ALTRO-C: A Fast Solver for Conic Model-Predictive Control

Brian E. Jackson
Robotics Institute
Carnegie Mellon University
Pittsburgh, USA
brianjackson@cmu.edu

Tarun Punnoose
Department of Electrical Engineering
Stanford University
Stanford, USA
punnoose@stanford.edu

Daniel Neamati
Department of Mechanical and Civil Engineering
California Institute of Technology
Pasadena, USA
dneamati@caltech.edu

Kevin Tracy
Robotics Institute
Carnegie Mellon University
Pittsburgh, USA
ktracy@cmu.edu

Rianna Jitosho
Department of Mechanical Engineering
Stanford University
Stanford, USA
rjitosho@stanford.edu

Zachary Manchester
Robotics Institute
Carnegie Mellon University
Pittsburgh, USA
zacm@cmu.edu

***Abstract*—Model-predictive control (MPC) is an increasingly popular method for controlling complex robotic systems in which optimal control problems are solved on board the robot at real-time rates. However, successful application of MPC depends critically on the performance of the algorithms used to solve the underlying optimization problems. An ideal solver should both leverage the structure of the MPC problem and support efficient "warm starting" so that information from previous solutions can be recycled to speed convergence. We present ALTRO-C, a high-performance solver with both of these properties that utilizes an augmented Lagrangian method to handle general convex conic constraints. We demonstrate the new solver's superior performance against several existing state-of-the-art solvers on a variety of benchmark control problems formulated as both quadratic and second-order cone programs.**

I. INTRODUCTION

Model-predictive control (MPC) has become a widely used approach for controlling complex robotic systems with several notable successes in recent years [1]–[3]. By transforming the control problem into an optimization problem with an explicit objective and constraints, MPC can achieve desired behaviors while accounting for complex dynamics, torque limits, and obstacle avoidance. However, because these optimization problems must typically be solved at rates of tens to hundreds of Hertz on board the robot, efficient and reliable solver algorithms are crucial to their success.

In practice, most MPC problems are formulated as convex optimization problems, since convex solvers are available that can guarantee convergence to a globally optimal solution, or provide a certificate of infeasibility if a solution does not exist. A variety of numerical techniques for solving such problems have been developed over the past several decades, and many high-quality solver implementations—both open-source and proprietary—are available. In the control community, a particular emphasis has been placed on high-performance solvers for quadratic programs (QPs) that are suited to real-time use on embedded computing hardware, including active-set [4], [5], interior-point [6]–[8], and alternating direction method of multipliers (ADMM) [9] methods. There has also been at least one interior-point solver for second-order cone programs (SOCPs) developed for embedded applications [10].

To achieve high performance on MPC problems, a solver must a) exploit problem structure with sparse matrix factorization techniques, b) take advantage of previous solutions to "warm start" the current solve, and c) efficiently handle convex conic constraints on states and inputs. We present ALTRO-C, a modified version of the ALTRO solver [11] originally developed for offline solution of nonlinear trajectory optimization problems, that achieves all three of these properties. As a result, it delivers state-of-the-art performance on both QPs and SOCPs in MPC applications, and can easily support optimization over other cones in the future. To the best of our knowledge, ALTRO-C is the first solver specifically designed for MPC applications that can handle second-order cone constraints. In summary, our contributions include:

- A novel method for incorporating conic—including second-order cone—constraints into a Differential Dynamic Programming (DDP)-based trajectory optimization solver.
- An open-source solver implementation in Julia that delivers state-of-the-art performance for convex MPC problems with a convenient interface for defining trajectory optimization problems.
- A suite of benchmark MPC problems that demonstrate the solver's performance, including a satellite with flexible appendages, a quadruped with both linearized and second-order friction cones, rocket soft-landing, and manipulation with contact.

The paper proceeds as follows: After defining our notation in Section II, we present our method for handling conic constraints with an augmented Lagrangian in Section III. Section IV summarizes the ALTRO-C algorithm and discusses various implementation details. We provide numerical comparisons on several MPC problems in Section V, with concluding remarks in Section VI.

II. Notation and Conventions

For an MPC problem with state $x \in \mathbb{R}^n$, control $u \in \mathbb{R}^m$, and horizon length N, we denote state and control trajectories as $X = \{x_1, x_2, \ldots, x_N\}$ and $U = \{u_1, u_2, \ldots u_{N-1}\}$. Unless otherwise noted, when constraints include subscripts k indicating time steps, they are assumed to apply to all indices $\mathcal{I}_N = \{1, \ldots, N\}$.

