An Efficient Algorithm for Optimal Trajectory Generation for Heterogeneous Multi-Agent Systems in Non-Convex Environments

D. Reed Robinson, Robert T. Mar, Katia Estabridis, and Gary Hewer

***Abstract*—Generating optimal trajectories for many vehicles in real-time becomes extremely challenging when including realistic dynamic models and time-varying obstacle constraints. We present a novel method to efficiently produce time-optimal collision-free trajectories in complex non-convex maze-like environments while enforcing nonlinear constraints on velocity, acceleration, jerk, and snap. The approach combines two complementary numerical techniques for optimal control: level set reachability analysis and pseudospectral orthogonal collocation. Applied in a multi-agent prioritized planning framework, the methodology allows for heterogeneous sizes, dynamics, endpoint conditions, and individualized optimization objectives, while achieving linear scaling in computation time relative to the number of vehicles. Simulation examples are shown with up to 26 static obstacles and 32 quadcopters in a tightly constrained space with average computation times of 1–3.5 s per vehicle, depending on scenario complexity.**

***Index Terms*—Motion and path planning, optimization and optimal control, path planning for multiple mobile robots or agents, collision avoidance.**

I. Introduction

A. Motivation

PATH planning for multiple robots with many degrees of freedom is challenging due to the curse of dimensionality. When differential constraints imposed by the physical dynamics and limited control inputs of the robot are also considered, planning becomes even harder and at times intractable. Trajectory planning for time critical applications requires simultaneous optimization in time and space resulting in a challenging solution space. Obstacle dense and maze-like environments further increase the difficulty of the problem. Methods that work with sparse or convex obstacle geometry can fail when environments become too complex. Moving obstacles add another level of complexity even when their motion is known at planning time. Thus, achieving theoretically optimal cost in practically useful computation time is an ongoing area of research.

Simultaneous multi-agent trajectory planning can become intractable because its computational complexity can grow exponentially with the number of agents. Decoupling methods are often used when dealing with large teams to divide the dimensionality into sub-problems in exchange for reduced performance [1]. Teams composed of heterogeneous agents impose constraints unique to each platform, thus requiring algorithms capable of incorporating a flexible framework. Nevertheless, useful results can still be obtained when some of the above requirements are relaxed for a variety of applications. Planning can be made tractable by limiting the number of robots, requiring homogeneity, reducing solution quality, allowing long off-line computation times, limiting environment complexity, requiring time-invariant obstacles, sacrificing time-optimality for a fixed-time minimum effort problem, ignoring control saturation, replacing nonlinear constraints with conservative linear constraints, or planning with infeasible simplified dynamics and post-processing to restore real-world feasibility. At the same time there are many other applications where multi-robot systems need to or are desired to operate at optimal performance. Examples include the use of multi-agent teams to aid in search and rescue operations after a disaster, to deliver supplies, or to search an area as quickly as possible. Time-critical performance of a team of quadcopters is representative of many robotics applications, and it is the focus of this work.

B. Related Work

Direct trajectory optimization using variational methods provides a methodology to search for optimal trajectories. Variational methods discretize the continuous-time state and control, which results in a static optimization problem. The choice of problem structure, discretization technique, and underlying solver distinguishes the quality and efficiency of the method. Simplified formulations are often used to avoid nonlinearities and enable solvers based on quadratic programming (QP) to work efficiently.

Impressive performance of a real quadcopter was demonstrated in [2] with planning based on linearized dynamics represented by minimum-snap piecewise polynomials that traverse a set of waypoints in fixed time with corridor constraints. In [3], a novel change of variables is used to generate time-optimal

Manuscript received September 10, 2017; accepted December 22, 2017. Date of publication January 17, 2018; date of current version February 8, 2018. This letter was recommended for publication by Associate Editor S. J. Guy and Editor N. Amato upon evaluation of the Reviewers' comments. This work was supported by The Office of Naval Research under Grant N0001416WX01332. *(Corresponding author: D. Reed Robinson.)*

The authors are with the Naval Air Warfare Center, Weapons Division, China Lake, CA 93555 USA (e-mail: david.r.robinson@navy.mil; robert.t.mar@navy.mil; katia.estabridis@navy.mil; gary.hewer@navy.mil).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/LRA.2018.2794582

trajectories with predetermined path length for a quadcopter with nonlinear dynamics and linear state and input constraints. Angular rates are limited in order to bound the trajectory following error. Both approaches focus on structured environments where predefined corridors or passages are given *a priori*.

