Distributed Optimization for Multi-quadrotor Slung Load Applications
Multi-robot AA277 Final Report

Brian Jackson, Taylor Howell

March 2019

1 Introduction

Cooperation among autonomous agents has many important impacts both in society and technology. As robots penetrate further into every-day life, they are simultaneously required to solve more complex problems while being increasingly safe. Though many tasks can be accomplished by a single robot, many others benefit greatly from using a team of robots to accomplish the same task. Imagine, for example, transporting a heavy load (say a package from Amazon): you can either deploy a single large, expensive, and potentially dangerous quadrotor, or deploy a group of many smaller quadrotors to lift the same load. While the benefits of cost, versatility, safety, and deployability of the group are clear, these benefits come at the cost of simplicity: controlling a single robot is fairly straightforward, while coordinating a group of robots to cooperatively accomplish a task becomes much more complicated.

This project aims to tackle the multi-robot consensus and formation control problems through distributed optimization. Specifically, we leverage trajectory optimization for motion planning of individual agents and Alternating Direction Method of Multipliers to achieve consensus among the agents in a distributed manner. Trajectory optimization is a powerful tool for motion planning, enabling the synthesis of dynamic motion for complex and often under-actuated robotics systems. This general framework can be applied to robots with nonlinear dynamics and constraints where other motion planning paradigms—such as sample-based planning or exploiting specialized dynamics as in the case of differential flatness—are impractical. Trajectory optimization algorithms solve following optimization problem,

$$\begin{aligned} \underset{x(t),\,u(t)}{\text{minimize}} \quad & \ell(x(t_f)) + \int_0^{t_f} \ell(x(t), u(t))dt \\ \text{subject to} \quad & \dot{x} = f(x, u), \\ & g(x, u) \leq 0, \\ & h(x, u) = 0. \end{aligned} \tag{1}$$

In order to be tractable, in all but a few special cases, the infinite-dimensional continuous-time optimization problem is discretized along the trajectory and a numerical integration scheme is used to form discrete dynamics constraints,

$$\begin{aligned} \underset{x_{1:N},\,u_{1:N-1}}{\text{minimize}} \quad & \ell(x_N) + \sum_{k=0}^{N-1} \ell(x_k, u_k)dt \\ \text{subject to} \quad & x_{k+1} = f_d(x_k, u_k), \\ & g(x_k, u_k) \leq 0, \\ & h(x_k, u_k) = 0. \end{aligned} \tag{2}$$

We propose to leverage the unique problem structure of trajectory optimization for discrete robotic systems with simple interaction forces (e.g. several quadrotors carrying a slung load) to generate locally optimal trajectories for a multi-agent system. Importantly, the robots considered will be both non-linear and differential flatness will not be leveraged, so the algorithm will extend to any group of robots. This stands in contrast to much of the multi-agent control literature which considers simple single or double-integrator dynamics with independent dimensions.

In order keep the algorithm complexity linear (or better) with the number of agents, we will use Alternating Direction Method of Multipliers (ADMM) to individually solve a trajectory optimization problem for each quadrotor and then use the results to update the dual variables corresponding to the coupling constraints between them. The resulting algorithm will only require communication of each quadrotor with a centralized agent that optimizes over the collective results from the individual quadrotors. The computational complexity is constant in the number of agents thanks to the parallelization.

In Section 2 we review some prior work in this area, and in Section 3 provide some background on the trajectory optimization framework used in this project, as well as the basics of ADMM. Section 4 covers the problem formulation. Sections 5 and 6 show the algorithm in action in simulation and provide concluding remarks.

