the manipulation fails if we introduce unexpected contacts during the task.

Motivated by these failure cases, we discuss how the closed-loop controller can handle them in Sec 9.4.3.2.

9.4.3 Tactile Estimator and Controller

9.4.3.1 Tactile Estimator Results

In this section, we discuss the results of our tactile estimator. To test the accuracy of our estimator, we perform three different kinds of experiments– the open-loop controller with no external disturbance, the open-loop controller with external disturbance, and the closed-loop controller with external disturbance. In all these experiments, the robot is trying to pivot the same box with the same tool. We perform 5 trials for each experiment. All results are shown in Fig. 9.5.

Our estimator works with the open-loop controller under no disturbances as shown in Fig. 9.5 (a) but does not work under disturbances as shown in Fig. 9.5 (b). Since our estimator assumes that contact is always maintained, once contact is broken (see Fig. 9.5 (b) around $t = 110$ s), the estimator diverges. In contrast, Fig. 9.5 (c) shows that our estimator works under disturbances since our MPC controller can react to the disturbance and maintain the desired contact state.

9.4.3.2 Tactile Controller Results

We demonstrate the effectiveness of our tactile controller to recover from different unexpected contacts in Sec 9.4.2.

We first discuss recovery from slipping of the tool in the gripper fingers, i.e., non-zero θ_S (see Fig. 9.3 for definition of θ_S) at $t = 0$ as described in failure case #1 in Sec 9.4.2. We

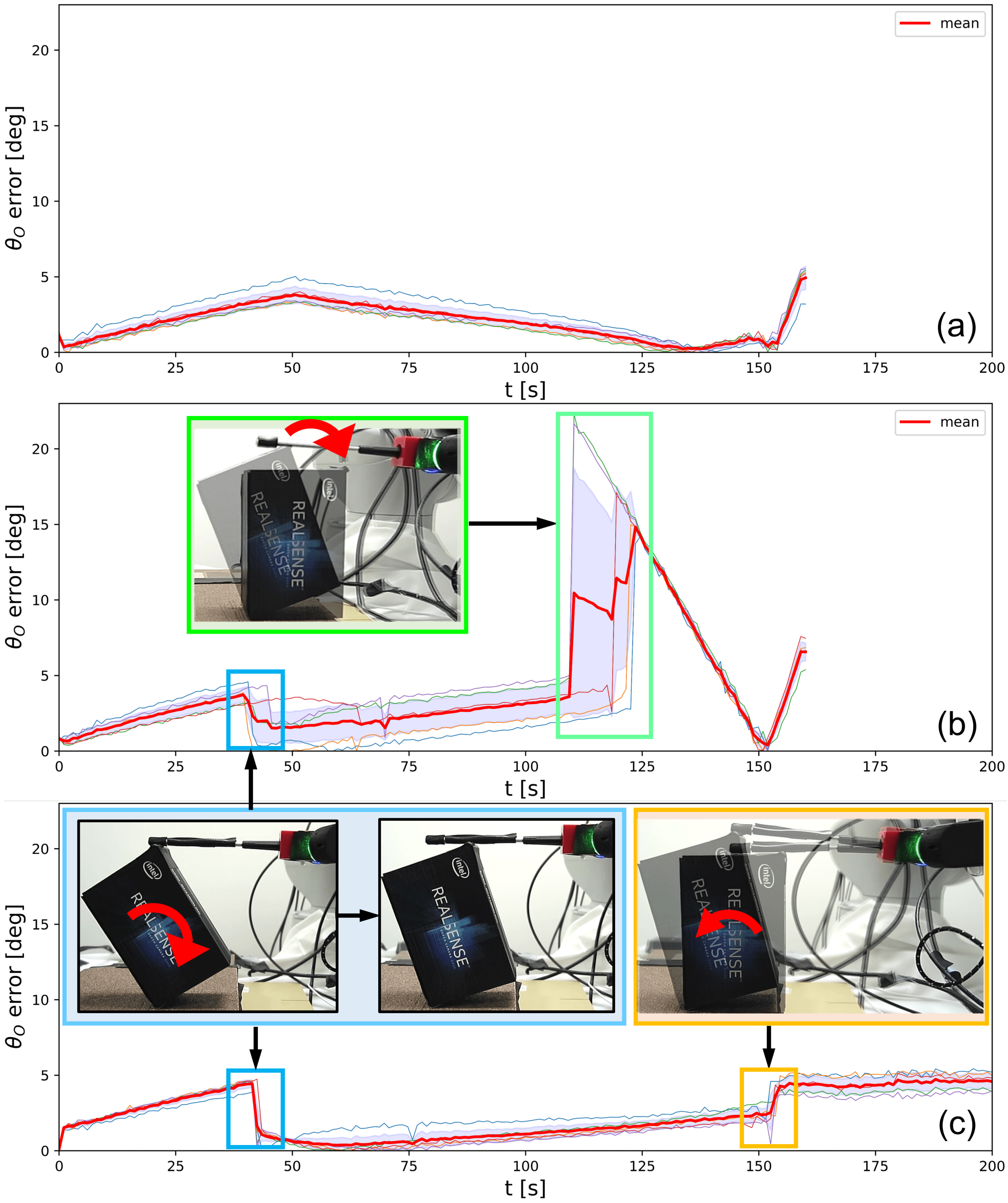


Figure 9.5: **Evaluation of the tactile estimator.** We show the time history of error of θ_O for 5 trials (a) with the open-loop controller under no disturbance, (b) with the open-loop controller under disturbance, and (c): with the closed-loop controller under disturbance. The red line shows the mean of and the blue region shows the 95% confidence interval. We added disturbance around $t = 40$ s for (b) and (c) (see the blue box). For (c), we added another disturbance around $t = 150$ s (see the orange box). The contact is lost around $t = 110$ s for (b) (see the green box). Note that for the open-loop controller, the trajectory runs until $t = 160$ s because open-loop controller is pre-defined.

implemented the open- and closed-loop controllers with the above disturbance at $t = 0$. We did this experiment for 5 trials per controller. We declare failure if the contact is broken. The result is summarized in Table 9.2 (Disturbance 1). For $\theta_S = 5°$, both the open- and the closed-loop controllers could complete the task. However, for $\theta_S = 10°, 15°$, we observed that the open-loop controller lost the contact between the tool and object around $t = 110$ s (see Fig. 9.5 (b)) while the closed-loop controller could still successfully conduct the pivoting.

