AAS 19-904

FAST SOLUTION OF OPTIMAL CONTROL PROBLEMS WITH L1 COST

Simon Le Cleac'h[*], Zachary Manchester [†]

We propose a fast algorithm for solving optimal control problems with L1 control cost. Convergence to the global optimum is guaranteed for systems with linear dynamics, and the algorithm can also be used to find local optima for nonlinear dynamical systems. Our approach relies on the alternating direction method of multipliers (ADMM) and uses a fast trajectory optimization solver based on iterative LQR. The low computational complexity coupled with the fast execution of this algorithm make it suitable for implementation in flight software.

INTRODUCTION

Classical optimal control theory stands on solid analytical foundations, foremost among which is the Pontryagin minimum principle.[1] This principle assumes continuity of the cost function and differentiability properties about the system's dynamics. However, dealing with non-differentiable cost functions has proven difficult as the ability to differentiate with respect to the control input facilitates the theoretical and numerical analysis of the problem.

Optimal control problems with L1 cost belong to this category of difficult problems because of the nondifferentiabilty of the L1-norm. Yet there is a great interest in these problems as the L1-norm is sometimes more representative of the true cost than typical quadratic cost functions. For instance, L1-norm problems naturally arise from minimum-time[2] or minimum-fuel[3] problems. This type of optimal control problem, therefore, has a broad range of applications in astrodynamics.

Vossen[4] proposed regularization and augmentation techniques to solve L1-norm problems in a restricted setting. Indeed, the proposed methods assume that the control command appears linearly in the dynamics. Bako[2] derived a numerical approach for minimum-time problems with linear time-variant dynamics by formulating the problem using an L1 cost on the control inputs and solving it numerically with CVX, a general-purpose solver for convex optimization problems.[5] This approach, however, neither fully exploits the sparse nature of optimal control problems, nor can it handle control problems with nonlinear dynamics.

In contrast, we derive a dedicated algorithm based on the alternating-direction method of multipliers (ADMM)[6] that also generalizes to non-linear dynamics. Moreover, it relies on a fast, robust solver with low-memory footprint.[7] These properties make possible an implementation of the solver on an embedded system. The algorithm proposed here is suitable for onboard real-time implementation in flight software for applications such as spacecraft formation flying.[8]

[*] Ph.D candidate, Department of Mechanical Engineering, Stanford University, Stanford, California, USA.

[†] Assistant Professor, Department of Aeronautics & Astronautics, Stanford University, Stanford, California, USA.

The paper proceeds as follows: the next section introduces the formulation of the L1 cost problem and provides an iterative solution technique. Then, the third section generalizes the iterative solution to problems with nonlinear dynamics. The fourth and fifth sections demonstrate a detailed application of the proposed algorithm to a spacecraft rendezvous problem and discusses the algorithm's performance. Finally, concluding remarks are outlined in the sixth section.

PROBLEM FORMULATION AND ITERATIVE SOLUTION

Our objective is to control a linear time-variant dynamical system with state x and control u, and with fixed initial state x_0. We define a cost function with quadratic stage costs l_k and quadratic final cost l_f applied to the state, and an L1 cost on the control u:

$$\begin{aligned} \underset{x_{1:N},\, u_{0:N-1}}{\text{minimize}} \quad & J(x_{0:N}, u_{0:N-1}) = l_f(x_N) + \sum_{k=0}^{N-1} l_k(x_k) + \alpha ||u_k||_1, \\ \text{subject to} \quad & x_{k+1} = A_k x_k + B_k u_k,\ k = 0, \dots, N-1, \\ & l_f(x_N) = \frac{1}{2} x_N^T Q_N x_N + x_N^T q_f + c_f, \\ & l_k(x_k) = \frac{1}{2} x_k^T Q x_k + x_k^T q + c. \end{aligned} \tag{1}$$

To solve this constrained optimization problem, we rely on the alternating direction method of multipliers (ADMM).[6] We introduce dummy variables y_k to replace the controls u_k in the cost function and formulate an equivalent problem by adding equality constraints:

$$\begin{aligned} \underset{x_{1:N},\, y_{0:N-1}}{\text{minimize}} \quad & J(x_{0:N}, y_{0:N-1}) = l_f(x_N) + \sum_{k=0}^{N-1} l_k(x_k) + \alpha ||y_k||_1, \\ \text{subject to} \quad & x_{k+1} = A_k x_k + B_k u_k,\ k = 0, \dots, N-1, \\ & u_k = y_k,\ k = 0, \dots, N-1. \end{aligned} \tag{2}$$