2 Literature Review

Aerial vehicles have been used to transport heavy loads since the 1960s [1]. Common applications include: delivering fire retardant to fight forest fires, carrying beams for civil infrastructure projects, moving harvested trees, carrying military vehicles, and transporting large animals. Because of their low cost, quadrotors have become a defacto testbed for such aerial vehicle applications and are typically classified in the literature as "UAVs with slung loads." Quadrotors are differentially flat, and it has been shown that quadrotors with a cable suspended load are also differentially flat which enables tracking control of either the quadrotor or load with an SE(3) controller [11]. Again, exploiting differential flatness, a Mixed Integer Program can formulated to model a hybrid system with a quadrotor and slung load that can go slack [16]. In order to perform more aggressive maneuvers a hybrid system is considered and solved using Iterative LQR (trajectory optimization) [12]. Unlike the previous work which required known trajectories for the quadrotor and load, the optimization framework generates these trajectories automatically for the full quadrotor state. Most recently, a slung load has been modeled as two revolute and one prismatic joint with a desired trajectory being found by solving a Mathematical Program with Complementary Constraints (which implicitly models the hybrid modes) [17].

Multi-quadrotor systems have also been explored. An SO(3) controller can be designed to track trajectories of an n-quadrotor lift system for a point mass [10]. This work was extended to tracking for an n-quadrotor lift system for a rigid body with finite size [14]. Both controllers model the load line as massless kinematic links and track desired trajectories for the quadrotors or load. Related to this work is a leader-follower consensus controller that avoids collisions between quadrotors in a swarm [13].

An alternative to the bottom up (ie, simple controller) framework for achieving consensus within a multi-robot system is optimization. Alternating Direction Method of Multipliers (ADMM) is a distributed optimization algorithm that breaks an optimization problem into subproblems that can be solved in parallel (eg, on separate quadrotors). These subproblems achieve consensus by the completion of the solve (to a desired tolerance) [8]. The ADMM framework has been explored for multi-agent networks with asynchronous updates [18] and trajectory planning that avoids collisions for multi-agent systems [9]. Distributed model predictive control has utilized ADMM to distribute computation across multi-robot systems to achieve formation control where the only coupling between agents is an interaction constraint [19].

3 Background

3.1 Constrained iLQR

The discrete trajectory optimization problem 2 can be solved in many ways. "Direct" methods transcribe states and inputs as decision variables and solve (2) using general-purpose nonlinear programming (NLP) solvers such as SNOPT [6] or Ipopt [7], and tend to be versatile and robust. It is straight forward to provide an initial state trajectory to the solver in such methods, even if it is dynamically infeasible. Direct transcription (DIRTRAN) [15] and direct collocation (DIRCOL) [3] are common direct algorithms.

Alternatively, "indirect" methods leverage the structure of (2) to solve a sequence of smaller sub-problems using Dynamic Programming. These are anytime algorithms, meaning they are always strictly dynamically feasible, allowing state and input trajectories at any iteration to be used in a tracking controller. However, it is often difficult to find a suitable initial guess for the control trajectory. Historically, indirect methods have been considered less robust and less suitable for reasoning about general state and control constraints but tend to be fast and amenable to implementation in embedded systems. Methods include Differential Dynamic Programming (DDP) [2] and Iterative LQR (iLQR) [5], as well as various shooting methods [20].

We recently developed ALTRO [21], an algorithm that combines the best aspects of indirect and direct methods by using iLQR within an augmented Lagrangian framework, coupled with tricks to improve numerical conditioning and robustness, provide initial state trajectories, and converge constraint satisfaction to arbitrarily tight tolerances. The result is a fast, versatile, and robust trajectory optimization solver amenable to implementation on embedded systems. We've used it to generate collision-free trajectories for systems with nonlinear dynamics, such as a quadrotor navigating a maze in Fig. 1.

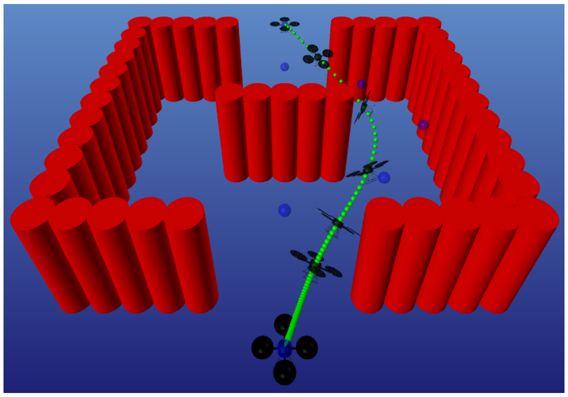

Figure 1: Quadrotor navigating a maze-like environment

3.2 Alternating Direction Method of Multipliers

Consider a convex optimization problem with a linearly separable cost function:

$$\begin{aligned} \underset{x,y,z}{\text{minimize}} \quad & f(x) + g(y) + h(z) \\ \text{subject to} \quad & Ax + By + Cz = d \end{aligned} \tag{3}$$