Next, we discuss how the closed-loop controller reacts to different unexpected contacts during the trajectory to tackle the failure case #3 in Sec 9.4.2. In these experiments, we add disturbance to the object (see blue and orange box in Fig. 9.5 (c)) around $t = 40$ s and $t = 150$ s. We conducted 5 trials. The time history of the object pose θ_O and the gripper angle θ_G is shown in Fig. 9.6. Fig. 9.6 (a) shows that the closed-controller could successfully track the reference trajectory even under these unexpected contacts. The reactive control efforts can be observed around $t = 40, 150$ s in Fig. 9.6 (b). The robot changes its gripper orientation to maintain the constraints discussed in Sec 9.2. The results discussed here are also summarized in Table 9.2 (Disturbance 2).

Finally, we demonstrate the closed-loop controller with incorrect mass (failure case #2 in Sec 9.4.2). In these experiments, we solve (9.5) and (9.9) with mass different from the true mass of the object and use the solution for implementation. The results are summarized in Table 9.3. We observed that the closed-loop controller can always successfully pivot the object while the open-loop controller fails especially once m_O is quite different from the true value. The open-loop controller can also work with significantly different m_O as the tactile sensors have significant compliance. This provides some inherent stability to the system during this task. Modeling this compliance and utilizing the model inside MPC as robust tube MPC is an interesting direction [171].

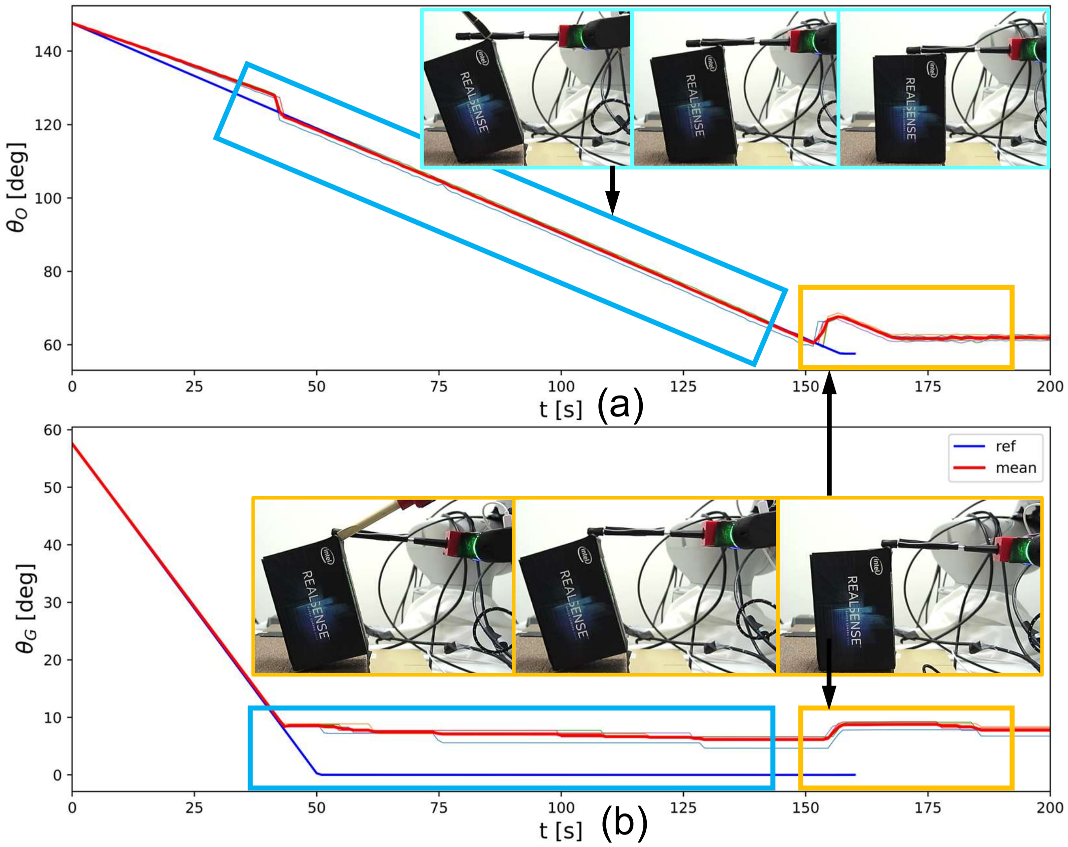


Figure 9.6: **Evaluation of the closed tactile controller.** We show time history of (a) θ_O and (b) θ_G, with the closed-loop controller under disturbances at $t = 40, 150$ s. The blue line is the reference trajectory computed offline and the red trajectory is the mean of the 5 trajectories computed online.

Table 9.2: **Evaluation of the closed tactile controller with disturbances**. The number of successful pivoting attempts of the box over 5 trials for different disturbances are summarized.

Box	Disturbance 1			Disturbance 2	
	5°	10°	15°	$t = 40$ s	$t = 150$ s
Open-loop	4/5	0/5	0/5	0/5	N/A
Close-loop	5/5	5/5	5/5	5/5	5/5

Table 9.3: **Evaluation of the closed tactile controller with inaccurate parameters**. The number of successful pivoting attempts of the box over 5 trials for different mass of the object are summarized. The true value of mass of the object is $m_O = 100$ g.

m_O [g]	15	200	1000
Open-loop	4/5	3/5	0/5
Close-loop	5/5	5/5	5/5

9.5 Conclusion

Closed-loop control of manipulation remains elusive. This is because contacts lead to complex, discontinuous constraints that need to be carefully handled. In this chapter, we presented tactile tool manipulation. More specifically, we presented the design and implementation of a closed-loop controller to control the complex mechanics of tool manipulation using tactile sensors and NLP. Through extensive experiments, we demonstrate that the proposed method provides robustness against parametric uncertainties as well as unexpected contact events during manipulation.

CHAPTER 10

Conclusion

In this chapter, we summarize our contributions. Then, we describe the limitations of our works presented in this dissertation and the potential future works.

10.1 Summary of Contributions

Robotic manipulation and locomotion have many research challenges because of contact. In this dissertation, we present our planning and control methods for contact-rich manipulation and locomotion. We show that unlike conventional planning and control framework for robots whose dynamics are continuous, planning and control (under uncertainty) for robotic manipulation and locomotion whose dynamics are non-smooth requires careful treatment to accomplish the mission.

In Chapter 3, we present a framework that uses TO as short-horizon planning and GSP as long-horizon planning, which results in better computation for complex motion planning problems.

In Chapter 4, we present a ADMM-based planner that solves MIQP for discrete decision-making (e.g., contact mode) and NLP for continuous decision-making (e.g., nonlinear dynamics of the body of the robot). Using our method, we could successfully solve the original computationally demanding motion planning for multi-limbed robots which is formulated as MINLP. We also present our patch-contact model for free-climbing tasks. Our planner is able to design various trajectories and we verify them in hardware experiments.