The aforementioned research addresses the single vehicle problem with static obstacles. An adaptation of the minimum-snap method [2] allows heterogeneous teams of up to four quadcopters to avoid each other and rectangular obstacles while moving through predetermined waypoints using mixed integer quadratic programming (MIQP). The approach uses linear dynamics and time scaling of the group trajectory to trade off performance and feasibility without explicit enforcement of control saturation limits. Optimal solutions (in the fixed-time minimum-snap sense) for scenarios involving 2–4 quadcopters required over 550 seconds while lower quality solutions can be generated in under 1 second. A single vehicle with complex obstacle geometry took 35 seconds. The method is limited to small teams and few obstacles due to the computational complexity of the MIQP [4].

Alternatively, a method based on Sequential Convex Programming (SCP) has fast solve times averaging 0.31 seconds for a fixed-time minimum snap trajectory of five quadcopters with no obstacles and linear dynamics [5]. However, the method has computational complexity greater than $O(N^6)$ in the number of vehicles, and generates trajectories that are discontinuous in jerk (requiring infinite thrust). Computation times for 12 vehicles are between 25 and 150 seconds.

Despite the successful demonstration of such methods for small groups, simultaneous coordinated planning remains a challenge for large groups because it requires that the state space be augmented for each agent causing the problem space to grow with the number of agents, thus becoming intractable.

Decoupled incremental SCP handles moving obstacles and improves scalability by using sequential prioritized planning to decouple inter-vehicle constraints. Solve time for ten agents is around four seconds for the reported scenarios [6]. It shows improved tolerance for non-convex obstacle constraints compared to standard SCP, but the probability of successfully finding a solution decreases as scenario congestion increases. The trajectories are fixed-time minimum-acceleration in 2-D, have linear limits on states, and are jerk discontinuous.

Among the most powerful techniques for time optimal planning for nonlinear systems are variational methods that use general purpose nonlinear programming (NLP). Compared to traditional Euler or Runge-Kutta based collocation methods, pseudospectral (PS) methods produce smaller sparse NLP problems that allow accurate and efficient solutions without simplification of the nonlinear dynamics and constraints. Hurni [7] combines PS methods with prioritized planning for four ground vehicles to produce time-optimal trajectories using nonlinear 6-state dynamics with bounded states and inputs while avoiding static and moving obstacles. Computation times for single static obstacle scenarios are around 35 seconds with straight line initialization. Maze-like scenarios are solved but require a manually provided initial trajectory. Local optimality is proven by numerically constructing the Hamiltonian using costate estimates provided by the PS method.

The PS method in [8] shows time-optimal multi-vehicle planning with up to 4 vehicles but, like other simultaneous planning methods, scalability remains an issue as scenario complexity and number of vehicles increase. The method addresses two common challenges for PS collocation methods by enforcing collision avoidance constraints along straight lines between additional constraint enforcement points in order to 1) avoid undetected constraint violations between collocation points and 2) allow different collocation times per vehicle to avoid restriction to common start and end times for all vehicles. The latter restriction arises in simultaneous planning because of the use of a single large system state for all vehicles with common collocation points, which is not the case for the sequential planning approach.

A Projection Operator method (PRONTO) completely avoids the undetected inter-sample constraint issue by use of a variable-step continuous-time ODE solver at the core of what is essentially an infinite-dimensional Newton's method for computation of locally optimal trajectories. It handles very general nonlinear dynamics and has been applied to time optimal single-vehicle planning [9], [3] and fixed-time minimum energy multi-agent planning [10]. The multi-vehicle implementation is limited to small groups with common start and end times as a result of simultaneous planning. The minimum-time adaptation requires a change of variables that eliminates time, thus it is unable to accommodate time-dependent constraints. This prevents its application in time-optimal multi-vehicle sequential planning. The continuous-time ODE solver is less efficient than more coarsely discretized methods, thus motivating work on a discrete-time version [11].