III. Conic Augmented Lagrangian

There has been great interest in recent years within the optimization community in optimization over cones, sometimes referred to as *generalized inequalities*. The second-order cone, $\mathcal{K}_{\text{soc}} = \{(v, s) \in \mathbb{R}^{n+1} \,|\, \|v\|_2 \leq s\}$, (also known as the *quadratic cone* or the *Lorentz cone*) has proven particularly useful in control applications including the rocket soft-landing problem [12], [13] and friction cone constraints that appear in manipulation or locomotion tasks [14]. Before methods existed for directly attacking SOCPs, many practitioners linearized this cone, resulting in increased problem sizes and less accurate or sub-optimal solutions.

Most existing specialized "conic" solvers are based on ADMM [15], [16] or interior-point [10] methods. Both of these approaches have tradeoffs: ADMM methods converge slowly (linearly) but are very warm-startable, while interior-point methods converge quickly (quadratically) but aren't well-suited to warm-starting. The augmented Lagrangian method (ALM) is an attractive middle-ground, offering both superlinear convergence and good warm-starting capabilities.

Despite some theoretical work showing promising convergence analyses of ALM for conic programs [17]–[21] and several ALM implementations for solving large-scale semidefinite programs (SDPs) [22], [23], no ALM implementation for SOCPs existed until very recently [24]. Building on that work, our aim is to develop an ALM SOCP solver that exploits the special structure of MPC problems.

Augmented Lagrangian methods solve constrained optimization problems by solving a series of unconstrained problems minimizing the *augmented Lagrangian*:

$$\mathcal{L}_A(x) = f(x) - \lambda^T c(x) + \mu \frac{1}{2} c(x)^T c(x), \tag{1}$$

where $f(x) : \mathbb{R}^n \mapsto \mathbb{R}$ is the objective function, $c(x) : \mathbb{R}^n \mapsto \mathbb{R}^m$ is an equality constraint function, $\lambda \in \mathbb{R}^m$ is a Lagrange multiplier, and $\mu \in \mathbb{R}$ is a penalty weight. This standard form can also be adapted to handle inequality constraints, as described in [11], [25].

After each unconstrained minimization of (1) with respect to x, the penalty μ is increased and the Lagrange multiplier λ is updated according to,

$$\lambda \leftarrow (\lambda - \mu c(x)), \tag{2}$$

which is equivalent to a gradient ascent step on the dual problem [26]. Augmented Lagrangian methods are theoretically capable of superlinear convergence rates, but often exhibit poor "tail-convergence" behavior in practice due to ill-conditioning as μ is increased.

Based on [17], we generalize the augmented Lagrangian method to enforce general conic constraints. Equation (1) can be rewritten as,

$$\mathcal{L}_A(x) = f(x) + \frac{1}{2\mu}(\|\lambda - \mu c(x)\|^2 - \|\lambda\|^2). \tag{3}$$

Comparing the first quadratic penalty term $\lambda - \mu c(x)$ with the standard dual ascent step (2), we see that this reformulation is effectively penalizing the difference between the current and updated Lagrange multiplier estimates.

If our constraint is instead required to lie within the cone $\mathcal{K}$, we can modify the augmented Lagrangian penalty to penalize the difference between the multipliers after the updated multiplier is projected back into the cone,

$$\mathcal{L}_A(x) = f(x) + \frac{1}{2\mu}(\|\Pi_{\mathcal{K}}(\lambda - \mu c(x))\|^2 - \|\lambda\|^2), \tag{4}$$

where $\Pi_{\mathcal{K}}(x) : \mathbb{R}^p \mapsto \mathbb{R}^p$ is the projection operator for the cone $\mathcal{K}$. We refer to (4) as the *conic augmented Lagrangian*.

For simple inequality constraints of the form $c(x) \leq 0$, the projection is onto the non-positive orthant: $\Pi_{\mathcal{K}_-}(x) = \min(0, x)$. Simple closed-form expressions for the projection operator exist for several other cones, including the second-order cone:

$$\Pi_{\mathcal{K}_{\text{soc}}}(x) = \begin{cases} 0 & \|v\|_2 \leq -s \\ x & \|v\|_2 \leq s \\ \frac{1}{2}(1 + s/\|v\|_2)[v^T \; \|v\|_2]^T & \|v\|_2 > |s| \end{cases} \tag{5}$$

where $v = \begin{bmatrix} x_0 \ldots x_{p-1} \end{bmatrix}^T$, $s = x_p$. Analogous to (2), the dual update in the conic case becomes,

$$\lambda \leftarrow \Pi_{\mathcal{K}}(\lambda - \mu c(x)). \tag{6}$$

IV. ALTRO-C Solver

In contrast to many modern "direct" methods for trajectory optimization, ALTRO relies on iterative LQR (iLQR) to remain dynamically feasible at every iteration and exploit the Markovian structure of MPC problems. At each iteration of the iLQR algorithm (summarized in Algorithm 1), a second-order Taylor series approximation of the problem is computed and a backward Riccati recursion is used to compute an update to the nominal (feed-forward) control trajectory and a time-varying LQR (TVLQR) feedback controller. The closed-loop dynamics are then simulated forward to compute an updated state trajectory.