The augmented Lagrangian can be formed,

$$\begin{aligned} \mathcal{L}_A(x, y, z, \lambda)_\rho = f(x) + g(y) + h(z) + \lambda^T(Ax + By + Cz - d) \\ + \frac{\rho}{2}||Ax + By + Cz - d||_2^2. \end{aligned} \tag{4}$$

And the (sequential) ADMM updates for the primal and dual variables are as follows,

$$x^{k+1} = \arg\max_x \mathcal{L}_A(x, y^k, z^k) \tag{5a}$$

$$y^{k+1} = \arg\max_y \mathcal{L}_A(x^{k+1}, y, z^k) \tag{5b}$$

$$z^{k+1} = \arg\max_z \mathcal{L}_A(x^{k+1}, y^{k+1}, z) \tag{5c}$$

$$\lambda^{k+1} = \lambda^k + \rho(Ax^{k+1} + By^{k+1} + Cz^{k+1} - d). \tag{5d}$$

4 Problem Formulation

We develop a controller for a system of b quadrotors carrying a slung load. Each quadrotor is modeled with thirteen-dimensional state dynamics (i.e., quaternion representation) and the load is modeled as a point mass.

4.1 Dynamics

The first step to developing the control algorithm is to define the dynamics of the joint systems. We assume the following dynamics model for the quadrotor,

$$\dot{x} = v \tag{6}$$
$$\dot{q} = \frac{1}{2} q \omega \tag{7}$$
$$\dot{v} = \frac{1}{m}(R_q F + F_{ext}) + g \tag{8}$$
$$\dot{\omega} = J^{-1}(\tau - \omega \times J\omega) \tag{9}$$
$$\dot{z} = (\dot{x}, \dot{q}, \dot{v}, \dot{\omega})^T, \tag{10}$$

where $x \in \mathbf{R}^3$ is the position, $v \in \mathbf{R}^3$ is the velocity, $q \in \mathbf{R}^4$ is the orientation represented as a quaternion, and $\omega \in \mathbf{R}^3$ is the angular velocity. $F \in \mathbf{R}^3$ and $\tau \in \mathbf{R}^3$ are the forces and torques from the motors, expressed in the body frame, and F_{ext} are the external forces. The double integrator dynamics are modeled as,

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{m}(u + F_{ext}) + g \end{bmatrix}. \tag{11}$$

Working from the framework given in [4], we write down the joint dynamics by simply treating the cables as external forces, and subsequently constraining the forces to be equal when we solve the problem. It is important that the dynamics of each the agents be independent, otherwise minimizing over the control inputs for one agent will affect the control and input trajectories for the other agents during the iLQR solve, which enforces strict dynamic feasibility at each iteration by simulating forward the trajectory. In the case of the team lift problem, this is trivially accomplished by augmenting the control inputs with the force from the cable. The dynamics for the mass are augmented with b 3-D forces, $f^{(m,i)}$, corresponding to the equal and opposite $f^{(i)}$ acting on quadrotor i, which is inserted into the dynamics as F_{ext}. The number of states and control inputs for the joint system is therefore $n = 6{+}13b$, $m = 3b{+}(4{+}3)b$, respectively.

4.2 Optimization Problem

As mentioned previously, our approach to solving this problem is to cast it as a trajectory optimization problem of the form of (2). We define a separable cost function with individual dynamics and joint constraints between each quadrotor and the mass,