Recent advances in PS methods and interior point NLP solvers have given rise to software (GPOPS-II) for numerical optimal control with unprecedented computational efficiency and accuracy for high-dimensional nonlinear dynamics [12]–[14]. A method developed by Fahroo and Ross for calculating costate estimates from the Lagrange multipliers of the NLP enables verification of local optimality [15]. The method directly solves minimum time problems with variable time horizon and time-dependent constraints.

A serious weakness of the PS, SCP, and PRONTO methods alike is that success and convergence rates are highly dependent on the quality of the initialization [11], especially for complex non-convex problems, where the solver must be initialized in the locally convex region of the desired solution. In crowded scenarios with many agents, or indoor scenarios with rooms and hallways creating a maze-like structure, the possibility of initialization near (and convergence to) a point of local infeasibility reduces the probability of success of these methods. In a sequential planning framework, the group success rate quickly approaches zero as the group size increases for such scenarios.

Level set (LS) methods based on reachability analysis generate high-quality time-optimal trajectories for low dimensional systems with static and dynamic obstacles. Decoupled multi-agent planning methods using LS and sequential planning are presented in [16], [17] respectively. LS methods produce globally optimal results without initialization and with computation time that is uncorrelated to the number and shape of obstacles. However, there are two main limitations: computational com-

plexity that scales exponentially with model dimension, which is a result of the lattice that must be constructed over the entire state space; and the method's accuracy, which is limited by the resolution of the grid. As a result, LS is limited to low dimensional dynamic models for trajectory generation.

C. Contributions

We demonstrate generation of time-optimal trajectories that are smooth enough to be followed by a quadcopter with finite thrust, avoid collision with time-dependent obstacles, and are not restricted to predefined corridors or waypoints. The generated trajectories exhibit reliable convergence to provably locally optimal solutions, with consistent computation times of a few seconds per vehicle without external initialization even for crowded or maze-like scenarios. This is achieved by combination of PS and LS methods, where LS provides the necessary initialization to overcome the primary weakness of the PS method.

The ability of the LS-PS planner to handle time-dependent obstacles allows for a sequential planning framework, enabling multi-agent planning that scales well for large groups and does not restrict agents to common start and end times. Using sequential planning methods that have been previously applied individually to LS, PS, and other methods [6], [7], [17], we show that unprecedented multi-vehicle results are achieved because of the unique capabilities of the underlying LS-PS techniques. The sequential LS-PS planner provides drastic improvements in group success rates compared to sequential planning with only PS, as subsequent statistics demonstrate in Section V-B for up to 30 vehicles.

It is worth noting that while our chosen PS method and underlying NLP solver are capable of very general physics-based nonlinear dynamics, our current examples use a 2-D quadruple-integrator dynamic model with conservative nonlinear bounds on derivatives to ensure feasibility in worst case conditions, resulting in conservative solutions compared to those of a physics-based model in which control inputs can be directly constrained without compromise. The focus of our current contribution is on the improved reliability of the LS-PS combined planner, while its extension to nonlinear physics-based dynamics is left for future work. The 2-D model also provides a useful test for handling traffic bottlenecks since it is more constrained than 3-D. Despite the apparent simplicity of the linear differential constraints of the quadruple-integrator, the minimum time objective creates a variable time horizon and causes a nonlinear relationship between decision variables and cost. These factors, when taken together with the number of states and nonlinear time-varying constraints, make our example problem intractable for all the methods reviewed except for the PS methods.

D. Organization

The remainder of the paper is organized as follows. Our problem formulation is provided in Section II. The combined LS-PS single vehicle planner methodology is presented in Section III, with extension to multi-vehicle planning discussed in Section IV. Planned trajectories and statistical results for multi-agent numerical examples are shown in Section V, followed by conclusions and future work in Section VI.