In Chapter 5, we present chance-constrained optimization for motion planning of multi-limbed robots for free-climbing where contact modes do not change. Our planner is able to design risk-aware trajectories based on user-defined violation probability. We also propose our leaning-based stochastic contact model using Gaussian Process Regression and we employ it to represent the distribution of the stochastic contact forces in our chance-constrained optimization.

In Chapter 6, we analyze the stochasticity in DLCS. In particular, we formulate chance-constrained optimization for contact-rich systems where contact modes can change. We propose MIQPCC and its MPC to achieve robust planning and control for SDLCS. However, we have some assumptions that limit the application of our algorithm in Chapter 6.

In Chapter 7, we analyze the frictional stability margin under different physical parameters such as mass, CoM location, coefficients of friction, and contact location for the pivoting manipulation. Using this analysis, we present our contact implicit bilevel optimization problem where the optimizer designs an optimal control input while improving the worst-case stability margin along the manipulation. Our work provides deep insights about the pivoting manipulation. Our algorithm is verified in various experiments including the hardware experiments.

In Chapter 8, we present our covariance steering for uncertain contact-rich systems. Our framework is able to design feedforward and feedback gains of linear controller over state and contact measurements. Our controller is robust against the distribution of SDLCS. Our framework uses particle-filter-based optimization to approximate the distribution of SDLCS and thus our framework is very general and can be applied to other robotic manipulation and locomotion tasks.

In Chapter 9, we present closed-loop controller for tool manipulation using NLP and visuotactile sensors. We first analyze the complex tool manipulation mechanics. We then formulate our planner and controller using NLP. We formulate our pose estimator using visuotactile sensors, which can estimate the pose of an object and a robot simultaneously. Using our framework, the robot is able to achieve the pivoting manipulation using an external tool even under unexpected contact disturbances.

10.2 Limitations and Future Works

10.2.1 Computational Complexity of ADMM

One limitation of our ADMM in Chapter 4 is that the computation is demanding once the number of discrete constraints increases. In order to use our framework in MPC, we need to run it with a much faster runtime. One promising direction is to design heuristics online [112]. We hope that we can accelerate our framework as ADMM iteration proceeds based on previous solutions.

10.2.2 Propagation of Uncertainty for SDLCS Analytically

In Chapter 8, we use particles to approximate the distribution of SDLCS, which makes the computational complexity increase dramatically. Thus, it would be very beneficial if we can an analytical representation of propagation of uncertainty for SDLCS as we have for linear dynamical systems with additive Gaussian noises in Kalman filter. Moment-based approach [40] can be promising although it is not clear how to consider the complementarity constraints in moment propagation.

Another challenge arises from P-matrix assumption. If we do not assume that matrix F is not P-matrix in SDLCS, we cannot even do forward dynamics calculation. Therefore, one strategy can be enforcing some constraints so that the system is guaranteed to have P-matrix property even under uncertainty.

10.2.3 Contact-Rich CIBO

In Chapter 7, we assume quasi-static assumption for objects. The natural extension of this work is to relax this assumption and consider quasi-dynamic model during manipulation. To work on these cases, we need to explicitly consider dynamic version of the stability margin. However, this is not trivial. We need to understand how we can propagate uncertainty for contact dynamics as it is not well understood. The stability margin needs to incorporate this uncertainty propagation for such cases. See [44] for more discussion about how uncertainty propagates for contact-rich dynamical systems.

10.2.4 Accurate Contact Mechanics

The natural extension of the work in Chapter 7 and Chapter 9 is to consider mechanics in 3D with generalized friction cones [167], which is not trivial. Additionally, the system has compliance at the contact locations and we believe that modeling the compliance would lead to a more effective and precise closed-loop controller.

10.2.5 Analysis of Controllability and Observability for Dexterous Manipulation

One of the fundamental questions that remains open in manipulation is that of controllability and observability. There have been remarkable works in controllability and observability for manipulation [172, 173, 174]. However, the theory of controllability and observability is limited to more dexterous manipulation (e.g., tool manipulation as we present in Chapter 9,). This limits the generality of model-based controller design for manipulation. Therefore, it would be useful to understand and study controllability and observability for frictional interaction tasks.

10.2.6 Hysteresis of Visuotactile Sensors

In Chapter 9, we use GelSlim 3 as a visuotactile sensor and use it to estimate the angle of the tool with respect to the gripper. However, we observe hysteresis of GelSlim in the rotational slipping of the tool with respect to the gripper. It means that the image from GelSlim with the zero rotational angle after slipping is different from the image with the zero rotational angle before slipping. In order to improve the performance of the closed-loop controller, it would be important to take into consideration hysteresis in the proposed framework.

10.2.7 Learning Manipulation Skills with Model-based Optimization

In this dissertation, all of our major contributions are based on model-based optimization. Although model-based optimization has shown impressive results for various manipulation and locomotion skills, it is often limited to the skills in highly structured environments. We observe that learning-based algorithms have shown quite robust performance, especially in the robotic locomotion community [175, 176]. However, it is in general quite challenging to make these algorithms work in robotic manipulation. In manipulation, objects do not have any actuators and sensors, meaning that they are zero-actuated and partially observable systems. Thus simply using learning-based algorithms might not work for dexterous manipulation skills such as tool manipulation. Thus, we are currently interested in using our model-based optimizer for providing learning framework with some initial guesses (i.e., warm-start) because our optimizer is able to design various complex manipulation tasks even under uncertainty. In practice, there are many details to make the system work.

Bibliography

[1] O. Stein, J. Oldenburg, and W. Marquardt, "Continuous reformulations of discrete–continuous optimization problems," *Comp. Chem. Eng.*, vol. 28, no. 10, pp. 1951–1966, 2004.

[2] L. Drnach and Y. Zhao, "Robust trajectory optimization over uncertain terrain with stochastic complementarity," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1168–1175, 2021.

[3] L. Drnach, J. Z. Zhang, and Y. Zhao, "Mediating between contact feasibility and robustness of trajectory optimization through chance complementarity constraints," *Front. Robo. AI*, vol. 8, 2021.

[4] M. T. Mason, "Toward robotic manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 1–28, 2018.

[5] "Boston dynamics." `https://bostondynamics.com/`. Accessed: 2023-12-31.

[6] A. Zermane, N. Dehio, and A. Kheddar, "Planning impact-driven logistic tasks," *IEEE Robotics and Automation Letters*, pp. 1–8, 2024.

[7] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research*, vol. 33, pp. 69–81, jan 2014.