We also introduce the following shorthand notation for readability purposes,

$$X = x_{0:N}, \tag{3}$$

$$U = u_{0:N-1}, \tag{4}$$

$$Y = y_{0:N-1}, \tag{5}$$

$$\Lambda = \lambda_{0:N-1}. \tag{6}$$

Following ADMM, we use an augmented Lagrangian method. We add to the Lagrangian a quadratic penalty function that penalizes violation of the equality constraints in Problem 2,

$$\mathcal{L}_\rho(X, U, Y, \Lambda) = J(X, Y) + \sum_{k=0}^{N-1} \lambda_k^T (u_k - y_k) + \frac{\rho}{2} ||u_k - y_k||_2^2, \tag{7}$$

where ρ is a penalty weight and λ is a Lagrange multiplier. The Lagrange multiplier and penalty term associated with the dynamics constraints in Problem 2 are omitted here, as they are implicitly enforced in the LQR algorithm used to solve for X and U. The ADMM algorithm solves Problem

2 iteratively. Each iteration decomposes into three sequential steps. In the following sections, the superscript i on variables x, u, y, and λ indicates the iteration step of the ADMM algorithm,

$$X^{i+1}, U^{i+1} = \text{argmin}_{X,U} \mathcal{L}_\rho(X, U, Y^i \Lambda^i), \tag{8}$$

$$Y^{i+1} = \text{argmin}_{Y} \mathcal{L}_\rho(X^{i+1}, U^{i+1}, Y, \Lambda^i), \tag{9}$$

$$\lambda_k^{i+1} = \lambda_k^i + \rho(u_k^{i+1} - y_k^{i+1}),\ k = 0, \dots, N-1. \tag{10}$$

The first step, defined by Equation (8), is equivalent to solving a linear quadratic regulator (LQR) problem, which has a closed-form solution. The second step, Equation (9), is a soft-threshold operation, which can also be computed in closed-form (see Equation (24)). Finally the third step is a Lagrange multiplier update. It is important to note that all these steps are fast to compute. The procedure used to solve the optimization problem is formally described in Algorithm 1. The procedure requires as input; the initial state of the system x_0, an initial guess for the control trajectory $U_{initial}$, a penalty weight ρ and a termination threshold ϵ. Once the termination threshold is reached, the procedure returns control and state trajectories minimizing the L1 cost function.

Algorithm 1 L1 Cost Optimizer

```
procedure L1CostOptimizer($x_0, U_{initial}; \rho, \epsilon$)
    $X \leftarrow$ DynamicsRollout($x_0, U_{initial}$)
    $Y, \Lambda \leftarrow 0, 0$
    repeat
        $X, U \leftarrow$ OptimalControlUpdate($Y, \Lambda; \rho$)
        $Y \leftarrow$ SoftThresholdUpdate($X, U, \Lambda, \rho$)
        $\Lambda \leftarrow \Lambda + \rho\,(U - Y)$
    until $||U - Y||_2 < \epsilon$
    return $X, U$
end procedure
function OptimalControlUpdate($Y, \Lambda; \rho$)
    $X, U \leftarrow$ Solution to LQR Problem 11
    return $X, U$
end function
function SoftThresholdUpdate($X, U, \Lambda; \rho$)
    $Y \leftarrow$ L1 projection operation using Equations (23) and (24)
    return $Y$
end function
```

Optimal Control Update

The first step consists of minimizing the augmented Lagrangian with respect to the trajectories in the state and control spaces. This is equivalent to solving the linear quadratic regulator problem. Indeed the L1 cost term is constant since the set of variables Y^i is fixed for this optimization step,

$$\begin{aligned} \underset{X,U}{\text{minimize}} \quad & \mathcal{L}_\rho(X, U, Y^i, \Lambda^i), \\ \text{subject to} \quad & x_{k+1} = A_k x_k + B_k u_k,\ k = 0, \dots, N-1. \end{aligned} \tag{11}$$

This LQR problem has a closed-form solution in terms of the dynamic Riccati equation, which we summarize here for convenience:

$$
\begin{aligned}
u_k &= -K_k x_k - b_k, && (12)\\
K_k &= D_k^{-1} C_k, && (13)\\
a_k &= q_k + A_k^T g_{k+1}, && (14)\\
b_k &= (\lambda_k^i + \rho y_k^i) + B_k^T g_{k+1}, && (15)\\
C_k &= B_k^T H_{k+1} A_k, && (16)\\
D_k &= \rho + B_k^T H_{k+1} B_k, && (17)\\
H_k &= Q_k + K_k^T D_k K_k - K_k^T C_k - C_k^T K_k, && (18)\\
H_N &= Q_N, && (19)\\
g_k &= q_k + a_k + (K_k^T D_k - C_k^T) b_k - K_k^T b_k, && (20)\\
g_N &= q_N. && (21)
\end{aligned}
$$

