MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Python-based Open Source Package for Optimization of Contact-rich Systems

Raghunathan, Arvind; Jha, Devesh K.; Romeres, Diego

TR2022-089 August 06, 2022

Abstract

PYROBOCOP is a Python-based package for control, optimization and estimation of robotic systems described by nonlinear Differential Algebraic Equations. In particular, the package can handle systems with contacts that are described by complementarity constraints and provides a general framework for specifying obstacle avoidance constraints. PYROBOCOP provides automatic reformulation of the complementarity constraints to perform optimization of robotic systems. The package is interfaced with ADOL-C [1] for automatic differentiation with sparse derivatives and IPOPT [5] as solver.

Robotics: Science and Systems 2022

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Python-based Open Source Package for Optimization of Contact-rich Systems

Arvind U. Raghunathan[1], Devesh K. Jha[1]and Diego Romeres[1]

***Abstract*—PyRoboCOP is a Python-based package for control, optimization and estimation of robotic systems described by nonlinear Differential Algebraic Equations. In particular, the package can handle systems with contacts that are described by complementarity constraints and provides a general framework for specifying obstacle avoidance constraints. PyRoboCOP provides automatic reformulation of the complementarity constraints to perform optimization of robotic systems. The package is interfaced with ADOL-C [1] for automatic differentiation with sparse derivatives and IPOPT [5] as solver.**

I. Introduction

We present a python-based robotic control and optimization package (called PyRoboCOP) that allows solution to a large class of mathematical programs with nonlinear and complementarity constraints. Contact-rich robotic manipulation tasks could be modeled as complementarity systems. Obtaining a feasible, let alone an optimal trajectory, can be challenging for such systems. To the best of our knowledge, none of the existing python-based open-source optimization packages can provide support for trajectory optimization with support for complementarity constraints that arise from contact-rich manipulation and simple specification of obstacle avoidance constraints. However, this is highly desirable to optimize a large-class of contact-rich robotic systems.

We present PyRoboCOP , a lightweight but powerful Python-based package for control and optimization of robotic systems. PyRoboCOP handles contact and collision avoidance in an unified manner and uses ADOL-C [1] and IPOPT [5] at its backend for automatic differentiation and optimization, respectively. The main features of PyRoboCOP are:

- Contact and obstacle avoidance modeling by complementarity constraints
- Support for minimum time problems
- Support for optimization over fixed mode sequence problems with unknown sequence time horizons
- Parameter estimation in linear complementarity systems.

Codes and instructions for installing and using PyRoboCOP could be found in [4] under a research only license.

II. Problem Description

PyRoboCOP solves the dynamic optimization problem

$$\min_{x,y,u,p} \int_{t_0}^{t_f} c(x(t), y(t), u(t), p)dt + \phi(x(t_f), p) \tag{1a}$$

$$\text{s.t. } f(\dot{x}(t), x(t), y(t), u(t), p) = 0,\ x(t_0) = \widehat{x} \tag{1b}$$

$$([y(t)]_{\sigma_{l,1}} - \nu_{l,1})([y(t)]_{\sigma_{l,2}} - \nu_{l,2}) = 0 \,\forall\, l \in \mathcal{L} \tag{1c}$$

$$\underline{x} \le x(t) \le \overline{x}, \underline{y} \le y(t) \le \overline{y}, \underline{u} \le u(t) \le \overline{u} \tag{1d}$$

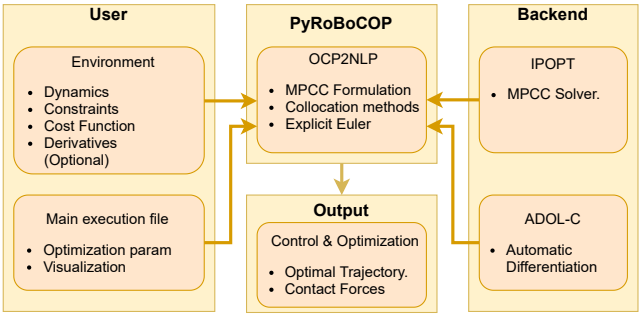


Fig. 1: Workflow in PyRoboCOP. The dynamics provided by the user create a MPCC which is optimized using IPOPT. The gradients are evaluated using automatic differentiation.