[8] E. Todorov, "Implicit nonlinear complementarity: A new approach to contact dynamics," in *2010 IEEE international conference on robotics and automation*, pp. 2322–2329, IEEE, 2010.

[9] B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini, "Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2531–2538, 2018.

[10] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[11] P. E. Gill, W. Murray, and M. A. Saunders, "Snopt: An sqp algorithm for large-scale constrained optimization," *SIAM review*, vol. 47, no. 1, pp. 99–131, 2005.

[12] E. Drumwright and D. A. Shell, "An evaluation of methods for modeling contact in multibody simulation," in *2011 IEEE International Conference on Robotics and Automation*, pp. 1695–1701, IEEE, 2011.

[13] J. Carius, R. Ranftl, V. Koltun, and M. Hutter, "Trajectory optimization for legged robots with slipping motions," *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 3013–3020, 2019.

[14] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and Trajectory Optimization for Legged Systems through Phase-based End-Effector Parameterization," *IEEE Robotics and Automation Letters*, pp. 1–8, 2018.

[15] Y. Zhu, Z. Pan, and K. Hauser, "Contact-implicit trajectory optimization with learned deformable contacts using bilevel optimization," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9921–9927, 2021.

[16] A. U. Raghunathan, D. K. Jha, and D. Romeres, "Pyrobocop: Python-based robotic control & optimization package for manipulation," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 985–991, 2022.

[17] Y. Shirai, D. K. Jha, A. U. Raghunathan, and D. Romeres, "Robust pivoting manipulation using bilevel contact-implicit optimization," in *RSS 2022 Workshop on The Science of Bumping Into Things Towards Robots That Aren't Afraid of Contact*, 2022.

[18] X. Lin, G. I. Fernandez, Y. Liu, T. Zhu, Y. Shirai, and D. Hong, "Multi-modal multi-agent optimization for limms, a modular robotics approach to delivery automation," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 12674–12681, IEEE, 2022.

[19] A. Aydinoglu, V. M. Preciado, and M. Posa, "Contact-aware controller design for complementarity systems," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1525–1531, 2020.

[20] H. Li, T. Zhang, W. Yu, and P. M. Wensing, "Versatile real-time motion synthesis via kino-dynamic mpc with hybrid-systems ddp," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9988–9994, IEEE, 2023.

[21] G. Kim, D. Kang, J.-H. Kim, S. Hong, and H.-W. Park, "Contact-implicit mpc: Controlling diverse quadruped motions without pre-planned contact modes or trajectories," *arXiv preprint arXiv:2312.08961*, 2023.

[22] H. Zhu, A. Meduri, and L. Righetti, "Efficient object manipulation planning with monte carlo tree search," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10628–10635, IEEE, 2023.

[23] V. Kurtz, A. Castro, A. Ö. Önol, and H. Lin, "Inverse dynamics trajectory optimization for contact-implicit model predictive control," *arXiv preprint arXiv:2309.01813*, 2023.

[24] T. Stouraitis, I. Chatzinikolaidis, M. Gienger, and S. Vijayakumar, "Online hybrid motion planning for dyadic collaborative manipulation via bilevel optimization," *IEEE Trans. Robot.*, vol. 36, no. 5, pp. 1452–1471, 2020.

[25] M. P. Polverini, A. Laurenzi, E. M. Hoffman, F. Ruscelli, and N. G. Tsagarakis, "Multi-contact heavy object pushing with a centaur-type humanoid robot: Planning and control for a real demonstrator," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 859–866, 2020.

[26] Y. Shirai, X. Lin, Y. Tanaka, A. Mehta, and D. Hong, "Risk-Aware Motion Planning for a Limbed Robot with Stochastic Gripping Forces Using Nonlinear Programming," *IEEE Robotics and Automation Letters*, vol. 5, pp. 4994–5001, oct 2020.

[27] I. Kumagai, M. Murooka, M. Morisawa, and F. Kanehiro, "Multi-contact locomotion planning with bilateral contact forces considering kinematics and statics during contact transition," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 6654–6661, 2021.

[28] C. Nguyen and Q. Nguyen, "Contact-timing and trajectory optimization for 3d jumping on quadruped robots," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 11994–11999, IEEE, 2022.

[29] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[30] R. Budhiraja, J. Carpentier, and N. Mansard, "Dynamics consensus between centroidal and whole-body models for locomotion of legged robots," in *2019 Int. Conf. Robot. Automat.*, pp. 6727–6733, 2019.

[31] Z. Zhou and Y. Zhao, "Accelerated admm based trajectory optimization for legged locomotion with coupled rigid body dynamics," in *Proc. 2020 American Control Conference*, pp. 5082–5089, 2020.

[32] A. Aydinoglu and M. Posa, "Real-time multi-contact model predictive control via admm," in *Proc. 2022 IEEE Int. Conf. Robot. Automat.*, pp. 3414–3421, 2022.

[33] O. Shorinwa and M. Schwager, "Distributed contact-implicit trajectory optimization for collaborative manipulation," in *Proc. 2021 Int. Symp. Multi. Robo. Multi. Agent. Syst.*, pp. 56–65, 2021.

[34] A. Geletu, M. Klöppel, H. Zhang, and P. Li, "Advances and applications of chance-constrained approaches to systems optimisation under uncertainty," *International Journal of Systems Science*, vol. 44, no. 7, pp. 1209–1232, 2013.

[35] L. Blackmore, Hui Li, and B. Williams, "A probabilistic approach to optimal robust path planning with obstacles," in *2006 American Control Conference*, vol. 2006, p. 7 pp., IEEE, 2006.

[36] L. Blackmore and M. Ono, "Convex chance constrained predictive control without sampling," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2009.

[37] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust optimization*, vol. 28. Princeton university press, 2009.

[38] K. Okamoto, M. Goldshtein, and P. Tsiotras, "Optimal Covariance Control for Stochastic Systems under Chance Constraints," *IEEE Control Systems Letters*, vol. 2, no. 2, pp. 266–271, 2018.

[39] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram, "Chance-constrained dynamic programming with application to risk-aware robotic space exploration," *Autonomous Robots*, vol. 39, pp. 555–571, 2015.

[40] A. Jasour, A. Wang, and B. C. Williams, "Moment-based exact uncertainty propagation through nonlinear stochastic autonomous systems," *arXiv preprint arXiv:2101.12490*, 2021.

[41] Y. Tassa and E. Todorov, "Stochastic complementarity for local control of discontinuous dynamics," *Robotics: Science and Systems VI*, 2010.

[42] X. Chen and M. Fukushima, "Expected residual minimization method for stochastic linear complementarity problems," *Mathematics of Operations Research*, vol. 30, no. 4, pp. 1022–1038, 2005.