For a detailed derivation of the set of Equations (12) - (21), we refer the reader to works of Li[9] and Manchester.[10]

Soft Threshold Update

As a second step, we minimize the augmented Lagrangian with respect to $y_{0:N-1}$ using the state and control trajectories obtained from Problem 11:

$$Y^{i+1} = \text{argmin}_Y \Big\{ \mathcal{L}_\rho(X^{i+1}, U^{i+1}, Y, \Lambda^i) \Big\}. \tag{22}$$

This optimization can be performed separately for each y_k since the augmented Lagrangian function is separable across these variables. Furthermore, the fact that the L1-norm is separable across dimensions allows us to minimize separately on each dimension of y_k. This minimization problem has a closed-form solution in terms of the soft-thresholding operator, $\mathcal{S}_\tau$:[6]

$$y_k^{i+1} = \mathcal{S}_{\alpha/\rho}\left(u_k^{i+1} + \frac{\lambda_k^i}{\rho} \right), \tag{23}$$

$$\mathcal{S}_\tau(s) = \begin{cases} s - \tau & s > \tau \\ 0 & |s| \le \tau \\ s + \tau & s < -\tau. \end{cases} \tag{24}$$

Lagrange Multiplier Update

The last step is the Lagrange multiplier update, and is defined as follows:

$$\lambda_k^{i+1} = \lambda_k^i + \rho(u_k^{i+1} - y_k^{i+1}). \tag{25}$$

Equation (25) can be interpreted as shifting the value of the penalty term into the Lagrange multiplier term.

GENERALIZATION TO NONLINEAR DYNAMICS

The method presented in the previous section can be generalized to control problems with non-linear dynamics and constraints on the state and inputs. To include these generalizations we adapt the optimal control update. We add the constraints on the state and control as well as the non-linear dynamics constraint to the optimal control update. The LQR problem is therefore replaced by an optimal control problem with state and control constraints and non-linear dynamics defined as follows:

$$\begin{aligned} \underset{X,U}{\text{minimize}} \quad & \mathcal{L}_\rho(X, U, Y^i, \Lambda^i), \\ \text{subject to} \quad & x_{k+1} = f(x_k, u_k),\ k = 0, \dots, N-1, \\ & \underline{x} \leq x_k \leq \overline{x},\ k = 1, \dots, N, \\ & \underline{u} \leq u_k \leq \overline{u},\ k = 0, \dots, N-1. \end{aligned} \tag{26}$$

We observe that the Problem 26 has a differentiable cost function. Practically, we rely on a fast trajectory optimization solver (ALTRO)[7] to perform this optimal control update. By including these generalizations, we loose the convexity of the control problem and therefore the algorithm is not guaranteed to converge to a global optimum. Solving the nonlinear problem is more complex than a single LQR solution. However, the algorithm reuses the previous control trajectory to warm-start the optimal control update. The solving time of this step, therefore, remains fast.

ASTRODYNAMICS APPLICATION

In this section, we present solutions obtained using our iterative solver. These solutions illustrate the performance of the solver on both linear and nonlinear dynamical systems.

Problem Motivation

We apply our solver to the satellite rendezvous problem. Our L1 cost minimization solver is particularly suited to this type of control problem. Indeed, some satellites rely on reaction control system (RCS) thrusters for attitude and/or translation control. These RCS thrusters can apply forces only inside a limited range. Applying a force close to zero or above a given limit is impossible. The benefit of using L1 cost to penalize thruster use is therefore twofold: first, the L1 cost is a metric that accurately quantifies fuel-consumption compared to a quadratic cost. Second, the L1 cost encourages sparsity of the control trajectory, which means that during large portions of the trajectory the applied control will be zero, and when control is applied, its value is saturated at the maximum possible force. This is similar to an impulsive or bang-off-bang control strategy that is desirable for RCS thrusters. Control trajectories obtained using a quadratic cost, in contrast, are typically smooth and non-zero over most of the trajectory, which is far more difficult to implement on real hardware. An example is provided in Figure 1 for a comparison of the control trajectories obtained using quadratic and L1 cost.