II. Problem Formulation

We plan for quadcopters in a 2-D Cartesian space with quadruple-integrator dynamics that are smooth enough for finite thrust when mapping to the differentially flat 3-D physical quadcopter model in [2]. We plan in the future to apply these methods to a 3-D nonlinear physics-based model that will realize performance approaching the physical capability of a real quadcopter. Currently the flat outputs $\sigma = [r_x, r_y, 0, 0]^T$ are used, where the vertical position and yaw angle are 0, and $\mathbf{r} = [r_x, r_y]^T$ is the position of the center of mass in the horizontal plane. We refer the reader to [2] for details of mapping a trajectory in the flat output space into 3-D states and inputs and for controlling a quadcopter along the trajectory. We generate a trajectory in σ by solving the following optimal control problem:

$$\begin{gathered}\mathbf{x} = [r_x, r_y, v_x, v_y, a_x, a_y, j_x, j_y]^T \in \mathbb{R}^8 \\ \mathbf{u} = [s_x, s_y]^T \in \mathbb{R}^2 \\ \left\{\begin{array}{rl} \text{Minimize } J[\mathbf{x}(\cdot), \mathbf{u}(\cdot), t_0, t_f] = & t_f - t_0 \\ \text{Subject to} \qquad \dot{\mathbf{x}} = & [v_x, v_y, a_x, a_y, j_x, j_y, s_x, s_y]^T \\ \mathbf{x}(t_0) = & \mathbf{x}_0 \\ \mathbf{x}(t_f) = & \mathbf{x}_f \\ \mathbf{x}_{\min} \leq \mathbf{x} \leq & \mathbf{x}_{\max} \\ v_x^2 + v_y^2 \leq & v_{\max}^2 \\ a_x^2 + a_y^2 \leq & a_{\max}^2 \\ j_x^2 + j_y^2 \leq & j_{\max}^2 \\ s_x^2 + s_y^2 \leq & s_{\max}^2 \\ \mathbf{0} \leq & \mathbf{h}(\mathbf{x}, t) \end{array}\right. \end{gathered} \tag{1}$$

where the state vector, $\mathbf{x}$, consists of $\boldsymbol{r}$ and its first three time derivatives: velocity, acceleration, and jerk. The control input, $\mathbf{u}$, is the fourth derivative of position, or snap.

The cost is time elapsed from start to end of flight. Although simple in concept, this creates a challenging variable time horizon and results in faster feasible flights than minimum acceleration or minimum snap flight. The examples in this letter use a fixed final time and require all vehicles to arrive in unison at their respective goals, although this method supports any combination of initial (t_0) and terminal (t_f) conditions per vehicle.

Differential constraints allow bounds on the derivatives through snap. This is critical for feasibility of the trajectory after it is mapped into 3-D physical space, since bounded snap in the flat outputs corresponds to the bounded thrust capability of a quadcopter motor.

Initial and final states, $\mathbf{x}_0$ and $\mathbf{x}_f$, derive their position components from the starting point and assigned goal point of each agent. The remaining derivatives are set to zero for our examples, such that vehicles begin and end at rest, although they may be specified to be nonzero or unspecified according to the application.

Inequality constraints include linear bounds on the state, which are used in our examples only for defining a rectangular operation space in position. The nonlinear inequality constraints

bound the magnitude of each derivative vector to limits based on physical capabilities of the quadcopter.

The vector-valued nonlinear function, $\mathbf{h}(\mathbf{x}, t) \in \mathbb{R}^{(n_{stat}+n_{dyn})}$, represents all obstacle constraints, with negative values indicating collision, where n_{stat} and n_{dyn} are the number of static and dynamic obstacles. These are modeled as the convex hull of four circles of radius R_o whose centers coincide with the four corners of a rectangle centered at $\mathbf{r_o}$ with half-widths $\mathbf{w_o} = [w_x, w_y]^T$. The signed distance, d, between a vehicle safety circle of radius R_v and the obstacle is given by

$$d = \|\max(\mathbf{0}, |\mathbf{r} - \mathbf{r_o}| - \mathbf{w_o})\| - (R_o + R_v). \quad (2)$$

Though not shown here, obstacles may also be rotated at any angle. This approach provides the flexibility to represent a variety of geometry with a single inequality constraint, including walls of any length or thickness and round or square ends. More complex non-convex shapes are constructed by overlapping multiple obstacles. Other vehicles are represented as circles, with the components of $\mathbf{w_o}$ set equal to zero.