While fast and efficient, the standard iLQR algorithm has no ability to deal with constraints on the states or controls. To handle constraints, ALTRO uses iLQR as the inner unconstrained solver in an augmented Lagrangian method. To overcome the poor tail convergence of the ALM in situations where tight solution tolerances are required, ALTRO performs a final active-set projected Newton "solution-polishing" step using a Cholesky factorization coupled with iterative refinement [11].

Algorithm 1 iLQR

1: **function** iLQR(ℓ, f, X, U)
2: **while** not converged **do**
3: $J(\delta X, \delta U) \leftarrow$ Quadratic expansion of ℓ at X, U
4: $A_{1:N}, B_{1:N} \leftarrow$ Linearize dynamics f at X, U
5: $K_{1:N}, d_{1:N} \leftarrow \text{TVLQR}(J, A_{1:N}, B_{1:N})$
6: $\alpha \leftarrow 1$
7: **for** k=1:N-1 **do**
8: $\bar{u}_k = u_k + K_k(\bar{x}_k - x_k) + \alpha d_k$
9: $\bar{x}_{k+1} \leftarrow f(\bar{x}_k, \bar{u}_k)$
10: **end for**
11: **if** line search conditions satisfied **then**
12: $X \leftarrow \bar{X}, U \leftarrow \bar{U}$
13: **else**
14: Reduce α and go to line 7
15: **end if**
16: **end while**
17: **return** X, U
18: **end function**

A. Implementation Details

The ALTRO solver was adapted to support second-order cone constraints using the augmented Lagrangian formulation introduced in Section III. Analytic first and second-order derivatives of (5) were implemented to calculate the expansions of (4) required by the iLQR algorithm. No changes were made to the active-set solution-polishing method.

In addition to adding second-order cone constraints, the Julia implementation of ALTRO-C has been improved substantially from the original version presented in [11], enabling the competitive timing results demonstrated in Section V. Memory allocations have been eliminated wherever possible, and ALTRO-C has been particularly optimized for small-to-medium-size problems by leveraging loop-unrolling and analytical linear algebra[1]. In addition to excellent performance, ALTRO-C provides a convenient API that dramatically simplifies MPC problem definition and provides convenient and efficient methods for updating the MPC problem between iterations.

V. Examples

The following examples were run on an Intel i7-1165G7 processor. For the QP examples, ALTRO-C was compared against OSQP [9], a state-of-the-art ADMM QP solver optimized for online optimization, and the SOCP examples were compared against ECOS [10], an interior-point SOCP solver designed for embedded applications, COSMO [15], a state-of-the-art ADMM SOCP method, and Mosek[2], a high-quality commercial convex optimization package. All problems were solved to cost and constraint tolerances of 10^{-4}, so the solution-polishing steps of both ALTRO-C and OSQP were disabled. The reported timing results only capture the time to solve the convex optimization problem: All overhead associated with modifying the problem during each MPC iteration is omitted. Future work with emphasis on applications will include the often non-trivial steps required to efficiently update the problem between iterations, although fast and allocation-free methods are already implemented in ALTRO-C[3]. Code for the examples is available at https://github.com/RoboticExplorationLab/altro-mpc-icra2021.

A. Random Linear MPC

We compared the general performance of ALTRO-C and OSQP on a set of random QPs of the following form:

$$\begin{aligned} \underset{X,U}{\text{minimize}} \quad & \sum_{k=1}^{N} \|x_k - \bar{x}_k\|_{Q_k}^2 + \sum_{k=1}^{N-1} \|u_k\|_R^2 \\ \text{subject to} \quad & x_{k+1} = Ax_k + Bu_k, \\ & x_1 = 0, \\ & u^- \leq u_k \leq u^+ \end{aligned} \tag{7}$$

The problems were generated from the following distributions: $Q_{ii} \sim \mathcal{U}(0, 10)$, $R_{ii} = 0.1$, $Q_f = (N-1)Q$, and $u_i^+ = u_i^- = 3$. The reference trajectory $\bar{x}_k, \bar{u}_k$ was generated by randomly generating a control trajectory $\bar{u} \sim \mathcal{N}(0, 1)$ and simulating the system forward. Random dynamics matrices, A and B, were generated such that the eigenvalues of A were within the unit circle and the system was controllable.

At each MPC iteration, the first control was used to simulate the system forward with additive noise: $x_{k+1} = Ax_k + Bu_k + \epsilon_k$, where $\epsilon_k \sim \mathcal{N}(0, \|x_k\|_\infty / 100)$. The primal and dual variables for each solver were then shifted by one time step to warm-start the next solve.