Model

This section presents our model of the rendezvous problem. In this setting, the objective is to control a chaser spacecraft so that it approaches a target spacecraft in order to perform a docking maneuver. We focus on a concrete example: the Pathfinder for Autonomous Navigation (PAN)

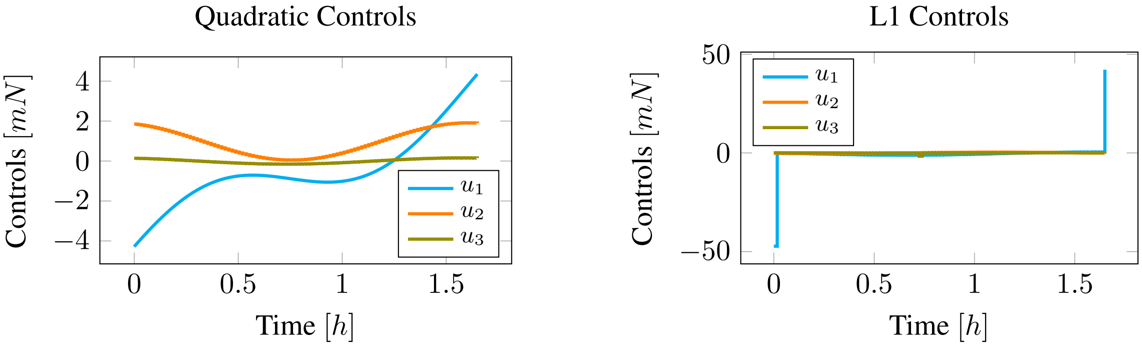

Figure 1. Comparison of the control trajectories obtained using a quadratic control cost and a L1 control cost. These control trajectories produce state trajectories with the same initial state and final state.

mission which was selected for NASA's CubeSat Launch Initiative.[11] In this context, both the chaser and the target spacecraft are in a low Earth orbit with small eccentricity.

Linear Dynamical Model: In order to verify the performance of our solver we apply it first on a linearized version of the rendezvous problem. We therefore use the Clohessy-Wiltshire equations.[12] We refer to the work of Curtis[13] for a detailed description of these equations. The state vector, x, is composed of the position vector $\mathbf{p}$ and velocity vector $\dot{\mathbf{p}}$ of the chaser spacecraft expressed in a frame centered on the target satellite. The control vector, u, is the force vector applied on the satellite. The linear dynamical model of the rendezvous problem is defined as follows,

$$x = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \dot{p}_1 \\ \dot{p}_2 \\ \dot{p}_3 \end{bmatrix}, \qquad \dot{x} = \begin{bmatrix} \dot{p}_1 \\ \dot{p}_2 \\ \dot{p}_3 \\ 3n^2 p_1 + 2np_2 + u1/m \\ -2n\dot{p}_1 + u2/m \\ -n^2 p_3 + u3/m \end{bmatrix}. \tag{27}$$

Where n is the mean motion of the target satellite's orbit.

Nonlinear Dynamical Model: The Clohessy-Wiltshire equations are obtained by linearizing the dynamics of the 2-satellite system. This linearization makes several assumptions; The target satellite's orbit must be circular, and the satellites are not subject to atmospheric drag. This set of assumptions is not respected in our example. We, therefore, also test our algorithm on a dynamical system that accounts for these nonlinearities. The forces applied to these satellites are as follows,

$$\mathbf{F}^{chaser} = \mathbf{F}_g^{chaser} + \mathbf{F}_d^{chaser} + \mathbf{F}_c^{chaser} \tag{28}$$

$$\mathbf{F}^{target} = \mathbf{F}_g^{target} + \mathbf{F}_d^{target}, \tag{29}$$

where $\mathbf{F}_g$ is the force due to Earth's gravitation, $\mathbf{F}_d$ is the drag force due to Earth's atmosphere, $\mathbf{F}_c$ is the control force applied to the chaser satellite. The force due to Earth's gravitation can be modeled as follows,