$$
\begin{aligned}
\underset{u_{1:N-1}}{\text{minimize}} \quad & \ell_f^{(m)}(x_N^{(m)}) + \sum_{i=1}^{b} \ell_f^{(i)}(x_N^{(i)}) \\
& + \sum_{k=0}^{N-1} \left[\ell^{(m)}(x_k^{(m)}, u_k^{(m)}) + \sum_{i=1}^{b} \ell^{(i)}(x_k^{(i)}, u_k^{(i)}) \right] \\
\text{subject to} \quad & x_{k+1}^{(i)} = f^{(i)}(x_k^{(i)}, u_k^{(i)}),\ i = 1, \dots, b, \\
& x_{k+1}^{(m)} = f^{(m)}(x_k^{(m)}, u_k^{(m)}), \\
& f^{(m,i)} = f^{(i)},\ i = 1, \dots, b, \\
& \left\| x^{(i)} - x^{(m)} \right\| = d^{(i)},\ i = 1, \dots, b,
\end{aligned}
\tag{12}
$$

where,

$$
\ell(x, u) = \frac{1}{2}(x - x_f)^T Q (x - x_f) + \frac{1}{2} u^T R u \tag{13}
$$

$$
\ell_f(x, u) = \frac{1}{2}(x - x_f)^T Q_f (x - x_f), \tag{14}
$$

$x_k^{(i)}$, $u_k^{(i)}$ are the state and control input vectors for agent i at time step k, and $x_k^{(m)}$, $u_k^{(m)}$ are the state and control input vectors for the load. Written without superscripts, x_k, u_k refer to the joint state and input vectors. The first and second constraints enforce the dynamics, the third enforces Newton's 3rd law for the cables (equal and opposite forces), and the last constraint specifies the length of the cable between the load and agent i, $d^{(i)}$. Note that in this formulation we assume the cables are always taunt, which is a fair assumption when working with 3 quadrotors. This assumption will be relaxed in future work.

4.3 Solving with ADMM

With a separable cost function and independent dynamics, we are ready to solve the problem using ADMM. The standard method for ADMM uses sequential updates as shown in (5). The primal updates can also be done in parallel, which reduces the time complexity of the algorithm but discards some of the theoretical guarantees of the sequential version. Algorithms for the slung load problem using both sequential and parallel updates are given in Algorithm 1 and Algorithm 2, respectively.

5 Results

In this section we present exploratory results using Double Integrator dynamics before demonstrating the use of ADMM to solve the slung load problem for full quadrotor state dynamics.

Algorithm 1 Solve ADMM

function ADMM_STEP(x)
  Initialize $X^{(m)}, U^{(m)}, X^{(i)}, U^{(i)}$
  for i=1:b **do**
    Update agent i: $X^{(i)}, U^{(i)}$ using ILQR and most recent joint info
  end for
  Update mass: $X^{(m)}, U^{(m)}$ using ILQR and most recent joint info
  Update joint Lagrange multipliers
  Update joint penalty term
end function

Algorithm 2 Solve parallel ADMM

function ADMM_PARALLEL_STEP(x)
  Initialize $X^{(m)}, X^{(m)}, X^{(i)}, X^{(i)}$ for all i
  Initialize $X^{(i)}, U^{(i)}$ on each agent **in parallel**
  for i=1:b **do in parallel**
    Update agent i: $X^{(i)}, U^{(i)}$ using ILQR
    Send $X^{(i)}, X^{(i)}$
  end for
  Update mass: $X^{(m)}, U^{(m)}$ using ILQR
  Update local Lagrange multipliers
  Update local penalty term
  for i=1:b **do in parallel**
    Send: $X^{(m)}, U^{(m)}$ to agent i
    Update local Lagrange multipliers
    Update local penalty term
  end for
end function

