Framework and Software for Real-Time Multi-Contact Model Predictive Control

Alp Aydinoglu, Adam Wei, Michael Posa

***Abstract*—Our recent work on consensus complementairy control (C3) proposes a model predictive control algorithm for hybrid systems that make and break contact with their environment [1]. C3 is based on the alternating direction method of multipliers (ADMM), that is capable of high-speed reasoning over potential contact events. Via a consensus formulation, the approach enables parallelization of the contact scheduling problem and is able to run at real-time rates. In this abstract, we build upon this with software improvements that increase the run-time by approximately 2 times over our previous implementation (achieving 80 Hz rate for a problem with 19 states, 12 complementarity variables and 0.5 seconds prediction horizon). Additionally we integrated our software with Drake [4]. The software parses models in URDF format into a non-smooth linear complementarity system and generates a controller that solves the optimal control problem. In addition to our previous work, we also validated our approach on a 3D manipulation example and are working towards validating it on a hardware setup with a robotic arm.**

I. Introduction

For many important tasks such as manipulation and locomotion, robots need to make and break contact with the environment. Even though such multi-contact systems are common, they are notoriously hard to control. The main challenge is finding policies and/or trajectories that explicitly consider the interaction of the robot with its environment in order to enable stable, robust motion. For a wide range of problems, it is computationally challenging to discover control policies and/or trajectories and the methods are not suitable for running in real-time speeds for complex problems.

Model predictive control (MPC) is one of the most powerful tools for automatic control. Predominant in robotics are MPC-based methods utilizing linearization, leading to quadratic programs which can be solved efficiently. However, for multi-contact systems, the algorithm must also decide when to initiate or break contact. As a result, linearizations are no longer appropriate and a hybrid formulation is required.

When linearizations are not viable, the resulting MPC algorithm includes the hybrid elements that result from making and breaking of contact. Because of this, developing MPC based control techniques that can reason about contact events and that do not require domain knowledge is a hard task. Recently, contact-implicit control with a primal-dual interior-point method has been explored with impressive results on simulation [2].

The primary contribution of our work is an algorithm, consensus complementarity control (C3), for solving the hybrid MPC problem approximately for multi-contact systems. We exploit the distributed nature of ADMM and demonstrate that the hard part of the problem, reasoning about contact events, can be parallelized. This enables our algorithm to be fast, robust to disturbances and also minimizes the effect of control horizon on the run-time of the algorithm. Since the original publication, we have rewritten the software library in C++ and improved the run-time ($\sim$ 2 x faster), now achieving 80 Hz rate for an example with 19 states and 12 complementarity variables. Additionally, we integrated our software with Drake and only require URDF models to run the algorithm. We also demonstrate the effectiveness of our method on (including new 3D manipulation) simulated examples as well as on a physical example. To the best of our knowledge, these are the first real-time control results on an underactuated system as complex and dynamic as the hybrid cart-pole and we are working towards validating our example on a 3D manipulation example with a robotic arm.

II. Methods

A standard approach to modeling robotic systems is through the framework of rigid-body systems with contacts. The continuous time dynamics can be modeled by manipulator equations

$$M(q)\dot{v} + C(q,v) = Bu + J(q)^T\lambda, \tag{1}$$

where q, v, λ, u represent the generalized coordinates, generalized velocities, the contact forces and the input respectively. Linear complementarity systems provide non-smooth, local approximations to (1):

$$\begin{aligned} &x_{k+1} = Ax_k + Bu_k + D\lambda_k + d, \\ &0 \leq \lambda_k \perp Ex_k + F\lambda_k + Hu_k + c \geq 0, \end{aligned} \tag{2}$$

where the orthogonality constraint embeds the hybridness.

A. Problem Formulation

In this work, we want to solve the following mathematical optimization problem:

$$\begin{aligned} \min_{x_k,\lambda_k,u_k} \quad & \sum_{k=0}^{N-1}(x_k^TQ_kx_k + u_k^TR_ku_k) + x_N^TQ_Nx_N \\ \text{s.t.} \quad & x_{k+1} = Ax_k + Bu_k + D\lambda_k + d, \\ & 0 \leq \lambda_k \perp Ex_k + F\lambda_k + Hu_k + c \geq 0, \\ & (x_k, \lambda_k, u_k) \in \mathcal{C}, \end{aligned} \tag{3}$$

where N is the planning horizon, Q_k, Q_N are positive semidefinite matrices, R_k are positive definite matrices and $\mathcal{C}$ is a convex set.

Given x_0, one solves the optimization (3) and applies u_0 to the plant and repeats the process in every time step in a receding horizon manner.