$$\mathbf{F}_g = -\frac{\mu m}{||\mathbf{p}||_2^3}\mathbf{p}, \tag{30}$$

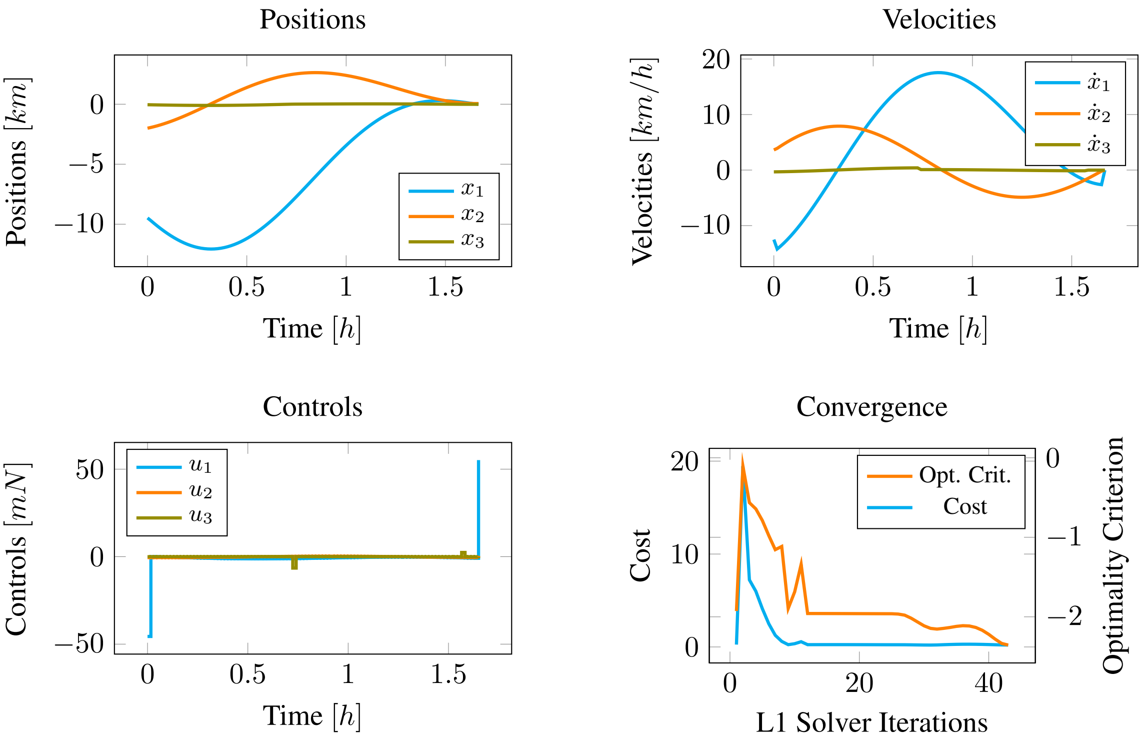

Figure 2. Linear dynamics: positions, velocities and unconstrained control trajectories.

where μ is Earth's standard gravitational parameter, m is the satellite mass, $\mathbf{p}$ is the satellite position in the Earth-centered inertial frame (ECI). The drag force due to Earth's atmosphere can be modeled as follows,

$$\mathbf{F}_d = -\frac{1}{2} C_d \rho A ||\mathbf{v}_{rel}||_2 \mathbf{v}_{rel} \tag{31}$$

$$\mathbf{v}_{rel} = \dot{\mathbf{p}} + \mathbf{p} \times \boldsymbol{\omega}, \tag{32}$$

where C_d is the drag coefficient of the satellite, ρ is the atmospheric density, A is the satellite's cross sectional area, $\mathbf{v}_{rel}$ is the velocity of the satellite with respect to the atmosphere, $\dot{\mathbf{p}}$ is the satellite's velocity in the ECI, $\boldsymbol{\omega}$ is Earth's angular velocity vector in the ECI.

Using the forces computed in Equations (28) and (29), we derive the dynamical model of the system:

$$x = \begin{bmatrix} \mathbf{p}^{chaser} \\ \mathbf{p}^{target} \\ \dot{\mathbf{p}}^{chaser} \\ \dot{\mathbf{p}}^{target} \end{bmatrix}, \qquad \dot{x} = \begin{bmatrix} \dot{\mathbf{p}}^{chaser} \\ \dot{\mathbf{p}}^{target} \\ \left(\mathbf{F}_g^{chaser} + \mathbf{F}_d^{chaser} + u\right)/m^{chaser} \\ \left(\mathbf{F}_g^{target} + \mathbf{F}_d^{target}\right)/m^{target}. \end{bmatrix}. \tag{33}$$

EXPERIMENTS

Scenario

This section presents the solutions obtained using the ADMM solver on a realistic rendezvous example. We define a scenario close the specifications of the Pathfinder for Autonomous Navigation