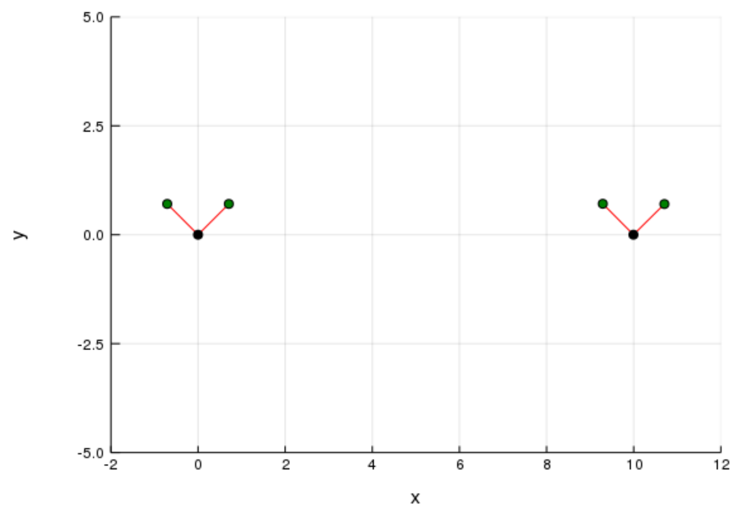

Figure 2: Initial position (left) and final position (right) of 2 Double Integrators and 1 Mass system.

5.1 Two Double Integrators and One Mass System in 2D

We use cost functions (13), (14) with $Q = 10^{-3}I$, $R = 10^{-3}I$, $Q_f = I$ for the Integrators and $Q = 0_{6\times 6}$, $R = 0.1I$, $Q_f = 1000I$ for the Mass, with implicit Euler integration, a horizon of $t_f = 1.0$s, time step $dt = 0.1$s, and cable length of $d = 1$. Sequential ADMM solves the problem in 0.76s and "parallel" ADMM solves it in 2.56s. For reference, our solver ALTRO solves the joint problem in 0.16s. For this paper, we did not implement a parallelized update but rather ran Algorithm (2) sequentially, mimicking the same behavior, but without the performance boost of parallelization. Both successfully move the mass 10 units in the x-direction see Fig. 2. The convergence rates for sequential and parallel ADMM are shown in Fig. 3

5.2 Three Double Integrators and One Mass System in 3D

The problem is extended to three dimensions and three integrators carrying a single point mass load. We use cost functions (13), (14) with $Q = 10^{-2}I$, $R = 10^{-4}I$, $Q_f = 1000I$ for the Integrators and $Q = 10^{-2}I$, $R = 10^{-6}I$, $Q_f = 1000I$ for the mass, with implicit Euler integration, a horizon of $t_f = 1.0$s, time step $dt = 0.1$s, and cable length of $d = 1.63$. Sequential ADMM solves the problem in 2.58s and "parallel" ADMM solves it in 18.44s. Both successfully move the mass 10 units in the x-direction see Fig. 4. The convergence rates for sequential and parallel ADMM are shown in Fig. 5. For reference, our solver ALTRO solves the joint problem in 0.117s (note that this is faster than the two integrator system above, this is likely due to different cost functions).

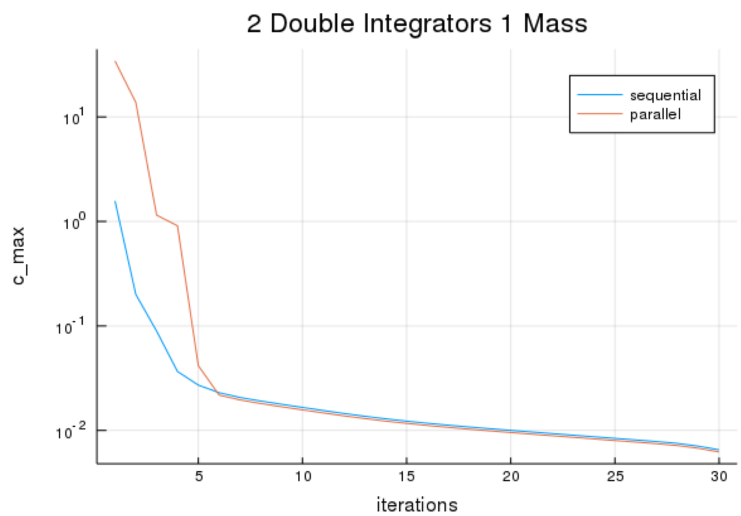

Figure 3: Maximum constraint violation during solve for 2 Double Integrators and 1 Mass system.