We compare solve times while varying the state dimension n in Fig. 1a, the control dimension m in Fig. 1b, and the time horizon N in Fig. 1c. As expected, both ALTRO-C and OSQP demonstrate linear scaling with respect to the horizon length, while ALTRO-C achieves significantly lower overall times. Both solvers exhibit polynomial scaling in the state and control dimensions, with ALTRO-C achieving better scaling with respect to state dimension. Because ALTRO-C directly optimizes control trajectories using a Ricatti recursion, it exhibits worse scaling with respect to input dimension. While ALTRO-C is faster than OSQP for $m < 15$, OSQP has an advantage for systems with larger input dimensions.

B. Flexible Spacecraft

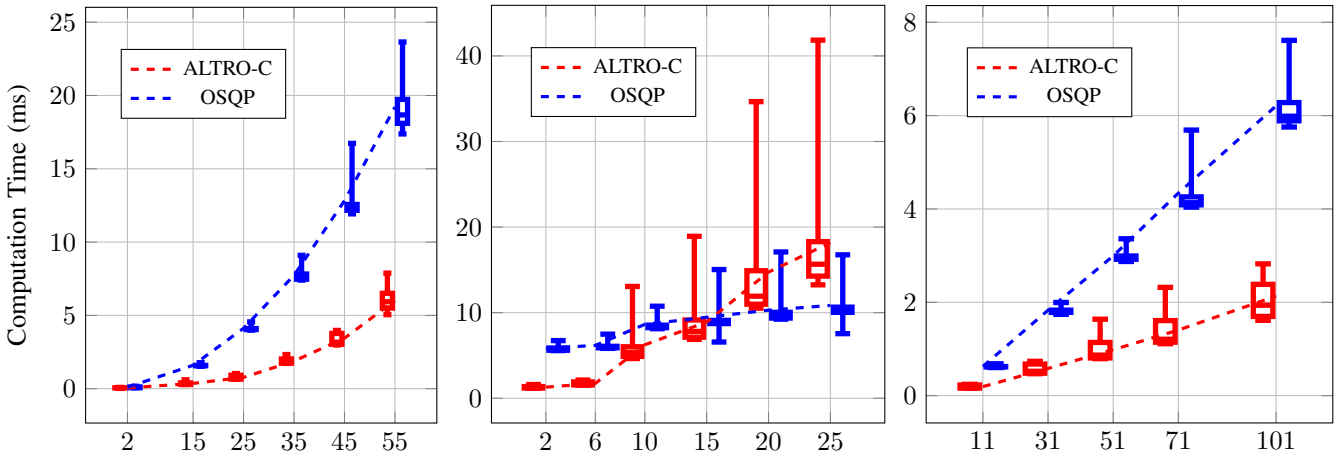
For spacecraft with flexible appendages, controlling the attitude with a naive feedback controller can result in poor performance due to excitation of flexible modes. Running an MPC controller in this scenario is advantageous because it can plan for known disturbances and better reason about actuator constraints [27]. The nonlinear dynamics of a flexible spacecraft [28] were linearized about a reference orientation, resulting in the following linear ODEs,

$$\begin{aligned} J\dot{\omega} + G^T\ddot{\eta} &= \tau_d - u, \\ \ddot{\eta} + C\dot{\eta} + K\eta + \Phi f &= -G\dot{\omega}, \end{aligned} \tag{8}$$

[1]These algorithms are part of the StaticArrays.jl package

[2]https://www.mosek.com/

[3]Available in the Altro.jl and TrajectoryOptimization.jl packages



(a) State dimension (n) (b) Control dimension (m) (c) Time horizon (N)

Fig. 1: Comparison of ALTRO-C and OSQP as a function of a) state dimension ($N = 21, m = 2$), b) control dimension ($N = 21, n = 30$), and c) horizon length ($n = 12, m = 6$). The x-axis is labeled at the sampled sizes.

where $\omega \in \mathbb{R}^3$ is the spacecraft's angular velocity, $\eta \in \mathbb{R}^3$ is the modal displacement, J is the spacecraft's inertia matrix, G is the Jacobian mapping modal displacement to angular displacement, Φ is the Jacobian mapping modal displacement to linear displacement, C is the damping matrix, K is the stiffness matrix, f is the disturbance force, τ_d is the disturbance torque, and $u \in \mathbb{R}^3$ is the control torque. This linearization is valid for inertial pointing scenarios where deviations from the reference attitude are limited to a few degrees. The flexible spacecraft pointing problem can be posed as a convex MPC problem with 12 states, 3 controls, and linear actuator constraints.

For large spacecraft, the dynamics are relatively slow, so a sample rate of 2 Hz and horizon of 40 seconds were used for this problem. The cost function included quadratic penalties on pointing error, angular velocity, excitation of the flexible modes, and the control inputs, which were bounded between $\pm 0.1\,\mathrm{N{\cdot}m}$. As shown in Fig. 2, both solvers are able to leverage warm-starting to reduce the number of iterations required for convergence after a handful of MPC steps, with ALTRO-C being at least twice as fast as OSQP after the first few steps.