[43] A. Prékopa, "Boole-bonferroni inequalities and linear programming," *Operations Research*, vol. 36, no. 1, pp. 145–162, 1988.

[44] Y. Shirai, D. K. Jha, and A. U. Raghunathan, "Covariance steering for uncertain contact-rich systems," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7923–7929, 2023.

[45] A. Wang, A. Jasour, and B. C. Williams, "Non-gaussian chance-constrained trajectory planning for autonomous vehicles under agent uncertainty," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6041–6048, 2020.

[46] T. Bretl and S. Lall, "Testing static equilibrium for legged robots," *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 794–807, 2008.

[47] A. Del Prete, S. Tonneau, and N. Mansard, "Zero step capturability for legged robots in multicontact," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1021–1034, 2018.

[48] K. Hauser, S. Wang, and M. R. Cutkosky, "Efficient equilibrium testing under adhesion and anisotropy using empirical contact force models," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1157–1169, 2018.

[49] R. Orsolino, M. Focchi, C. Mastalli, H. Dai, D. G. Caldwell, and C. Semini, "Application of wrench-based feasibility analysis to the online trajectory optimization of legged robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3363–3370, 2018.

[50] H. Dai and R. Tedrake, “Planning robust walking motion on uneven terrain via convex optimization,” in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 579–586, IEEE, 2016.

[51] H. Audren and A. Kheddar, “3-d robust stability polyhedron in multicontact,” *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 388–403, 2018.

[52] Y. Hou, Z. Jia, and M. Mason, “Manipulation with shared grasping,” in *Robotics: Science and Systems*, 2020.

[53] F. R. Hogan, J. Ballester, S. Dong, and A. Rodriguez, “Tactile dexterity: Manipulation primitives with tactile feedback,” in *2020 IEEE international conference on robotics and automation (ICRA)*, pp. 8863–8869, IEEE, 2020.

[54] E. Donlon, S. Dong, M. Liu, J. Li, E. Adelson, and A. Rodriguez, “Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1927–1934, IEEE, 2018.

[55] W. Li, A. Alomainy, I. Vitanov, Y. Noh, P. Qi, and K. Althoefer, “F-touch sensor: Concurrent geometry perception and multi-axis force measurement,” *IEEE Sensors Journal*, vol. 21, no. 4, pp. 4300–4309, 2020.

[56] F. E. Viña B., Y. Karayiannidis, K. Pauwels, C. Smith, and D. Kragic, “In-hand manipulation using gravity and controlled slip,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5636–5641, 2015.

[57] F. E. Viña B., Y. Karayiannidis, C. Smith, and D. Kragic, “Adaptive control for pivoting with visual and tactile feedback,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 399–406, 2016.

[58] S. Cruciani and C. Smith, “In-hand manipulation using three-stages open loop pivoting,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1244–1251, 2017.

[59] N. C. Dafle, A. Rodriguez, R. Paolini, B. Tang, S. S. Srinivasa, M. Erdmann, M. T. Mason, I. Lundberg, H. Staab, and T. Fuhlbrigge, “Extrinsic dexterity: In-hand manipulation with external forces,” in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1578–1585, 2014.

[60] Y. Hou, Z. Jia, and M. T. Mason, “Fast planning for 3d any-pose-reorienting using pivoting,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1631–1638, IEEE, 2018.

[61] B. Aceituno-Cabezas and A. Rodriguez, “A global quasi-dynamic model for contact-trajectory optimization,” in *Robotics: Science and Systems (RSS)*, 2020.

[62] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[63] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[64] A. Wang, X. Huang, A. Jasour, and B. Williams, "Fast risk assessment for autonomous vehicles using learned models of agent futures," *arXiv preprint arXiv:2005.13458*, 2020.

[65] A. Hotz and R. E. Skelton, "Covariance control theory," *International Journal of Control*, vol. 46, no. 1, pp. 13–32, 1987.

[66] R. Holladay, T. Lozano-Pérez, and A. Rodriguez, "Force-and-motion constrained planning for tool use," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7409–7416, IEEE, 2019.

[67] K. Fang, Y. Zhu, A. Garg, A. Kurenkov, V. Mehta, L. Fei-Fei, and S. Savarese, "Learning task-oriented grasping for tool manipulation from simulated self-supervision," *The International Journal of Robotics Research*, vol. 39, no. 2-3, pp. 202–216, 2020.

[68] G. Izatt, G. Mirano, E. Adelson, and R. Tedrake, "Tracking objects with point clouds from vision and touch," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4000–4007, 2017.

[69] D. Ma, S. Dong, and A. Rodriguez, "Extrinsic contact sensing with relative-motion tracking from distributed tactile measurements," in *Proc. 2021 IEEE Int. Conf. Robot. Automat.*, pp. 11262–11268, 2021.

[70] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese, "Keto: Learning keypoint representations for tool manipulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7278–7285, 2020.

[71] F. R. Hogan, E. R. Grau, and A. Rodriguez, "Reactive planar manipulation with convex hybrid mpc," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 247–253, 2018.

[72] N. Doshi, O. Taylor, and A. Rodriguez, "Manipulation of unknown objects via contact configuration regulation," in *Proc. 2022 IEEE Int. Conf. Robot. Automat.*, pp. 2693–2699, 2022.

[73] S. Dong, D. K. Jha, D. Romeres, S. Kim, D. Nikovski, and A. Rodriguez, "Tactile-RL for insertion: Generalization to objects of unknown geometry," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6437–6443, 2021.

[74] O. Taylor, N. Doshi, and A. Rodriguez, "Object manipulation through contact configuration regulation: multiple and intermittent contacts," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8735–8743, IEEE, 2023.

[75] A. K. Valenzuela, “Mixed-integer convex optimization for planning aggressive motions of legged robots over rough terrain,” *Massachusetts Institute of Technology*, 2016.

[76] Z. Kingston, M. Moll, and L. E. Kavraki, “Sampling-Based Methods for Motion Planning with Constraints,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 159–185, 2018.

[77] J. Ibarz, J. Tan, C. Finn, M. Kalakrishnan, P. Pastor, and S. Levine, “How to train your robot with deep reinforcement learning: lessons we have learned,” *The International Journal of Robotics Research*, p. 027836492098785, jan 2021.

[78] M. Kelly, “An introduction to trajectory optimization: How to do your own direct collocation,” *SIAM Review*, vol. 59, no. 4, pp. 849–904, 2017.