III. Single Vehicle Trajectory Generation

We first show how the combination of LS and PS methods results in a fast, robust, optimal single-vehicle planner in many dimensions with avoidance of time-dependent obstacles.

A. Graduated Optimization

Sensitivity to initial conditions is a major weakness of iterative trajectory optimization methods and non-convex optimization techniques in general. Graduated optimization is a heuristic global optimization technique that can greatly increase success for difficult problems by solving a simplified problem whose solution lies in the locally convex region around the global optimum of a more difficult problem. This solution to the simple problem is used to initialize the search for the solution to the difficult problem. The process can be repeated with increasing difficulty until the problem of interest is solved as demonstrated by Gashler *et al.* for manifold learning [18]. Although this may lead to globally optimal solutions, a more important result for our application is that this technique can reliably generate locally optimal solutions for previously unsolved problems. In experiments with various initialization methods, we found that any trajectory that is continuously feasible in terms of obstacle constraints is sufficient for problems with static obstacles. However, problems with dynamic obstacles are sensitive to the timing of the initial trajectory, especially for highly congested scenarios. Therefore, an initialization method is required that handles differential constraints and moving obstacles and is robust to complex non-convex environments. LS methods are well matched to this task.

The simplified problem for our graduated optimization scheme (solved by the LS method) is restricted to $\mathbb{R}^2$ instead of the high order model in (1) (solved by the PS method). It shares position and velocity with the high order model, and velocities are now directly controlled. The obstacle constraints and terminal conditions are shared between the low fidelity and high fidelity formulations.

B. Level Set Method

Level Set method is a numerical technique for solving partial differential equations [19]. Hamilton-Jacobi equations are solved via the LS method by constructing a discrete signed distance function (SDF) from the reachable (goal) set. The scalar valued SDF represents the distance to the reachable set where zero defines the set boundary, negative values are contained inside the set, and positive values exist outside the set. This SDF is propagated backwards in discrete time according to the Hamiltonian, which causes the reachable set to grow until the initial state is contained. The optimal control is calculated using the gradient of the SDF as described in Section III-B2. Assuming a solution exists and the state space is sufficiently resolved, this algorithm is guaranteed to find the global minimum.

1) Resolution Versus Accuracy: Typically, the LS solution is computed with high-order gradients and Runge-Kutta integration to minimize the numeric dissipation required for stability. The LS solution under this framework provides initialization in a graduated optimization sequence, resulting in lower accuracy requirements, which in turn allows the use of first-order gradients and integration. This and other code optimizations result in computation times that are orders of magnitude faster than the standard implementation.

The spatial resolution of the LS solution is dictated by the scenario requirements. For example, if a 1 meter vehicle needed to pass through a 1.2 meter gap, the resolution must be smaller than 0.1 meter to prevent aliasing of the gap. The temporal resolution required for numerical stability is approximately proportional to the spatial resolution; finer spatial resolutions require a finer temporal resolution during integration. Osher [19] quotes *O(N log N)* complexity for the time-invariant Fast Marching version of the LS where *N* is the number of grid nodes. The number of nodes at a given resolution grows exponentially with dimension, thus restricting the use of LS to up to four dimensions.

2) Optimal Trajectory Generation From the LS SDF: The LS method indirectly solves the Hamilton-Jacobi equation using the SDF. Post-processing is necessary to obtain the optimal state and control trajectory from that solution. Special consideration must be given to dynamic obstacles, numeric dissipation, and terminal conditions, in order to properly generate the optimal trajectory.