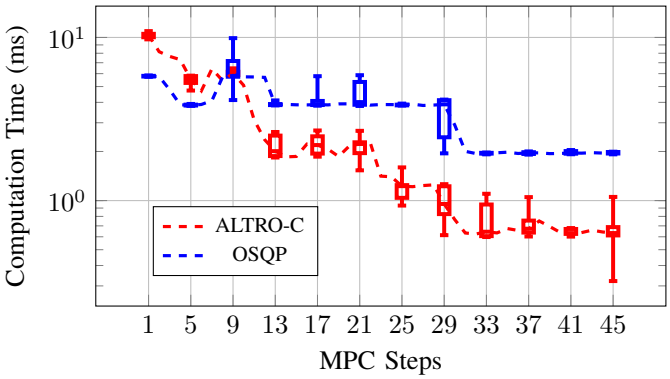


Fig. 2: Solve times for OSQP and ALTRO-C for the flexible spacecraft pointing problem. Both ALTRO-C and OSQP benefit from warm starting, leading to reduced solve times as the simulation progresses. ALTRO-C reaches a significantly faster steady state solve time than OSQP.

C. Quadruped

Quadrupeds are high dimensional, underactuated robotic systems that are challenging to control. Current state-of-the-art approaches simplify the dynamics and pre-specify a foot contact sequence so that the problem can be cast as a quadratic program that can readily be solved online.

Based on [3], we model the robot as a single rigid body with with $x_k \in \mathbb{R}^{12}$ and contact forces acting at the legs treated as control inputs $u_k \in \mathbb{R}^{12}$. A convex problem is obtained by linearizing the dynamics about the current reference, resulting in a problem similar to (7), but with additional friction-cone constraints on the foot contact forces,

$$\left\|f_{i,k}^t\right\|_2 \leq \mu f_{i,k}^n \qquad i \in \{1,\ldots,4\}, \tag{9a}$$

$$0 \leq f_{i,k}^n \qquad i \in \{1,\ldots,4\} \tag{9b}$$

where $f_{i,k}^t$ and $f_{i,k}^n \in \mathbb{R}^3$ are the tangential and normal components of the contact forces of foot i at time step k, respectively. Most MPC implementations further linearize the second-order friction cone constraint (9a),

$$\begin{aligned} -\tilde{\mu} f^n \leq f^{t_x} \leq \tilde{\mu} f^n, \\ -\tilde{\mu} f^n \leq f^{t_y} \leq \tilde{\mu} f^n, \end{aligned} \tag{10}$$

resulting in a QP. Both the original SOCP and linearized QP versions of the problem can be solved with ALTRO-C.

ALTRO-C was compared to OSQP using the QP formulation and ECOS for the SOCP formulation. The MuJoCo simulator [29] was used to simulate the full nonlinear dynamics of the quadruped walking in place. ALTRO-C and OSQP

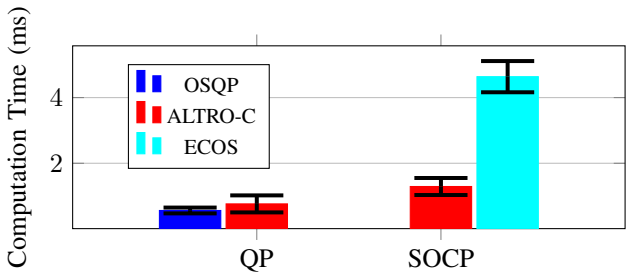


Fig. 3: Timing results for the quadruped benchmark averaged over 60 MPC iterations. The QP results are for the linearized friction cone, while the SOCP results are for the second-order friction cone. Error bars denote one standard deviation.

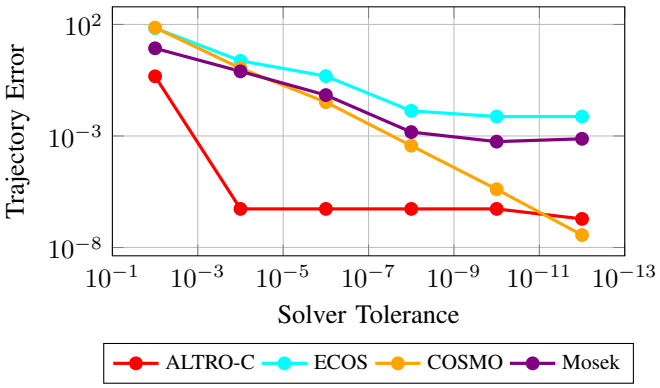


Fig. 4: Convergence comparison for conic solvers. The error relative to a reference trajectory is shown as a function of solver tolerance.

were warm started with the solution from the previous step. Timing results are shown in Fig. 3. While OSQP is slightly faster than ALTRO-C, both solvers achieve sub-millisecond times. Interestingly, ALTRO-C achieves nearly the same solve time on the SOCP formulation, while ECOS is many times slower. This result demonstrates that the common practice of linearizing friction cones to achieve faster MPC solve times may be unnecessary with better-optimized solvers.