[79] B. Cohen, M. Phillips, and M. Likhachev, “Planning Single-arm Manipulations with n-Arm Robots,” in *Robotics: Science and Systems X*, vol. 2015-Janua, pp. 226–227, Robotics: Science and Systems Foundation, jul 2014.

[80] Y. Kuwata and J. P. How, “Cooperative distributed robust trajectory optimization using receding horizon milp,” *IEEE Transactions on Control Systems Technology*, vol. 19, no. 2, pp. 423–431, 2011.

[81] R. Fletcher and S. Leyffer, “Numerical experience with lower bounds for miqp branch-and-bound,” *SIAM Journal on Optimization*, vol. 8, no. 2, pp. 604–616, 1998.

[82] M. Conforti, G. Cornuejols, and G. Zambelli, “Integer programming,” *Springer*, vol. 271, 2014.

[83] P. E. Hart, N. J. Nilsson, and B. Raphael, “A formal basis for the heuristic determination of minimum cost paths,” *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[84] L. Campos-Macías, D. Gómez-Gutiérrez, R. Aldana-López, R. De La Guardia, and J. I. Parra-Vilchis, “A Hybrid Method for Online Trajectory Planning of Mobile Robots in Cluttered Environments,” *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 935–942, 2017.

[85] K. Bergman, O. Ljungqvist, and D. Axehill, “Improved Path Planning by Tightly Combining Lattice-Based Path Planning and Optimal Control,” *IEEE Transactions on Intelligent Vehicles*, vol. 6, pp. 57–66, mar 2021.

[86] X. Zhang, A. Liniger, and F. Borrelli, “Optimization-based collision avoidance,” *IEEE Transactions on Control Systems Technology*, pp. 1–12, 2020.

[87] C. M. Dellin and S. S. Srinivasa, “A unifying formalism for shortest path problems with expensive edge evaluations via lazy best-first search over paths with edge selectors,” *Proceedings International Conference on Automated Planning and Scheduling, ICAPS*, vol. 2016-Janua, pp. 459–467, 2016.

[88] B. Ponton, A. Herzog, S. Schaal, and L. Righetti, “A convex model of humanoid momentum dynamics for multi-contact motion generation,” in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 842–849, 2016.

[89] M. Likhachev, G. Gordon, and S. Thrun, “ARA *: Anytime A * with Provable Bounds on,” *Science*, pp. 767—-774, 2004.

[90] L. E. Kavraki, P. Svestka, J. . Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[91] R. Bohlin and L. E. Kavraki, “Path planning using lazy prm,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, pp. 521–528 vol.1, 2000.

[92] J. J. Kuffner and S. M. LaValle, “Rrt-connect: An efficient approach to single-query path planning,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, pp. 995–1001 vol.2, 2000.

[93] A. Shkolnik, M. Levashov, I. R. Manchester, and R. Tedrake, “Bounding on rough terrain with the LittleDog robot,” *International Journal of Robotics Research*, vol. 30, no. 2, pp. 192–215, 2011.

[94] G. Optimization, “Llc,” in *Gurobi Optimizer Reference Manual*, p. 0, 2020.

[95] A. Richards and J. How, “Mixed-integer programming for control,” *Proceedings of the American Control Conference*, vol. 4, pp. 2676–2683, 2005.

[96] T. Koch, “Miplib 2010,” in *Math. Program. Comp.*, vol. 3, pp. 103–163, 2011.

[97] S. Aine, S. Swaminathan, V. Narayanan, V. Hwang, and M. Likhachev, “Multi-Heuristic A*,” *International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 224–243, 2016.

[98] F. Shi, T. Homberger, J. Lee, T. Miki, M. Zhao, F. Farshidian, K. Okada, M. Inaba, and M. Hutter, “Circus anymal: A quadruped learning dexterous manipulation with its limbs,” in *Proc. 2021 IEEE Int. Conf. Robot. Automat.*, 2021.

[99] K. Bouyarmane and A. Kheddar, “Humanoid robot locomotion and manipulation step planning,” *Adv. Robot.*, vol. 26, no. 10, pp. 1099–1126, 2012.

[100] M. Murooka, I. Kumagai, M. Morisawa, F. Kanehiro, and A. Kheddar, “Humanoid loco-manipulation planning based on graph search and reachability maps,” *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1840–1847, 2021.

[101] Y. Shirai, X. Lin, A. Mehta, and D. Hong, “Lto: lazy trajectory optimization with graph-search planning for high dof robots in cluttered environments,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7533–7539, IEEE, 2021.

[102] Y. Shirai, H. Minote, K. Nagaoka, and K. Yoshida, “Gait analysis of a free-climbing robot on sloped terrain for lunar and planetary exploration,” in *Proc. Int. Symp. Artif. Intell. Robot. Autom*, 2018.

[103] A. Parness, N. Abcouwer, C. Fuller, N. Wiltsie, J. Nash, and B. Kennedy, “Lemur 3: A limbed climbing robot for extreme terrain mobility in space,” in *Proc. 2017 IEEE Int. Conf. Robot. Automat.*, pp. 5467–5473, 2017.

[104] K. Uno, N. Takada, T. Okawara, K. Haji, A. Candalot, W. F. R. Ribeiro, K. Nagaoka, and K. Yoshida, “Hubrobo: A lightweight multi-limbed climbing robot for exploration in challenging terrain,” in *Proc. 2020 IEEE-RAS Conf. Humanoid Robots*, pp. 209–215, 2021.

[105] K. Uno, W. F. Ribeiro, W. Jones, Y. Shirai, H. Minote, K. Nagaoka, and K. Yoshida, “Gait planning for a free-climbing robot based on tumble stability,” in *2019 IEEE/SICE International Symposium on System Integration (SII)*, pp. 289–294, IEEE, 2019.

[106] R. D. Howe and M. R. Cutkosky, “Practical force-motion models for sliding manipulation,” *Int. J. Rob. Res.*, vol. 15, no. 6, pp. 557–572, 1996.

[107] S. Wang, H. Jiang, and M. R. Cutkosky, “Design and modeling of linearly-constrained compliant spines for human-scale locomotion on rocky surfaces,” *Int. J. Rob. Res.*, vol. 36, no. 9, pp. 985–999, 2017.

[108] Y. Tanaka, Y. Shirai, X. Lin, A. Schperberg, H. Kato, A. Swerdlow, N. Kumagai, and D. Hong, “Scaler: A tough versatile quadruped free-climber robot,” in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5632–5639, IEEE, 2022.

[109] Y. Tanaka, Y. Shirai, A. Schperberg, X. Lin, and D. Hong, “Scaler: Versatile multi-limbed robot for free-climbing in extreme terrains,” *arXiv preprint arXiv:2312.04856*, 2023.

