FCCQP - A Whole Body Control QP Solver with Full Friction Cones

February 23, 2024

1 Introduction

Optimization based control architectures for dynamic legged robots have converged to the "QP approach" of formulating reactive controllers as quadratic programs (QPs). The decision variables in these QPs include the generalized accelerations, inputs, contact forces, and other constraint forces of a Lagrangian dynamics model of the robot [6]. The costs encode information such as desired task-space accelerations or contact forces. Often, the only inequality constraints in the QP are input limits and friction cone constraints, leading to QPs of the form (1).

$$\underset{x}{\text{minimize}} \ \frac{1}{2}x^T Q x + b^T x + c \tag{1a}$$
$$\text{subject to } A_{eq}x = b_{eq} \tag{1b}$$
$$\lambda_c \in \mathcal{F} \tag{1c}$$
$$b_l \leq z \leq b_u \tag{1d}$$

Here, $x = \begin{bmatrix} \dot{v}^T & u^T & \lambda_h^T & \lambda_c^T \end{bmatrix}^T$ is a vector of the stacked generalized accelerations, inputs, holonomic constraint forces, and contact forces, and $z = \begin{bmatrix} \dot{v}^T & u^T & \lambda_h^T \end{bmatrix}^T$ contains all variables except the contact forces. A_{eq} and b_{eq} specify dynamics constraints, holonomic constraints (such as loop closures and fixed joints), and contact constraints. $\mathcal{F}$ is the set of friction cones, and b_l and b_u are bounds on the decision variables[1].

Usually, (1) is solved with general purpose QP solvers, requiring the friction cone constraints to be approximated by pyramidal linear constraints, and ignoring advantageous problem structure.

We notice that both the bounds on z and the Lorentz cone constraint defining the full friction cone have feasible sets which are, informally, "easy to project

[1] Usually all variables except u are unbounded by b_l and b_u, and contact forces are bounded only by friction cones

to", and decoupled from other inequality constraints. This structure allows a vanilla implementation of the Alternating Direction Method of Multipliers (ADMM) to solve (1) quickly and robustly. We introduce FCCQP (Friction Cone Constrained QP), a QP solver specifically for solving convex Lorentz-Cone-Constrained QPs of the form (1) using ADMM.

2 ADMM For Convex Optimization [2]

Before formulating our solver, we will briefly review the recipe for solving a convex optimization problem with ADMM. The generic convex optimization problem

$$\underset{x \in \mathcal{C}}{\text{minimize}} \; f(x) \tag{2}$$

can be transformed into an equivalent problem amenable to ADMM by introducing a slack variable, y:

$$\underset{x,y}{\text{minimize}} \; f(x) + I_{\mathcal{C}}(y) \tag{3a}$$

$$\text{subject to } x = y \tag{3b}$$

where $I_{\mathcal{C}}(y)$ is the indicator function

$$I_{\mathcal{C}}(y) = \begin{cases} 0, & y \in \mathcal{C}, \\ \infty, & y \notin \mathcal{C}. \end{cases} \tag{4}$$

The scaled form of the ADMM iterations are then given by:

$$x_{k+1} = \arg\min_{x} \left(f(x) + \frac{\rho}{2} \left\| x - y_k + w_k \right\|_2^2 \right), \tag{5a}$$

$$y_{k+1} = \mathcal{P}_{\mathcal{C}} \left(x_{k+1} + w_k \right) \tag{5b}$$

$$w_{k+1} = w_k + x_{k+1} - y_{k+1}. \tag{5c}$$

Where $\mathcal{P}_{\mathcal{C}}(v)$ is the projection of v onto $\mathcal{C}$, (5a) is the primal update, (5b) is the slack update, and (5c) is the dual update.