where $x(t) \in \mathbb{R}^{n_x}$, $y(t) \in \mathbb{R}^{n_y}$, $u(t) \in \mathbb{R}^{n_u}$, $\dot{x}(t) \in \mathbb{R}^{n_x}$, $p \in \mathbb{R}^{n_p}$ are the differential, algebraic, control, time derivative of differential variables and time-invariant parameters respectively. Each $l \in \mathcal{L}$ defines a pair of indices $\sigma_{l,1}, \sigma_{l,2} \in \{1, \ldots, n_y\}$ that specifies the complementarity constraint between the algebraic variables $[y(t)]_{\sigma_{l,1}}$ and $[y(t)]_{\sigma_{l,2}}$. In (1c) $\nu_{l,1}, \nu_{l,2}$ correspond to either the lower or upper bounds on the corresponding algebraic variables. For example, if they are set respectively to the lower and upper bounds of corresponding algebraic variables then (1c) in combination with the bounds (1d) model the complementarity constraint

$$0 \le [y(t) - \underline{y}]_{\sigma_{l,1}} \perp [\overline{y} - y(t)]_{\sigma_{l,2}} \ge 0.$$

The dynamic optimization problem in (1) is transcripted to a NonLinear Program (NLP) using the Implicit Euler time-stepping scheme. If complementarity constraints are present then it is an instance of a Mathematical Program with Complementarity Constraints. PyRoboCOP implements two possible relaxation schemes for complementarity constraints

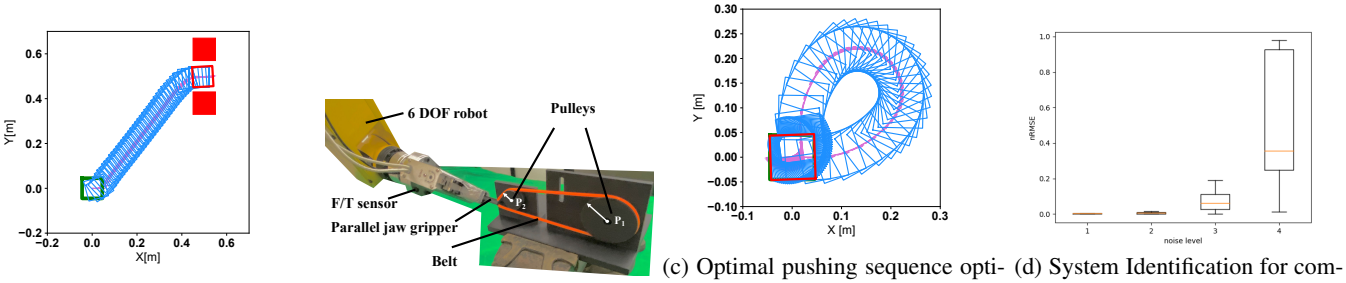
$$\alpha_l([y_i]_{\sigma_{l,1}} - \nu_{l,1})([y_i]_{\sigma_{l,2}} - \nu_{l,2}) \le \delta \,\forall\, l \in \mathcal{L} \tag{2a}$$

$$\sum_{l \in \mathcal{L}} \alpha_l([y_i]_{\sigma_{l,1}} - \nu_{l,1})([y_i]_{\sigma_{l,2}} - \nu_{l,2}) \le \delta \tag{2b}$$

where $\alpha_l = 1$ if the involved bounds $(\nu_{l,1}, \nu_{l,2})$ are either both lower or both upper bounds. More details could be found in the extended version of the paper in [3].