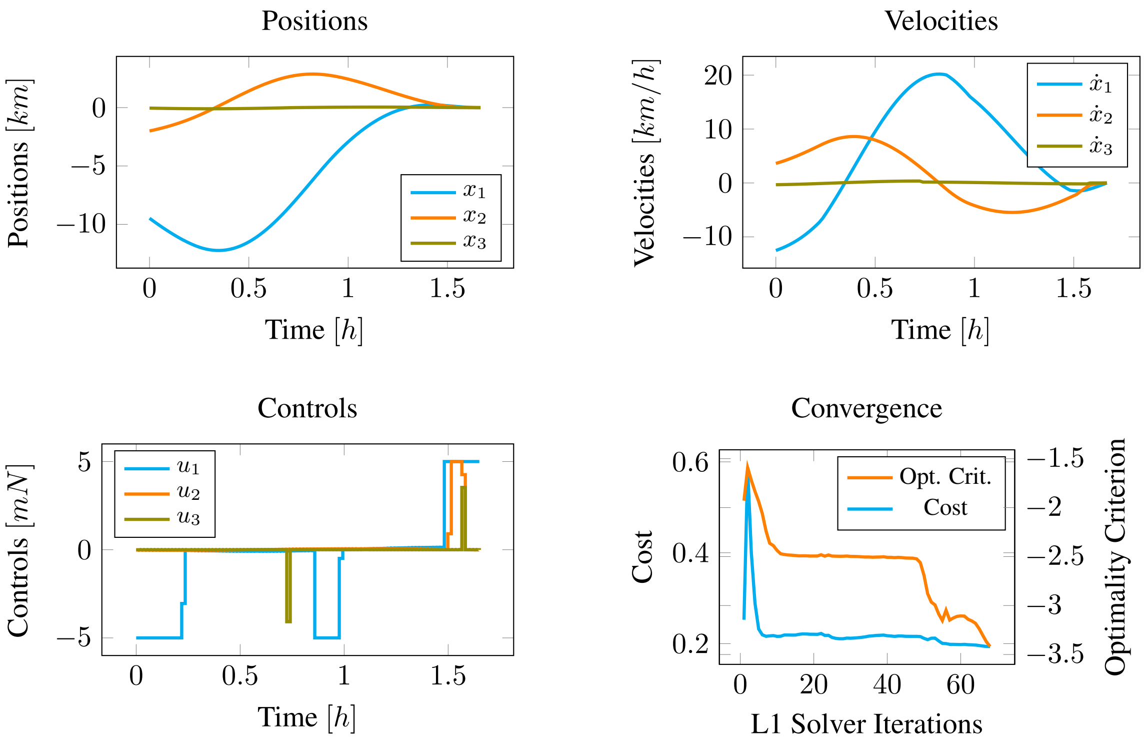

Figure 3. Linear dynamics: positions, velocities and constrained control trajectories.

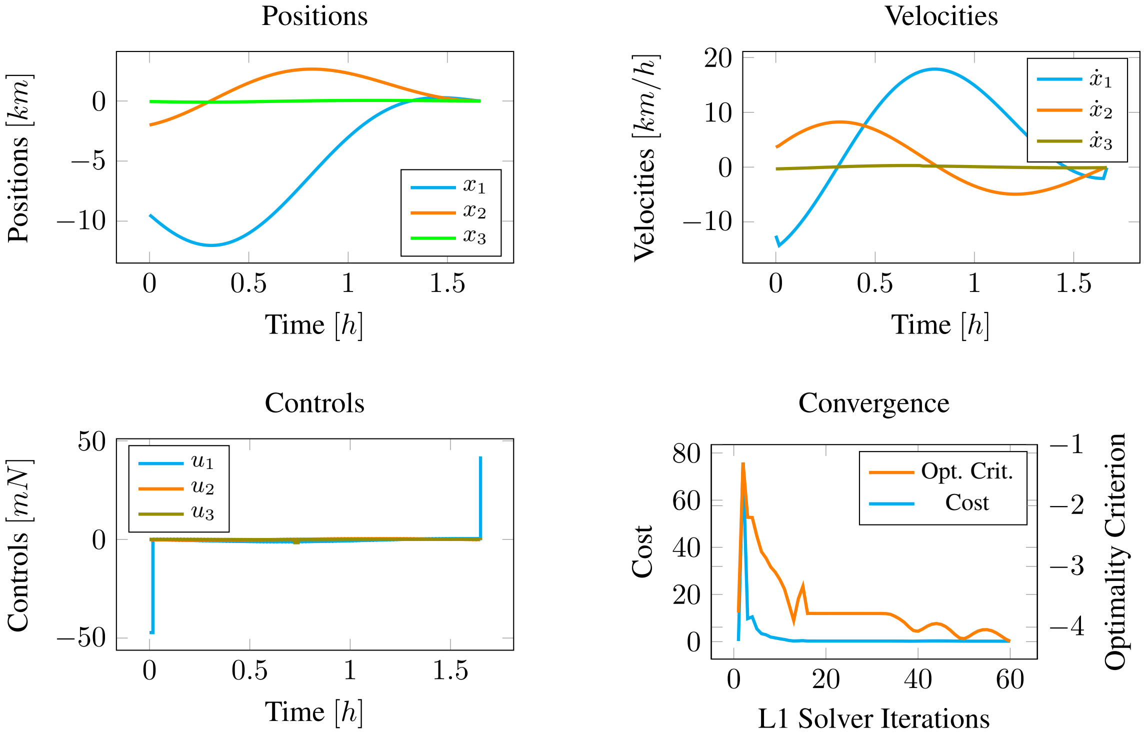

Figure 4. Nonlinear dynamics: positions, velocities and unconstrained control trajectories.