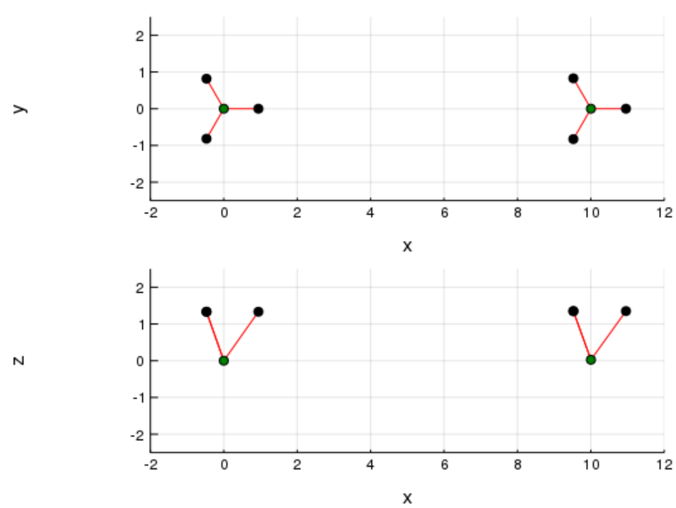

Figure 4: Initial position (left) and final position (right) with xy (top) and xz (bottom) views of 3 Double Integrators and 1 Mass system.

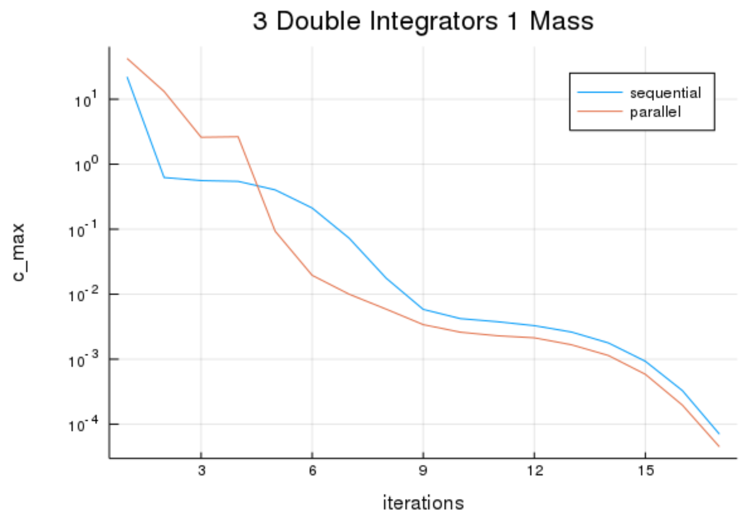

Figure 5: Maximum constraint violation during solve for 3 Double Integrators and 1 Mass system.

5.3 Three Quadrotors One Mass in 3D

Finally, we solve the slung load problem using thirteen-dimensional quadrotor state dynamics. We use cost functions (13), (14) with $Q = 10^{-2}I$, $R = 10^{-4}I$, $Q_f = 1000I$ for the quadrotors and $Q = 10^{-2}I$, $R = 10^{-5}I$, $Q_f = 10000I$ for the mass, with third-order Runge Kutta integration, a horizon of $t_f = 5.0$s, time step $dt = 0.35$s, and cable length of $d = 1.63$. Sequential ADMM solves the problem in 6.52s and successfully moves the mass 10 units in the x-direction. For reference, our solver ALTRO solves the joint problem in 18.19s. See Fig. 6 for a 2D representation, Fig. 8 for a 3D visualization, and Fig. 9 for a collection of snapshots of the system along its joint trajectory. The convergence rate for sequential ADMM is shown in Fig. 7. Attempting to solve this system with the parallel updates didn't converge, and is left as future work.

6 Conclusion

We have demonstrated that it is possible for multi-agent systems comprised of individual agents with complex dynamics to be coordinated in a useful fashion by formulating the joint problem in an ADMM framework. Further, we present initial results demonstrating that the slung load problem can be solved in a parallel fashion. We find that for simple dynamics (e.g., double integrators) solving the joint problem is often faster than using ADMM. However, the system with full quaternion dynamics (sequential) ADMM solved the problem approximately three times faster. Future work includes a more careful implementation to achieve faster runtime performance, including actual parallelization, and a

