Comparison of Optimal Control Software Frameworks in the Context of Bipedal Walking

Julian Eßer

February 19, 2020

Contents

Chapter 1

INTRODUCTION

Chapter 2

CROCODDYL - LAAS-CNRS

2.1 Introduction

2.1.1 Motivation

Crocoddyl is an **optimal control library for robot control under contact sequence**. Its solver is based on an efficient Differential Dynamic Programming (DDP) algorithm. Crocoddyl computes optimal trajectories along with optimal feedback gains. It uses Pinocchio for fast computation of robot dynamics and its analytical derivatives [1].

Crocoddyl is focused on multi-contact optimal control problem (MCOP) which as the form:

$$\mathbf{X}^*, \mathbf{U}^* = \begin{Bmatrix} \mathbf{x}_0^*, \cdots, \mathbf{x}_N^* \\ \mathbf{u}_0^*, \cdots, \mathbf{u}_N^* \end{Bmatrix} = \arg\min_{\mathbf{X},\mathbf{U}} \sum_{k=1}^{N} \int_{t_k}^{t_k+\Delta t} l(\mathbf{x}, \mathbf{u}) dt$$

subject to

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}),$$

$$\mathbf{x} \in \mathcal{X}, \mathbf{u} \in \mathcal{U}, \boldsymbol{\lambda} \in \mathcal{K}.$$

where

- the state $\mathbf{x} = (\mathbf{q}, \mathbf{v})$ lies in a manifold, e.g. Lie manifold $\mathbf{q} \in SE(3) \times \mathbb{R}^{n_j}$, n_j being the number of degrees of freedom of the robot.
- the system has underactuacted dynamics, i.e. $\mathbf{u} = (\mathbf{0}, \boldsymbol{\tau})$,
- $\mathcal{X}, \mathcal{U}$ are the state and control admissible sets, and
- $\mathcal{K}$ represents the contact constraints.

Note that $\boldsymbol{\lambda} = \mathbf{g}(\mathbf{x}, \mathbf{u})$ denotes the contact force, and is dependent on the state and control.

2.1.2 Features

According to the description in the Github repository [1], it comprises the following features:

Crocoddyl is **versatible**:

- various optimal control solvers (DDP, FDDP, BoxDDP, etc) - single and multi-shooting methods
- analytical and sparse derivatives via Pinocchio
- Euclidian and non-Euclidian geometry friendly via Pinocchio
- handle autonomous and nonautomous dynamical systems
- numerical differentiation support

Crocoddyl is **efficient** and **flexible**:

- cache friendly,
- multi-thread friendly
- Python bindings (including models and solvers abstractions)
- C++ 98/11/14/17/20 compliant
- extensively tested

2.2 How-To

2.2.1 Install

2.2.1.1 Two ways to go

Basically there are existing two ways of installing Crocoddyl:

- Option 1: Installation via the *robotpkg* package manager
- Option 2: Installation from source

I personally would recommend the installation through *robotpkg*, since it preserves you from dealing with the multiple dependencies of Crocoddyl and therefore seems to be the faster approach. Generally you should decide beforehand which python version you want to use. This effects the robotpkg version as well as the export of the PYTHONPATH variable.

2.2.1.2 Installation via robotpkg (preferred)

Steps for installing via robotpkg according to the installation section of [1]

1. Add robotpkg as source repository to apt:

   ```
   sudo tee /etc/apt/sources.list.d/robotpkg.list <<EOF
   deb [arch=amd64] http://robotpkg.openrobots.org/wip/packages/debian/pub $(lsb_release -sc) robotpkg
   deb [arch=amd64] http://robotpkg.openrobots.org/packages/debian/pub $(lsb_release -sc) robotpkg
   EOF
   ```

2. Register the authentication certificate of robotpkg:

   ```
   curl http://robotpkg.openrobots.org/packages/debian/robotpkg.key | sudo apt-key add -
   ```

3. You need to run at least once apt update to fetch the package descriptions:

   ```
   sudo apt-get update
   ```

4. The installation of Crocoddyl:

   ```
   sudo apt install robotpkg-py27-crocoddyl # for Python 2
   sudo apt install robotpkg-py36-crocoddyl # for Python 3
   ```

5. Finally you will need to configure your environment variables, e.g.:

   ```
   export PATH=/opt/openrobots/bin:$PATH
   export PKG_CONFIG_PATH=/opt/openrobots/lib/pkgconfig:$PKG_CONFIG_PATH
   export LD_LIBRARY_PATH=/opt/openrobots/lib:$LD_LIBRARY_PATH
   export PYTHONPATH=/opt/openrobots/lib/python2.7/site-packages:$PYTHONPATH
   ```

2.2.1.3 (Installation from source)

If you prefer installing Crocoddyl from source, the following steps should do the work:

```
git clone https://github.com/loco-3d/crocoddyl.git
git submodule update --init
mkdir build && cd build
export PKG_CONFIG_PATH=/opt/openrobots/lib/pkgconfig
cmake -DCMAKE_INSTALL_PREFIX=/opt/openrobots  ..
make
sudo make install
```

Additionally you will have to install the dependent libraries (i.e. pinocchio, example-robot-data (optional for examples, install Python loaders), gepetto-viewer-corba (optional for display), jupyter (optional for notebooks) and matplotlib (optional for examples) and fix the incude paths.

2.2.2 Running the Examples

Since the installation through robotpkg did not provide you with the examples from the git repository, you should clone the repo [1] for getting the data. You do not have to build the library, since it already is installed. In the cloned repository go to */examples*. For running e.g. the bipedal walking example, just type

```
python3 bipedal_walk.py
```

and you will see the calculations for optimal gait trajectories running in the console. You propably want to view your results now. For displaying the results, we need to install the gepetto-viewer:

```
 sudo apt install robotpkg-py36-qt4-gepetto-viewer-corba
```

The examples provide a *plot* and *display* argument. In order to display the 3D results and also plot some data, just do

```
gepetto-gui
```

for starting the 3D environment and then, in another terminal

```
python3 bipedal_walk.py display plot
```

2.3 Abstract Workflow

2.4 Detailed Example

2.5 Background

Chapter 3

DRAKE - MIT CSAIL

Chapter 4

CONTROL TOOLBOX - ETH Zurich

Chapter 5

COMPARISON

Chapter 6

CONCLUSION AND OUTLOOK

Bibliography

[1] Rohan Budhiraja Carlos Mastalli, Nicolas Mansard, et al. Crocoddyl: a fast and flexible optimal control library for robot control under contact sequence. https://gepgitlab.laas.fr/loco-3d/crocoddyl/wikis/home, 2019.