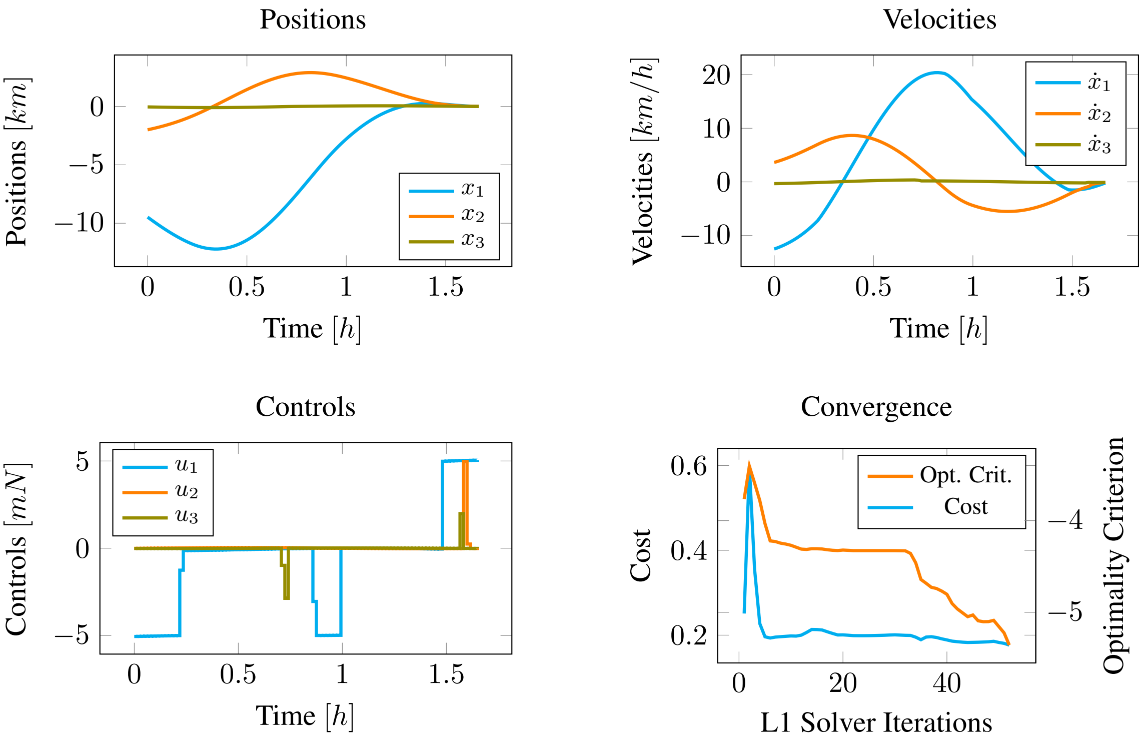

Figure 5. Nonlinear dynamics: positions, velocities and constrained control trajectories.

mission. The target satellite is released on a circular orbit at 500 km altitude. The chaser satellite is released at the same time within a distance of 50.0 m from the target satellite. The initial relative velocity between the chaser and target satellites is about 1 m/s. Using these initial conditions, we simulate the dynamical system forward for one hour without control. During this period, the two satellites drift away from each other. At the end of this period, we compute an optimal control trajectory for the chaser satellite to approach the target satellite. The control trajectory is computed with a horizon of approximately two hours, which corresponds to slightly more than one complete orbit. For our experiments, we use a cost function with two terms that account for the fuel-minimization objective and the final rendezvous goal,

$$J(x_{0:N}, u_{0:N-1}) = \frac{1}{2} x_N^T Q_f x_N + \sum_{k=0}^{N-1} \alpha ||u_k||_1. \tag{34}$$

Results

We test the scenario described above using both the linear and the nonlinear dynamical models. For both models, we evaluate the impact of adding control constraints. The set of parameters used for the experiments is presented in Table 1. This table also presents the total number of LQR iterations required to solve each problem. The results are presented in Figures 2, 3, 4, 5. For further details regarding the parameters used, along with the implementation of the algorithm, we refer to the source code which is available on GitHub.