3 Solving (1) with ADMM

We solve (1) via ADMM by splitting it into an equality constrained QP (the primal update) and independent projections onto the variable bounds and friction cone constraints (the slack update). More explicitly,

$$
\begin{aligned}
x_{k+1} = \arg\min_{x} \quad & \frac{1}{2} x^T Q x + b^T x + \frac{\rho}{2} \left\| x - \overline{x}_k + w_k \right\|_2^2 && \text{(6a)} \\
\text{subject to} \quad & A_{eq} x = b_{eq} \\
\overline{\lambda}_{c,k+1} = \mathcal{P}_{\mathcal{F}}(\lambda_{c,k+1} + w_{\lambda_{c,k}}) & && \text{(6b)} \\
\overline{z}_{k+1} = \mathcal{P}_{bounds}(z_{k+1} + w_{z,k}) & && \text{(6c)} \\
w_{k+1} = w_k + x_{k+1} - \overline{x}_{k+1}. & && \text{(6d)}
\end{aligned}
$$

Where $\overline{z}$ and $\overline{\lambda}_c$ are slack variables for z and λ_c, $\overline{x} = \begin{bmatrix} \overline{z}^T & \overline{\lambda}_c^T \end{bmatrix}^T$, and $w = \begin{bmatrix} w_z^T & w_{\lambda_c}^T \end{bmatrix}^T$.

The primal update (6a) can be computed by solving the KKT system [3],

$$
\begin{bmatrix} \tilde{Q} & A_{eq}^T \\ A_{eq} & 0 \end{bmatrix} \begin{bmatrix} x \\ \nu \end{bmatrix} = \begin{bmatrix} -\tilde{b} \\ b_{eq} \end{bmatrix}, \tag{7}
$$

where $\tilde{Q} = Q + \rho I$ and $\tilde{b} = b - \rho(\overline{x}_k - w_k)$.

The slack update projections are handled individually for each contact force in λ_c, and element-wise for the box constraint on z. The projection to a friction cone with coefficient of friction μ is given by

$$
\mathcal{P}_{\mathcal{F}_\mu}(\lambda) = \begin{cases} \lambda, & \mu\lambda_z \geq \|\lambda_{xy}\| \\ 0, & \lambda_z < -\mu\|\lambda_{xy}\| \\ \frac{\lambda^T v}{\|v\|^2} v \mid v = (\frac{\mu}{\|\lambda_{xy}\|}\lambda_x, \frac{\mu}{\|\lambda_{xy}\|}\lambda_y, 1), & \text{otherwise} \end{cases} \tag{8}
$$

and the box constraint projection is given by

$$
\mathcal{P}_{bounds}(v) = \min(\max(v, b_l), b_u) \tag{9}
$$

where the min and max are computed element-wise. We initialize x and $\overline{x}$ to the solution of (1) with only equality constraints, resulting in algorithm 1.

Algorithm 1 FCCQP ADMM Implementation

Input: Problem Data $Q, b, A_{eq}, b_{eq}, \mathcal{F}, b_l, b_u$, Hyper-parameters m, ϵ, ρ
Initialization: $x_0 = \overline{x}_0 = \arg\min_x \ x^T Q x + b^T x$ s.t. $A_{eq} x = b_{eq}$.
for $k = 0 \dots m$ **do**
  Update x_{k+1} by solving (7)
  Update $\overline{\lambda}_{c,k+1}, \overline{z}_{k+1}$ with (6b), (6c)
  Update duals (6d)
  if $\|x_{k+1} - \overline{x}_{k+1}\| < \epsilon$ **then**
    break
  end if
end for
Output: Final solution x^*.

4 Operational Space Control Example

In this section, we use FCCQP to solve an Operational Space Control QP for the underactuated biped Cassie [1] [7]. In this case, the operational space commands are to realize a basic linear-inverted pendulum style walking controller, similar to [4]. Experiments are conducted on hardware.