It can be sufficient to find the latest time at which the reachable set contains the current state and propagate in the gradient direction at the nominal maximum speed to form the trajectory for the case without dynamic obstacles. However, due to errors in the SDF that are a side effect of numeric dissipation terms, this can result in timing errors that would cause collisions with moving obstacles. The dissipation terms cause the reachable set to grow faster or slower than the nominal speed in areas of significant curvature. The trajectory becomes out of sync with the reachable set when taking full time steps at the nominal speed. This is avoided by requiring each trajectory propagation step to coincide with the reachable boundary at the next time step. In very degenerate cases, the gradient direction can also lead towards an obstacle. It is then better to sample the SDF and replace the gradient with the direction that minimizes the

SDF at the next time step. This results in trajectories that avoid obstacles earlier and more directly.

Motion planning problems may specify a goal point rather than a goal region, as is the case for (1). Standard LS methods do not directly support planning to a final point in state space. We complete the trajectory by interpolating between the point of arrival at the goal set and the desired final state.

C. Pseudospectral Method

The second and final stage in the graduated optimization approach employs a multi-interval direct pseudospectral orthogonal collocation method which generates trajectories that satisfy necessary conditions for local optimality. The continuous-time optimal control problem is time-discretized and transcribed to a finite-dimensional static optimization problem that can be solved by mature nonlinear programming software.

1) Choice of Methodology: In recent decades, extensive study of numerical methods [13], [14], [20] for optimal control has led to significant progress that contributes to the success of our method. We prefer direct methods because they do not require derivation of optimality conditions or initialization of the costate, and are more tolerant of low-quality initialization. We choose collocation over shooting for further robustness to initialization and better accuracy over long time horizons. Pseudospectral discretization methods are a major improvement over Euler or Runge-Kutta methods because the high-accuracy quadrature provides exponential (compared to polynomial) convergence rate. Other critical improvements include the computation of accurate costates for verification of optimality and efficient mesh refinement. Pseudospectral methods using orthogonal basis functions (Legendre polynomials) collocated at the polynomial roots result in sparse differentiation matrices and a sparse NLP. In an orthogonal pseudospectral method, the choice of discretization points is critical for avoiding the Runge phenomenon and defects in costate estimation. Legendre-Gauss-Radau (LGR) points are preferred over Legendre-Gauss-Lobatto (LGL) points because they provide an accurate estimate of the control and costate and are better suited to multi-interval (also known as *h p*) methods. These *h p* methods use a chain of low order polynomials instead of one global polynomial (*p* method) spanning the entire time interval [14]; this provides dramatic improvements in computational efficiency for problems requiring many collocation points. The use of many points improves obstacle detection resolution, which enables safety at high speeds and over long time horizons (see Section III-C4).

Among the robust implementations of PS methods, we select GPOPS-II [12] because it uses: *h p* methods with LGR points, accurate costate estimation, and the powerful open-source interior-point sparse NLP solver, IPOPT.

2) Initialization: The solution of the LS low order problem directly initializes the position and velocity for the high fidelity problem in (1), and the remaining derivatives are simply set to zero.

3) Collision Avoidance Constraints: The distance function in (2) has discontinuous first and second derivatives w.r.t. $\mathbf{r}$ at certain locations. We find it necessary to use a smoothing interpolation function in the obstacle interior. This effectively replaces the first derivative discontinuous region with a similarly shaped superellipse of the form used in [7]. This interpolation improved NLP efficiency and convergence without affecting geometry. It was also found to solve more quickly than with superellipse obstacles alone despite the second derivative discontinuity.

4) Safe Handling of Small Obstacles, High Speeds, and Long Time Horizon: A common challenge in collocation methods for motion planning arises when the distance between position at the collocation times (nodes) is large relative to obstacle size. This can result in trajectories that satisfy collision constraints at each node but not along the polynomial interpolated trajectory. This becomes more likely with small obstacles, high speed, and long duration trajectories. Our solution has three parts:

1) Use *h p* methods to increase node density.
2) Enforce obstacle constraints at $n_v - 1$ virtual nodes per node via interpolation between nodes, where n_v is an integer constraint enforcement multiplier.
3) Increase the vehicle safety radius by a distance that guarantees safety of the original radius for the worst case (smallest obstacle/largest node spacing).