Finally, we note that non-convex versions of this problem have also been successfully demonstrated using the IPOPT solver [30]. Because ALTRO is a general nonlinear solver, it can also seamlessly handle such non-convex extensions without having to change solvers, and while maintaining high performance.

D. Rocket Soft Landing

The rocket soft-landing problem involves landing a vehicle on the surface of a planet while respecting actuator and safety constraints. Several previous works have formulated this problem as a convex second-order cone program [12], [13], [31], [32]. It is standard to decompose the MPC problem into two separate controllers: a long-horizon controller for the translation dynamics, and a short-horizon attitude controller. Here we consider the translation problem, adapted from [13],

$$\begin{aligned}
\underset{X,U}{\text{minimize}} \quad & \sum_{k=1}^{N} \|x_k - \bar{x}_k\|_{Q_k}^2 + \sum_{k=1}^{N-1} \|u_k\|_R^2 && \text{(11a)}\\
\text{subject to} \quad & x_{k+1} = Ax_k + Bu_k + b, && \text{(11b)}\\
& x_1 = x_{init}, && \text{(11c)}\\
& \|u_k\|_2 \le u_{max}, && \text{(11d)}\\
& \left\|[u_{k,x}\ u_{k,y}]^T\right\|_2 \le \alpha_u u_{k,z}, && \text{(11e)}\\
& \left\|[x_{k,x}\ y_{k,y}]^T\right\|_2 \le \alpha_x x_{k,z} && \text{(11f)}
\end{aligned}$$

where $x_k \in \mathbb{R}^6, u_k \in \mathbb{R}^3 = \begin{bmatrix} u_{k,x} & u_{k,y} & u_{k,z} \end{bmatrix}^T$, and the dynamics are approximated by a point mass subject to gravity. The three second-order cone constraints (11d), (11e), and (11f) enforce a maximum thrust, thrust angle, and a safe glideslope region, respectively.

With a horizon length of 20 time steps, ALTRO-C was roughly ten times faster than ECOS, converging in under 4 iterations and 0.37 ± 0.33 ms, compared to 3.80 ± 0.18 ms for ECOS. Fig. 4 highlights the convergence characteristics of ALTRO-C compared to several other state-of-the-art conic solvers. A reference "ground-truth" solution was generated using COSMO [15] with very tight tolerances, and the distance to the ground-truth solution was computed for decreasing solver tolerances. As shown, ALTRO-C converges extremely quickly to a solution very close to the true solution, while other solvers require much tighter tolerances (implying longer solve times) to achieve similar results.

E. Grasp Optimization

Grasp optimization for robotic manipulators can be formulated as an SOCP [14]. We model a two-finger robotic manipulator grasping a box and tracking a reference trajectory. The box is subject to contact forces from the gripper F^1 and $F^2 \in \mathbb{R}^3$, and gravity F_g. The inward pointing normal v^i for the i^{th} contact point is useful for separating the contact forces into the tangential and normal components. The problem setup is illustrated in Fig. 5.

Similar to (7), a quadratic cost function is used to penalize distance from a reference trajectory. Contact forces are treated as control inputs, and are subject to the following constraints:

$$\begin{aligned}
& \left\|(I - v_k^i (v_k^i)^T) F_k^i\right\|_2 \le \mu (v_k^i)^T F_k^i, \quad i = 1,2 && \text{(12a)}\\
& (v_k^i)^T F_k^i \le f_{max}, \quad i = 1,2 && \text{(12b)}\\
& \alpha_k = C_k u_k. && \text{(12c)}
\end{aligned}$$

Equation (12a) enforces Coulomb friction, where $\left\|(I - v^i (v^i)^T) F^i\right\|_2$ is the magnitude of the normal component, $(v^i)^T F^i$ is the magnitude of the tangential component, and μ is the coefficient of friction. Equation (12b) bounds the normal force, and (12c) ensures that the torques generate the pre-specified reference angular acceleration $\alpha_k \in \mathbb{R}^3$.