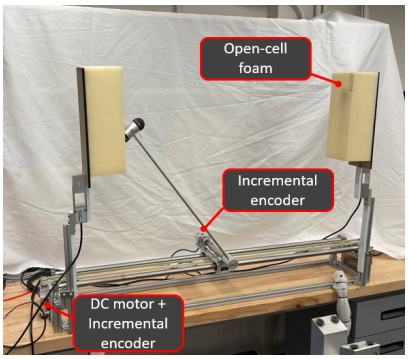

Fig. 1. Experimental setup for cart-pole with soft walls.

B. Consensus Complementarity Control

Our approach consists of solving quadratic programs (QP) followed by low-dimensional projections onto the complementarity constraints (MIQP or heuristics), details can be found in [1]. The QP can be solved quickly via off-the-shelf solvers and is analogous to solving the MPC problem for a linear system without contact. Projection step requires projecting onto the complementarity constraints and is the most challenging part (contact scheduling). Regardless, ADMM approach allows us to split contact scheduling into N independent sub-problems and leads to dramatically increased performance by minimizing the effect of control horizon on the run-time.

III. Software Details

The code[1] is written in C++ with a Python interface. Software consists of three main components: LCS generation, main solver operations (including QP) and projection.

Using a URDF model and contact pairs, the software can generate an LCS model (2) for any given state. The LCS model is based on a implicit time-stepping scheme [3] that replaces the friction cone with a polyhedral approximation. The software allows changing the number edges of the polyhedral approximation that introduces a trade-off between model accuracy and model complexity.

Given an LCS object and costs, the C3 (solver) object can be defined. We use OSQP for solving the QP's and users can easily add constraints after creating an object of the C3 class.

We solve the projection problem via MIQP projection using Gurobi.

IV. Results/Discussion

Previously [1], we validated our results on three numerical examples, including two frictional contact problems, and physical experimentation on an underactuated multi-contact system as in Figure 1. We provide the run-time details in Table I.

We are currently validating our approach on the manipulation task shown in Figure 2. Our goal is to roll a rigid sphere along a given trajectory using the Franka Emika Panda arm. C3 finds strategies, such as pushing from the side or rolling, to manipulate the sphere while the low-level controller computes torques for the Panda arm to track the desired strategies.

[1]https://github.com/DAIRLab/dairlib/blob/admm/systems/controllers/c3_controller.h

TABLE I
Run-time Rate / Number of ADMM Iterations / State Dimension / Complementarity Variable Dimension/ Input Dimension/ Horizon

Example	Freq.	Iter.	x	λ	u	N
Finger Gaiting	20 Hz (10 Hz before)	10	6	6	4	10 (1 sec)
Pivoting	27 Hz (16 Hz before)	5	10	10	4	10 (1 sec)
3D Manipulation	80 (25 Hz before)	3	19	12	3	5 (0.5 sec)

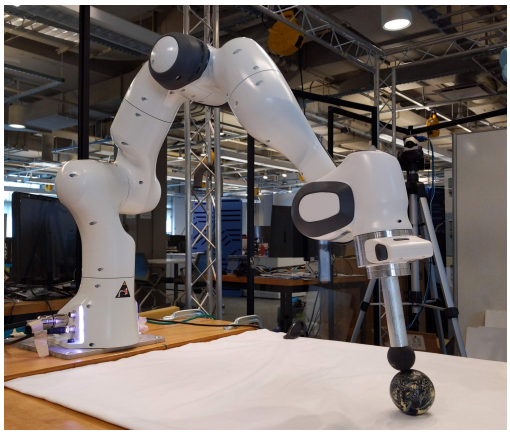

Fig. 2. Manipulating a rigid object (sphere) with a spherical end effector attached to the Franka Emika Panda arm.

Approximately 70 percent of the solve time is spent on the projection step and almost the majority of the remaining solve time is spent on the QP. We are implementing warm-start procedures to speed up the QP part, and also exploring heuristics for the projection step. Our goal is to achieve fast, dynamic rolling motions that follow a path, including verification on a experimental setup.

Integration with learned models is in the scope of future work with few successful preliminary trials on the cart-pole experimental setup.

References

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

[2] Simon Le Cleac'h, Taylor Howell, Mac Schwager, and Zachary Manchester. Linear contact-implicit model-predictive control. *arXiv preprint arXiv:2107.05616*, 2021.

[3] David Stewart and Jeffrey C Trinkle. An implicit time-stepping scheme for rigid body dynamics with coulomb friction. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 162–169. IEEE, 2000.

[4] Russ Tedrake and the Drake Development Team. Drake: Model-based design and verification for robotics, 2019. URL https://drake.mit.edu.