[110] Y. Tanaka, Y. Shirai, Z. Lacey, X. Lin, J. Liu, and D. Hong, “An under-actuated whippletree mechanism gripper based on multi-objective design optimization with auto-tuned weights,” in *Proc. 2021 Int. Conf. Intell. Rob. Syst.*, pp. 6139–6146, 2021.

[111] A. Schperberg, Y. Shirai, X. Lin, Y. Tanaka, and D. Hong, “Adaptive force controller for contact-rich robotic systems using an unscented kalman filter,” in *Proc. 2023 IEEE-RAS 23rd International Conference on Humanoid Robots*, 2023.

[112] T. Marcucci and R. Tedrake, “Warm start of mixed-integer programs for model predictive control of hybrid systems,” *IEEE Trans. Auto. Cont.*, vol. 66, no. 6, pp. 2433–2448, 2021.

[113] T. Bretl, S. Lall, J.-C. Latombe, and S. Rock, “Multi-step motion planning for free-climbing robots,” *Algorithmic Foundations of Robotics VI*, pp. 59–74, 2005.

[114] K. Nagaoka, H. Minote, K. Maruya, Y. Shirai, K. Yoshida, T. Hakamada, H. Sawada, and T. Kubota, “Passive Spine Gripper for Free-Climbing Robot in Extreme Terrain,” *IEEE Robotics and Automation Letters*, vol. 3, pp. 1765–1770, jul 2018.

[115] X. Lin, H. Krishnan, Y. Su, and D. W. Hong, “Multi-limbed robot vertical two wall climbing based on static indeterminacy modeling and feasibility region analysis,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4355–4362, IEEE, 2018.

[116] A. Pashkevich, A. Klimchik, and D. Chablat, “Enhanced stiffness modeling of manipulators with passive joints,” *Mechanism and Machine Theory*, vol. 46, no. 5, pp. 662–679, 2011.

[117] M. SEEGER, “Gaussian processes for machine learning,” *International Journal of Neural Systems*, vol. 14, no. 02, pp. 69–106, 2004. PMID: 15112367.

[118] H. Jiang, S. Wang, and M. R. Cutkosky, “Stochastic models of compliant spine arrays for rough surface grasping,” *Int. J. Rob. Res.*, vol. 37, no. 7, pp. 669–687, 2018.

[119] A. T. Asbeck, S. Kim, M. R. Cutkosky, W. R. Provancher, and M. Lanzetta, “Scaling hard vertical surfaces with compliant microspine arrays,” in *Robotics: Science and Systems I*, vol. 1, pp. 193–200, Robotics: Science and Systems Foundation, jun 2005.

[120] K. Autumn, A. Dittmore, D. Santos, M. Spenko, and M. Cutkosky, “Frictional adhesion: a new angle on gecko attachment,” *Journal of Experimental Biology*, vol. 209, no. 18, pp. 3569–3579, 2006.

[121] R. Fletcher, “Practical methods of optimization,” *Hoboken,NJ, USA: Wiley*, vol. 2, 1981.

[122] X. Lin, J. Zhang, J. Shen, G. Fernandez, and D. W. Hong, “Optimization Based Motion Planning for Multi-Limbed Vertical Climbing Robots,” *IEEE International Conference on Intelligent Robots and Systems*, 2019.

[123] M. Ono and B. C. Williams, “An efficient motion planning algorithm for stochastic dynamic systems with constraints on probability of failure,” *Proceedings of the National Conference on Artificial Intelligence*, vol. 3, pp. 1376–1382, 2008.

[124] J. A. E. Andersson, “Casadi: a software framework for nonlinear optimization and optimal control,” *Math. Prog. Comp.*, vol. 11, 2019.

[125] O. S. Tas and C. Stiller, “Limited visibility and uncertainty aware motion planning for automated driving,” in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1171–1178, 2018.

[126] B. Brogliato, *Nonsmooth mechanics*, vol. 3. Springer, 1999.

[127] E. Drumwright and D. Shell, “Modeling contact friction and joint friction in dynamic robotic simulation using the principle of maximum dissipation,” in *Algorithmic foundations of robotics IX*, pp. 249–266, Springer, 2010.

[128] M. Anitescu and F. A. Potra, “Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems,” *Nonlinear Dynamics*, vol. 14, no. 3, pp. 231–247, 1997.

[129] S. C. Billups and K. G. Murty, “Complementarity problems,” *Journal of Computational and Applied Mathematics*, vol. 124, no. 1, pp. 303–318, 2000. Numerical Analysis 2000. Vol. IV: Optimization and Nonlinear Equations.

[130] W. Heemels, J. M. Schumacher, and S. Weiland, “Linear complementarity systems,” *SIAM journal on applied mathematics*, vol. 60, no. 4, pp. 1234–1269, 2000.

[131] Y. Shirai, D. K. Jha, A. U. Raghunathan, and D. Romeres, “Robust pivoting: Exploiting frictional stability using bilevel optimization,” in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 992–998, IEEE, 2022.

[132] S. Jin, D. Romeres, A. Ragunathan, D. K. Jha, and M. Tomizuka, “Trajectory optimization for manipulation of deformable objects: Assembly of belt drive units,” *arXiv preprint arXiv:2106.00898*, 2021.

[133] Y. Shirai, D. K. Jha, and A. U. Raghunathan, “Robust pivoting manipulation using contact implicit bilevel optimization,” *arXiv preprint arXiv:2303.08965*, 2023.

[134] Y. Shirai, D. K. Jha, A. U. Raghunathan, and D. Hong, “Tactile tool manipulation,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12597–12603, 2023.

[135] Y. Shirai, D. K. Jha, A. Raghunathan, and D. Hong, “Closed-loop tactile controller for tool manipulation,” in *ICRA 2023 Workshop on Embracing contacts. Making robots physically interact with our world*, 2023.

[136] M. K. Camlibel, J.-S. Pang, and J. Shen, “Lyapunov stability of complementarity and extended systems,” *SIAM J Optimization*, vol. 17, no. 4, pp. 1056–1101, 2006.

[137] A. Aydinoglu, P. Sieg, V. M. Preciado, and M. Posa, “Stabilization of complementarity systems via contact-aware controllers,” *IEEE Transactions on Robotics*, vol. 38, no. 3, pp. 1735–1754, 2021.

[138] A. U. Raghunathan and J. L. Linderoth, “Stability analysis of discrete-time linear complementarity systems,” *arXiv*, 2020.

[139] R. Cottle, J. Pang, and R. Stone, *The Linear Complementarity Problem.* Classics in Applied Mathematics, Society for Industrial and Applied Mathematics, 2009.