III. Software Description

Figure 1 provides a high-level summary of the control flow in PyRoboCOP. Detailed descriptions on the software API and classes are available in the Software Description in [4]. A user provided class specifies the dynamic optimization problem (1). This is also briefly described in Figure 1. The user needs to provide the equality constraints for the dynamical



(a) Pushing with obstacles (b) Belt Drive Unit (c) Optimal pushing sequence optimizing a mode sequence. (d) System Identification for complementarity systems

Fig. 2: Examples of contact-rich and parameter estimation problems solved with PyRoboCOP .

system. These constraints could include the dynamics information for the system, the bounds on the system state and inputs, and information about complementarity constraints, if any. Furthermore, a user needs to provide the objective function. PyRoboCOP is interfaced with ADOL-C [1] to compute derivatives (see the Backend block in Figure 1). Note that the ADOL-C can also provide the sparsity pattern of the constraint Jacobian and Hessian of the Lagrangian.

IV. Numerical Results

PyRoboCOP is validated in several robotic simulations and real world experiments. To foster reproducibility, the source code of the following examples is available in [4] and more details about the experiments are available in [3].

Benchmark systems. First, PyRoboCOP was tested in many of the standard control benchmarks with and without contact interactions e.g., the inverted pendulum, the acrobat system and the cart pole with softwalls.

Top-table manipulation. A second class of problems that was solved with PyRoboCOP is the pushing objects in a plane class. Consider a pusher-slider system, the pusher interacts with the slider by exerting forces in the normal and tangential direction as well as a torque about the center of the mass of the object. The applied wrench causes the object to move in a perpendicular direction to the limit surface. Moreover, consider the case when the pusher has to avoid obstacles in order to push the slider in a determined goal position as shown in Fig. 2(a). PyRoboCOP can handle complementarity and obstacle avoidance constraints simultaneously.

Experiments on physical devices. Third, PyRoboCOP solved the trajectory optimization problem of a Belt Drive Unit as described in our previous paper [2]. The goal was to wrap with a robotic manipulator a flexible belt around two pulleys, see Fig. 2(b). This is a complex manipulation problem that involves contact, flexible objects and collision avoidance.

Optimization with Mode Enumeration. Fourth, we show our optimization approach over fixed mode sequences using the pusher-slider system. When considering sticking contact at the 4 faces of the slider, the modes appear based on which face the pusher contacts the slider. The optimization process ensures continuity of dynamics and selection of final time for each mode in a trajectory. For this example, we use two modes, pushing from the left face followed by pushing from the top face of the slider and the goal is to bring the slider to its original position rotate of $180°$. The optimal trajectory is shown in Fig. 2(c).

System Identification for Complementarity Systems. Fifth, the problem of parameter estimation for systems with complementarity constraints is a particular case of the NLP transcription of the optimization problem (1). The objective is to identify the physical parameters of a system given a set of collected data. As a case of study, we consider the cartpole with softwalls and the cost function is the normalized Root Mean Square Error between the observed trajectory and the trajectory obtained from the estimation procedure. We validated the method with a Monte Carlo simulation on different cartpole systems with increasing levels of additive independent Gaussian noise. The results are shown in Figure 2(d) with noise standard deviation for each experiment of $[0.0001, 0.001, 0.01, 0.05]$.

V. Concluding Remarks

This paper presented PyRoboCOP , a Python-based optimization package for model-based control of robotic systems. We showed that PyRoboCOP can be used to solve trajectory optimization problems of a number of dynamical systems in presence of contact and collision avoidance constraints. More details are available in the software package [4].

References

[1] Andreas Griewank, David Juedes, and Jean Utke. Algorithm 755: Adol-c: A package for the automatic differentiation of algorithms written in c/c++. *ACM Trans. Math. Softw.*, June 1996. doi: 10.1145/229473.229474.

[2] S. Jin, D. Romeres, A. Ragunathan, D. Jha, and M. Tomizuka. Trajectory optimization for manipulation of deformable objects: Assembly of belt drive units. In *IEEE ICRA*, 2021.

[3] A. U. Raghunathan, D. K. Jha, and D. Romeres. Pyrobocop: Python-based robotic control & optimization package for manipulation and collision avoidance, arXiv, 2021.

[4] Arvind U. Raghunathan, Devesh K. Jha, and Diego Romeres. Pyrobocop, 2021. URL https://www.merl.com/research/license/PyRoboCOP.

[5] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106:25–57, 2006. doi: https://doi.org/10.1007/s10107-004-0559-y.