For task space PD commands $\ddot{y}_{cmd} = \ddot{y}_{des} + K_p(y_{des} - y) + K_d(\dot{y}_{des} - \dot{y})$, the goal of OSC is to find feasible inputs, u, such that the tasks space accelerations $\ddot{y} = J_y \dot{v} + \dot{J}_y v$ match the commands as closely as possible. This yields a QP in the form (1),

$$\underset{\dot{v}, u, \lambda_h, \lambda_c}{\text{minimize}} \ \sum_i^N \widetilde{\ddot{y}}_i^T W_i \widetilde{\ddot{y}}_i + \|u\|_W^2 + \|\dot{v}\|_W^2 \tag{10a}$$

$$\text{subject to } M\dot{v} + C = Bu + J_h^T \lambda_h + J_c^T \lambda_c \tag{10b}$$

$$J_h \dot{v} = -\dot{J}_h v \tag{10c}$$

$$J_c \dot{v} = -\dot{J}_c v \tag{10d}$$

$$\lambda_c \in \mathcal{F} \tag{10e}$$

$$u_{min} \leq u \leq u_{max}. \tag{10f}$$

The holonomic constraint $J_h \dot{v} = -\dot{J}_h v$ represents Cassie's four-bar linkages and fixed joint constraints to model Cassie's leaf spring springs, and the task space acceleration errors are $\widetilde{\ddot{y}}_i = \ddot{y}_{cmd} - (J_{y,i}\dot{v} + \dot{J}_{y,i} v)$.

FCCQP's solve times compared to OSQP [5] can be seen in fig. 1. FCCQP is slightly slower than OSQP, but more consistent due to the ability to more easily tune stopping criteria (see discussion).

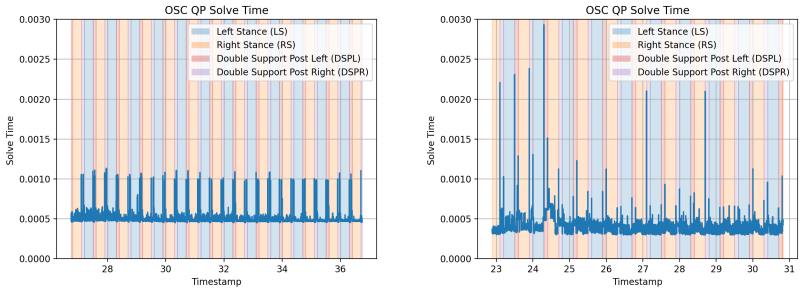

Figure 1: FCCQP (left) and OSQP (right) solve times during Casse walking experiments. Solve times spike during double-stance due to additional constraints.

5 Discussion and TODOs

5.1 Tune-ability of solve-time

Unlike OSQP, FCCQP does not use ADMM to enforce equality constraints, but instead solves an equality constrained QP at every primal iteration. Therefore, every intermediate solution x_k respects the dynamics and contact constraints. This allows for more trustworthy early termination in principle, but should be verified experimentally.

5.2 Linear Solving

The vast majority of the runtime of FCCQP lies in factorizing two different KKT matrices, one to solve the initial equality constrained QP, and one to solve the ADMM primal update problems. Once the KKT matrix (7) is factorized, that factorization can be re-used for the remaining ADMM iterations.

In practice, the KKT matrix is ill-conditioned and rank deficient, so a balance must be stuck between solver accuracy and speed. Additionally, we take the least-squares solution to (7) for numerical stability and consistency between solves. Of the solvers available in the Eigen Dense module, CompleteOrthogonalDecomposition is the fastest solver to yield acceptable accuracy. It is certainly possible that using a different linear solver could improve FCCQP's runtime.

References

[1] T. Apgar, P. Clary, K. Green, A. Fern, and J. Hurst. Fast Online Trajectory Optimization for the Bipedal Robot Cassie. In *Robotics: Science and Systems XIV*. Robotics: Science and Systems Foundation, June 2018.

[2] 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*, 3(1):1–122, 2011.

[3] S. P. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[4] Y. Gong and J. Grizzle. One-Step Ahead Prediction of Angular Momentum about the Contact Point for Control of Bipedal Locomotion: Validation in a LIP-inspired Controller. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2832–2838, May 2021.

[5] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020.

[6] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, and A. D. Prete. Optimization-Based Control for Dynamic Legged Robots. *IEEE Transactions on Robotics*, 40:43–63, 2024. Conference Name: IEEE Transactions on Robotics.

[7] W. Yang and M. Posa. Impact Invariant Control with Applications to Bipedal Locomotion. *arXiv:2103.06907 [cs]*, Mar. 2021. arXiv: 2103.06907.