[140] A. Schperberg, S. Tsuei, S. Soatto, and D. Hong, “Saber: Data-driven motion planner for autonomously navigating heterogeneous robots,” *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 8086–8093, 2021.

[141] K. Okamoto and P. Tsiotras, “Optimal stochastic vehicle path planning using covariance steering,” *IEEE Robot. Autom. Lett.*, vol. 4, no. 3, pp. 2276–2281, 2019.

[142] O. Celik, H. Abdulsamad, and J. Peters, “Chance-constrained trajectory optimization for non-linear systems with unknown stochastic dynamics,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6828–6833, 2019.

[143] F. R. Hogan and A. Rodriguez, “Reactive planar non-prehensile manipulation with hybrid model predictive control,” *Int. J. Rob. Res.*, vol. 39, no. 7, pp. 755–773, 2020.

[144] J. J. Moré, B. S. Garbow, and K. E. Hillstrom, “User guide for minpack-1,” tech. rep., CM-P00068642, 1980.

[145] L. Blackmore, M. Ono, and B. C. Williams, “Chance-constrained optimal path planning with obstacles,” *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.

[146] S. Goyal, A. Ruina, and J. Papadopoulos, “Planar sliding with dry friction part 1. limit surface and moment function,” *Wear*, vol. 143, no. 2, pp. 307–330, 1991.

[147] Y. Shirai, D. K. Jha, A. U. Raghunathan, and D. Romeres, “Chance-constrained optimization in contact-rich systems,” in *2023 American Control Conference (ACC)*, pp. 14–21, 2023.

[148] M. Vidyasagar, *Nonlinear systems analysis.* SIAM, 2002.

[149] YALMIP, “Equalities with uncertainty,” *https://yalmip.github.io/equalityinuncertainty*, 2018.

[150] A. HSL, “collection of fortran codes for large-scale scientific computation,” *See http://www. hsl. rl. ac. uk*, 2007.

[151] J. K. Salisbury, “Active stiffness control of a manipulator in cartesian coordinates,” in *1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pp. 95–100, 1980.

[152] D. K. Jha, D. Romeres, W. Yerazunis, and D. Nikovski, “Imitation and supervised learning of compliance for robotic assembly,” in *2022 European Control Conference (ECC)*, pp. 1882–1889, 2022.

[153] D. K. Jha, D. Romeres, S. Jain, W. Yerazunis, and D. Nikovski, "Design of adaptive compliance controllers for safe robotic assembly," *arXiv preprint arXiv:2204.10447*, 2022.

[154] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3400–3407, 2011.

[155] M. Ono and B. C. Williams, "Iterative risk allocation: A new approach to robust Model Predictive Control with a joint chance constraint," *Proceedings of the IEEE Conference on Decision and Control*, no. 6, pp. 3427–3432, 2008.

[156] L. Blackmore, M. Ono, A. Bektassov, and B. C. Williams, "A probabilistic particle-control approximation of chance-constrained stochastic predictive control," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 502–517, 2010.

[157] T. Lew, R. Bonalli, and M. Pavone, "Chance-constrained sequential convex programming for robust trajectory optimization," in *2020 European Control Conference (ECC)*, pp. 1871–1878, 2020.

[158] Y. K. Nakka and S.-J. Chung, "Trajectory optimization of chance-constrained nonlinear stochastic systems for motion planning and control," *arXiv preprint arXiv:2106.02801*, 2021.

[159] J. Luedtke and S. Ahmed, "A sample approximation approach for optimization with probabilistic constraints," *SIAM Journal on Optimization*, vol. 19, no. 2, pp. 674–699, 2008.

[160] B. K. Pagnoncelli, S. Ahmed, and A. Shapiro, "Sample average approximation method for chance constrained programming: theory and applications," *Journal of optimization theory and applications*, vol. 142, no. 2, pp. 399–416, 2009.

[161] N. Jorge and J. W. Stephen, "Numerical optimization," 2006.

[162] Y. Shirai, D. K. Jha, A. Raghunathan, and D. Romeres, "Chance-constrained optimization in contact-rich systems for robust manipulation," *arXiv preprint arXiv:2203.02616*, 2022.

[163] Y. Shirai, X. Lin, A. Schperberg, Y. Tanaka, H. Kato, V. Vichathorn, and D. Hong, "Simultaneous contact-rich grasping and locomotion via distributed optimization enabling free-climbing for multi-limbed robots," in *Proc. 2022 IEEE/RSJ Int. Conf. Intell. Rob. Syst.*, pp. 13563–13570, 2022.

[164] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, 2019.

[165] Y. Shirai, D. Jha, and A. Raghunathan, "Contact-aware covariance control of stochastic contact-rich systems," in *IROS 2023 Workshop on Leveraging Models for Contact-Rich Manipulation*, 2023.

[166] T. Marcucci, R. Deits, M. Gabiccini, A. Bicchi, and R. 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)*, pp. 31–38, 2017.

[167] M. Erdmann, “On a representation of friction in configuration space,” *Int. J. Rob. Res.*, vol. 13, no. 3, pp. 240–271, 1994.

[168] N. Xydas and I. Kao, “Modeling of contact mechanics and friction limit surfaces for soft fingers in robotics, with experimental results,” *Int. J. Robo. Res.*, vol. 18, no. 9, pp. 941–950, 1999.

[169] I. H. Taylor, S. Dong, and A. 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)*, pp. 10781–10787, 2022.

[170] N. Hogan, “Impedance Control: An Approach to Manipulation: Part II—Implementation,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, pp. 8–16, 03 1985.

[171] W. Langson, I. Chryssochoos, S. Raković, and D. Q. Mayne, “Robust model predictive control using tubes,” *Automatica*, vol. 40, no. 1, pp. 125–133, 2004.

[172] K. M. Lynch and M. T. Mason, “Dynamic nonprehensile manipulation: Controllability, planning, and experiments,” *The International Journal of Robotics Research*, vol. 18, no. 1, pp. 64–92, 1999.

[173] A. Bicchi, “Hands for dexterous manipulation and robust grasping: a difficult road toward simplicity,” *IEEE Transactions on Robotics and Automation*, vol. 16, no. 6, pp. 652–662, 2000.

[174] N. Brook, M. Shoham, and J. Dayan, “Controllability of grasps and manipulations in multi-fingered hands,” *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, pp. 185–192, 1998.

[175] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, “Learning robust perceptive locomotion for quadrupedal robots in the wild,” *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[176] Z. Fu, X. Cheng, and D. Pathak, “Deep whole-body control: learning a unified policy for manipulation and locomotion,” in *Conference on Robot Learning*, pp. 138–149, PMLR, 2023.