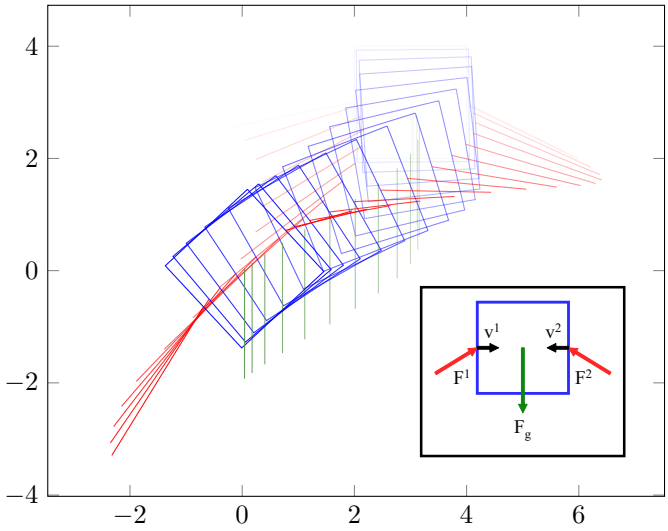


Fig. 5: A free-body diagram and trajectory snapshots for a manipulation task. F^1 and F^2 are the contact forces from the gripper, and v^i is the inward pointing normal for the object at the i^{th} contact point. F_g is the gravitational force. The object pose is shown in blue, contact forces are shown in red, and the gravity vector is in green. Snapshots that are more transparent correspond to earlier time steps in the trajectory.

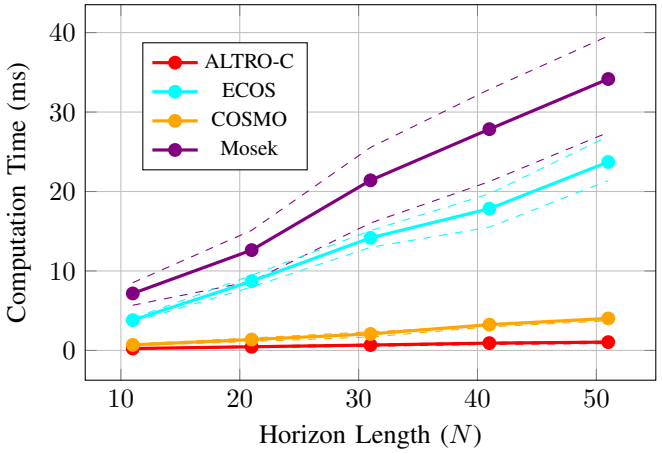


Fig. 6: Computation time comparison for the grasp optimization MPC problem. The solid lines represent average run-time, while the dotted lines represent the 1st and 3rd quartiles.

To ease visualization, the reference trajectory in Fig. 5 was kept planar, but could easily be extended to 3D. The performance of ALTRO-C was compared to several other conic solvers for different MPC horizon lengths N. As shown in Fig. 6, while all solvers exhibit linear scaling with respect to N, ALTRO-C exhibits the fastest run-times—averaging 0.24 ms for 11 knot points and 1.0 ms for 51 knot points.

VI. CONCLUSIONS

We have presented ALTRO-C, a conic augmented Lagrangian method for solving model-predictive control problems with general convex conic constraints. The method was implemented by modifying the open-source ALTRO solver and demonstrated on several benchmark control problems. ALTRO-C fully exploits the structure of trajectory optimization problems, achieves fast convergence to moderate tolerances, and offers good warm-starting capabilities, making it ideal for MPC applications. Comparisons to several QP and SOCP solvers show that our algorithm delivers state-of-the-art performance on small-to-medium-size MPC problems and is suitable for many real-time control applications. Additionally, unlike other specialized conic solvers, ALTRO-C can be readily applied to nonlinear and non-convex problems.

Future work will investigate extending ALTRO-C to work with additional cones, including the semi-definite cone. Further benchmarking results against other state-of-the-art MPC solvers, such as the newly released HPIPM [8], are also left for future work. While the current Julia implementation offers significant benefits, a lightweight C implementation of the algorithm could also be useful for resource-constrained embedded applications.

ACKNOWLEDGMENT

This work was supported by a NASA Early Career Faculty Award (Grant Number 80NSSC18K1503). This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1656518. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] L. Blackmore, "Autonomous Precision Landing of Space Rockets," *The Bridge*, vol. 4, no. 46, pp. 15–20, 2016.

[2] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for Atlas," *Autonomous Robots*, vol. 40, no. 3, pp. 429–455, 2016.

[3] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic Locomotion in the MIT Cheetah 3 Through Convex Model-Predictive Control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid: IEEE, Oct. 2018, pp. 1–9.

[4] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: A parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[5] S. Kuindersma, F. Permenter, and R. Tedrake, "An Efficiently Solvable Quadratic Program for Stabilizing Dynamic Locomotion," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, Hong Kong, China: IEEE, May 2014, pp. 2589–2594.

[6] G. Frison, D. K. M. Kufoalor, L. Imsland, and J. B. Jørgensen, "Efficient implementation of solvers for linear model predictive control on embedded devices," in *2014 IEEE Conference on Control Applications (CCA)*, 2014, pp. 1954–1959.

[7] G. Frison, H. H. B. Sørensen, B. Dammann, and J. B. Jørgensen, "High-performance small-scale solvers for linear Model Predictive Control," in *2014 European Control Conference (ECC)*, 2014, pp. 128–133.