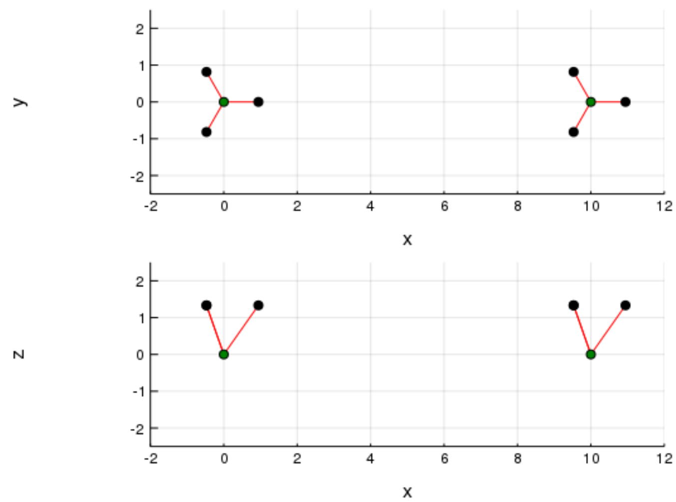

Figure 6: Initial position (left) and final position (right) with xy (top) and xz (bottom) views of 3 Quadrotors and 1 Mass system.

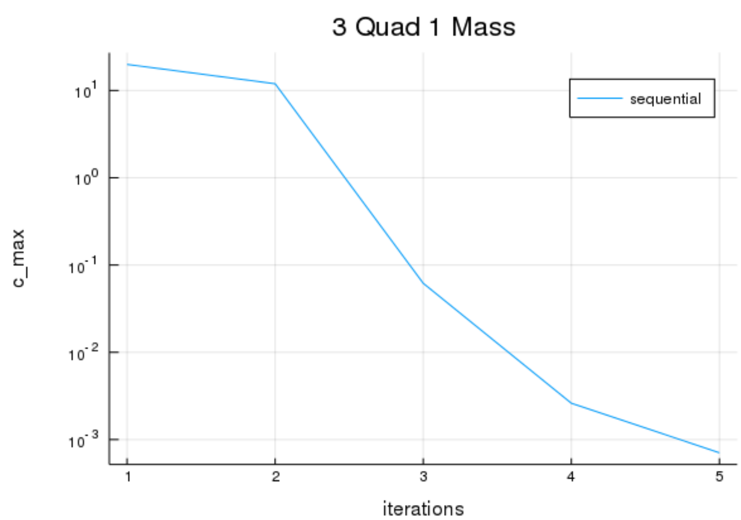

Figure 7: Maximum constraint violation during solve for 3 Quadrotors and 1 Mass system.

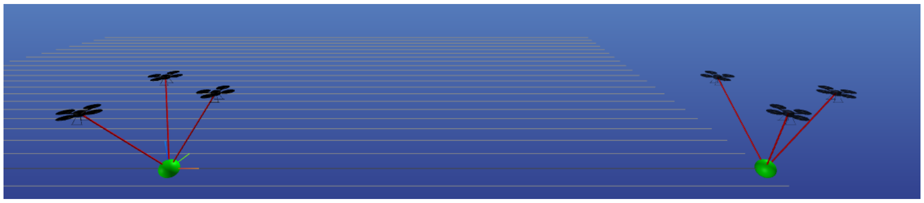

Figure 8: 3D visualization of 3 Quadrotor and 1 Mass system moving from initial position (left) to goal (right).

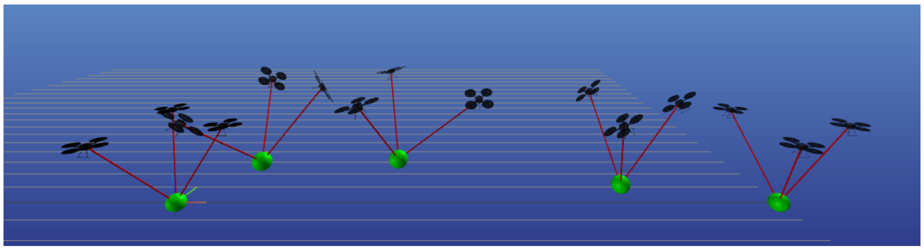

Figure 9: Collection of 5 snapshots along 3 Quadrotor 1 Mass system trajectory.

demonstration on hardware. The current implementation only considers coupling constraints between agents but it should be straight-forward to consider constraints on individual agents (e.g., thrust limits or obstacles) and handle these during the individual solves using ALTRO for constrained minimization instead of solving an unconstrained problem with iLQR. A final avenue of research is exploring techniques for a completely decentralized outer loop update such that no central coordination is required. Our Julia implementation can be found at `https://github.com/RoboticExplorationLab/TrajectoryOptimization.jl/tree/ADMM`.