In the unconstrained case, for both the linear and the nonlinear dynamical model in Figures 2, 4, we observe an impulsive control trajectory. The control commands are focused on the first and the last time step of the sequence. This is typical of L1 cost minimization, however, these control trajectories are, in most cases, not feasible as the satellite RCS has a finite maximum thrust. By adding a constraint on the applicable control we obtain a bang-off-bang control trajectory in both the linear and nonlinear cases, see Figures 3 and 5. This bang-off-bang behavior is desirable because it is suitable to be executed by an RCS system and it minimizes fuel consumption.

Table 1. Optimization Problem Parameters and Performances

Problem Type	Linear unconstrained	Linear constrained	Nonlinear unconstrained	Nonlinear constrained
Horizon (h)	1.7	1.7	1.7	1.7
Node Points (s)	100	100	100	100
α	1	1	1	1
Q_f	1e3I	1e3I	1e3I	1e3I
ρ	1e-2	1e-1	1e-1	1e1
ϵ	4.5e-3	4e-4	6e-5	5e-6
Number LQR passes	86	404	120	206

CONCLUSION

This study presents an iterative algorithm to solve optimal control problems with L1 cost. The algorithm relies on ADMM to decompose an optimization over a nonsmooth cost function into a sequence of optimization problems with smooth cost functions. When the dynamics of the system is linear, convergence to a global minimum is guaranteed. Moreover, each optimization subproblem of the sequence is an LQR problem, which has a closed-form solution. This enables the algorithm to converge rapidly to a solution. In the case of nonlinear dynamics, the sequence of problems with smooth cost function is solved efficiently using a fast, robust iterative LQR solver with low-memory footprint.[7] These strengths allow for implementation in flight software on resource-constrained computing hardware. The spacecraft rendezvous problem inspired from the Pathfinder for Autonomous Navigation mission developed in this paper opens the way to applications in astrodynamics. For instance, the algorithm's implementation onboard a CubeSat for embedded trajectory optimization could enable rendezvous maneuvers for this type of spacecraft. Our implementation of the L1 cost optimizer is available at `https://github.com/RoboticExplorationLab/L1CostOptimizer.jl`.

REFERENCES

[1] L. S. Pontryagin, *Mathematical theory of optimal processes*. Routledge, 2018.

[2] L. Bako, D. Chen, and S. Lecoeuche, "A numerical solution to the minimum-time control problem for linear discrete-time systems," *arXiv preprint arXiv:1109.3772*, 2011.

[3] I. Ross, "How to find minimum-fuel controllers," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, p. 5346.

[4] G. Vossen and H. Maurer, "On L1-minimization in optimal control and applications to robotics," *Optimal Control Applications and Methods*, Vol. 27, No. 6, 2006, pp. 301–321.

[5] M. Grant, S. Boyd, and Y. Ye, "CVX: Matlab software for disciplined convex programming," 2009.

[6] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, Vol. 3, No. 1, 2011, pp. 1–122.

[7] T. Howell, B. Jackson, and Z. Manchester, "ALTRO: A Fast Solver for Constrained Trajectory Optimization Problems," *(submitted) Conference on Intelligent Robots and Systems (IROS)*, 2019.

[8] A. W. Koenig, T. Guffanti, and S. DAmico, "New state transition matrices for spacecraft relative motion in perturbed orbits," *Journal of Guidance, Control, and Dynamics*, Vol. 40, No. 7, 2017, pp. 1749–1768.

[9] W. Li and E. Todorov, "Iterative Linear Quadratic Regulator Design for Nonlinear Biological Movement Systems," *Proceedings of the 1st International Conference on Informatics in Control, Automation and Robotics*, Setubal, Portugal, 2004.

[10] Z. Manchester and S. Kuindersma, "Derivative-free trajectory optimization with unscented dynamic programming," *2016 IEEE 55th Conference on Decision and Control (CDC)*, IEEE, 2016, pp. 3642–3647.

[11] S. Jackson, "NASA Announces Ninth Round of Candidates for CubeSat Space Missions," *nasa.gov*, Mar 2018. `https://www.nasa.gov/feature/nasa-announces-ninth-round-of-candidates-for-cubesat-space-missions`.

[12] R. Wiltshire and W. Clohessy, "Terminal guidance system for satellite rendezvous," *Journal of the Aerospace Sciences*, Vol. 27, No. 9, 1960, pp. 653–658.

[13] H. D. Curtis, *Orbital mechanics for engineering students*. Butterworth-Heinemann, 2013.