[8] G. Frison and M. Diehl, "HPIPM: A high-performance quadratic programming framework for model predictive control," *arXiv preprint arXiv:2003.02547*, 2020.

[9] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.

[10] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP solver for embedded systems," in *2013 European Control Conference (ECC)*, Zurich: IEEE, Jul. 2013, pp. 3071–3076.

[11] T. A. Howell, B. E. Jackson, and Z. Manchester, "ALTRO: A Fast Solver for Constrained Trajectory Optimization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, Nov. 2019.

[12] B. Açikmeşe and S. R. Ploen, "Convex Programming Approach to Powered Descent Guidance for Mars Landing," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 5, pp. 1353–1366, Sep. 2007.

[13] B. Açikmeşe, J. M. Carson III, and L. Blackmore, "Lossless Convexification of Nonconvex Control Bound and Pointing Constraints of the Soft Landing Optimal Control Problem," *IEEE Transactions on Control Systems Technology*, vol. 21, pp. 2104–2113, Nov. 2013.

[14] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra and its Applications*, vol. 284, no. 1-3, pp. 193–228, Nov. 1998.

[15] M. Garstka, M. Cannon, and P. Goulart, "COSMO: A conic operator splitting method for large convex problems," in *European Control Conference*, Naples, Italy, 2019.

[16] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd, "Conic Optimization via Operator Splitting and Homogeneous Self-Dual Embedding," *Journal of Optimization Theory and Applications*, vol. 169, no. 3, pp. 1042–1068, Jun. 2016.

[17] Y.-J. Liu and L.-W. Zhang, "Convergence of the augmented Lagrangian method for nonlinear optimization problems over second-order cones," *Journal of optimization theory and applications*, vol. 139, no. 3, pp. 557–575, 2008.

[18] A. Shapiro and J. Sun, "Some properties of the augmented Lagrangian in cone constrained optimization," *Mathematics of Operations Research*, vol. 29, no. 3, pp. 479–491, 2004.

[19] D. Sun, J. Sun, and L. Zhang, "The rate of convergence of the augmented Lagrangian method for nonlinear semidefinite programming," *Mathematical Programming*, vol. 114, no. 2, pp. 349–391, 2008.

[20] Y. Cui, D. Sun, and K.-C. Toh, "On the R-superlinear convergence of the KKT residuals generated by the augmented Lagrangian method for convex composite conic programming," *Mathematical Programming*, vol. 178, no. 1-2, pp. 381–415, 2019.

[21] N. T. V. Hang, B. S. Mordukhovich, and M. E. Sarabi, "Augmented Lagrangian Method for Second-Order Cone Programs under Second-Order Sufficiency," *arXiv:2005.04182 [cs, math]*, May 2020.

[22] X.-Y. Zhao, D. Sun, and K.-C. Toh, "A newton-cg augmented lagrangian method for semidefinite programming," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1737–1765, 2010.

[23] X. Li, D. Sun, and K.-C. Toh, "QSDPNAL: A two-phase augmented Lagrangian method for convex quadratic semidefinite programming," *Mathematical Programming Computation*, vol. 10, no. 4, pp. 703–743, 2018.

[24] L. Liang, D. Sun, and K.-C. Toh, *An Inexact Augmented Lagrangian Method for Second-order Cone Programming with Applications*, 2020.

[25] M. Toussaint, "A Novel Augmented Lagrangian Approach for Inequalities and Convergent Any-Time Non-Central Updates," *arXiv:1412.4329 [math]*, Dec. 2014.

[26] D. P. Bertsekas, *Constrained Optimization and Lagrange Multiplier Methods*. Athena Scientific, 1996.

[27] K. Tracy and Z. Manchester, "Model-Predictive Attitude Control for Flexible Spacecraft During Thruster Firings," in *AAS/AIAA Astrodynamics Specialist Conference*, Lake Tahoe, CA, Aug. 2020.

[28] P. W. Likins and G. E. Fleischer, "Results of flexible spacecraft attitude control studies utilizing hybrid coordinates," *Journal of Spacecraft and Rockets*, vol. 8, no. 3, pp. 264–273, Mar. 1971.

[29] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proceedings of the IEEERAS International Conference on Intelligent Robots*, 2012.

[30] G. Bledt and S. Kim, "Implementing Regularized Predictive Control for Simultaneous Real-Time Footstep and Ground Reaction Force Optimization," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China: IEEE, Nov. 2019, pp. 6316–6323.

[31] L. Blackmore, B. Açikmeşe, and D. P. Scharf, "Minimum-Landing-Error Powered-Descent Guidance for Mars Landing Using Convex Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 4, pp. 1161–1171, Aug. 2010.

[32] L. Blackmore, B. Açikmeşe, and J. M. Carson III, "Lossless convexification of control constraints for a class of nonlinear optimal control problems," in

2012 American Control Conference (ACC), Jun. 2012, pp. 5519–5525.