References

[1] T. Lancashire, R. T. Lytwyn, G. Wilson, and D. Harding, "Investigation of the mechanics of cargo handling by aerial crane-type aircraft," BOEING CO MORTON PA VERTOL DIV, Tech. Rep., 1966.

[2] D. Q. Mayne, "A second-order gradient method of optimizing non- linear discrete time systems," *Int J Control*, vol. 3, p. 8595, 1966.

[3] C. R. Hargraves and S. W. Paris, "Direct Trajectory Optimization Using Nonlinear Programming and Collocation," *J. Guidance*, vol. 10, no. 4, pp. 338–342, 1987.

[4] D. Baraff, "Linear-time dynamics using lagrange multipliers," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM, 1996, pp. 137–146.

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

[6] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP Algorithm for Large-scale Constrained Optimization," *SIAM Review*, vol. 47, no. 1, pp. 99–131, 2005.

[7] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," en, *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, Mar. 2006, ISSN: 0025-5610, 1436-4646. DOI: 10.1007/s10107-004-0559-y.

[8] 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, pp. 1–122, 2011.

[9] J. Bento, N. Derbinsky, J. Alonso-Mora, and J. S. Yedidia, "A message-passing algorithm for multi-agent trajectory planning," in *Advances in neural information processing systems*, 2013, pp. 521–529.

[10] T. Lee, K. Sreenath, and V. Kumar, "Geometric control of cooperating multiple quadrotor uavs with a suspended payload," in *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, IEEE, 2013, pp. 5510–5515.

[11] K. Sreenath, T. Lee, and V. Kumar, "Geometric control and differential flatness of a quadrotor uav with a cable-suspended load.," in *CDC*, Citeseer, 2013, pp. 2269–2274.

[12] C. De Crousaz, F. Farshidian, and J. Buchli, "Aggressive optimal control for agile flight with a slung load," in *IROS 2014 Workshop on Machine Learning in Planning and Control of Robot Motion*, Citeseer, 2014.

[13] Y. Kuriki and T. Namerikawa, "Consensus-based cooperative formation control with collision avoidance for a multi-uav system," in *American Control Conference (ACC), 2014*, IEEE, 2014, pp. 2077–2082.

[14] T. Lee, "Geometric control of multiple quadrotor uavs transporting a cable-suspended rigid body," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*, IEEE, 2014, pp. 6155–6160.

[15] D. Pardo, L. Möller, M. Neunert, A. W. Winkler, and J. Buchli, "Evaluating direct transcription and nonlinear optimization methods for robot motion planning," pp. 1–9, Apr. 2015.

[16] S. Tang and V. Kumar, "Mixed integer quadratic program trajectory generation for a quadrotor with a cable-suspended payload," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, IEEE, 2015, pp. 2216–2222.

[17] P. Foehn, D. Falanga, N. Kuppuswamy, R. Tedrake, and D. Scaramuzza, “Fast trajectory optimization for agile quadrotor maneuvers with a cable-suspended payload,” in *Robotics: Science and Systems*, 2017, pp. 1–10.

[18] S. Kumar, R. Jain, and K. Rajawat, “Asynchronous optimization over heterogeneous networks via consensus admm,” *IEEE Transactions on Signal and Information Processing over Networks*, vol. 3, no. 1, pp. 114–129, 2017.

[19] R. Van Parys and G. Pipeleers, “Distributed mpc for multi-vehicle systems moving in formation,” *Robotics and Autonomous Systems*, vol. 97, pp. 144–152, 2017.

[20] H. B. Keller, *Numerical methods for two-point boundary-value problems.* Courier Dover Publications, 2018.

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