Item 3 has the effect of dilating the obstacles until they cannot fit between the nodes.

The additional inequality constraints introduced in item 2 do not necessarily cause a significant increase in *active* constraints, just as there is only one point where a circle and sampled tangent line touch regardless of sampling resolution. In IPOPT, additional inactive inequality constraints have little effect on iteration steps and computation time. However, additional collocation points have a significant impact because the number of always-active differential equality constraints is increased. The net effect of the virtual node constraints is to enable efficient computation of faster and longer safe flights with smaller obstacles.

5) Feasibility and Local Optimality Check: Trajectories generated by PS methods can be checked for feasibility by propagating the state through the dynamic model by numerical integration according to the control input and verifying that the problem constraints and terminal conditions are met. Long time horizons and unstable dynamics may require closed loop propagation, where small corrections to the open loop controls are made by a trajectory following controller to avoid gradual divergence.

Local optimality is checked using the costate estimates to construct the time history of the Hamiltonian and verifying that it is constant as required by optimal control theory [15] (see [7] for examples). The verification may fail if too few collocation points are used, the problem has no solution, or the initialization was inadequate for convergence.

D. Multi-Start for Congested Dynamic Environments

Initialization of the PS solver via the LS method is straightforward for static and uncongested dynamic environments. As scenarios become more tightly constrained, it becomes necessary to use a value of $v_{\max}$ in the LS method that is less than

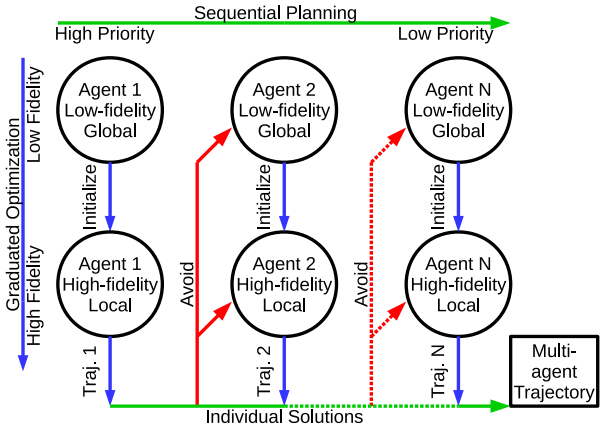


Fig. 1. Sequential graduated planning.

the limit allowed in the PS solver. The LS solution is not subject to derivative constraints beyond speed and therefore finds solutions that may require infinite acceleration.

We address this through a conservative $v_{\max}$ in the LS method to increase the chance of the existence of a feasible snap-limited solution in the neighborhood of the velocity-limited LS solution. The effectiveness can be increased by solving multiple LS-PS methods in parallel with varying $v_{\max}$ values used in the LS method. The first successful result is then selected as the solution. Note that the conservative $v_{\max}$ is used only for the LS method and does not reduce the optimality of the final PS trajectory.

IV. Multi-Vehicle Algorithm

Sequential planning trades-off full coordination for independence. Each agent's trajectory is decoupled and coordinated through the avoidance of time-varying obstacles that represent all higher priority agents [17]. The planning problem becomes a succession of near independent, low-dimension subproblems. Each subproblem can have independent start/end time and cost functional; this level of heterogeneity is impossible with simultaneous planning. The computational complexity pitfalls of coordinated planning are also avoided, since the marginal cost of adding an agent is simply that agent's subproblem.

Sequential planning imposes a strict agent priority that must be assigned *a priori* by an external entity. Prioritized planning may lead to infeasible subproblems for lower priority vehicles due to lack of consideration by higher priority vehicles, but is still capable of impressive success rates in congested scenarios as shown in Section V-B.

The combination of graduated optimization and sequential planning results in the bootstrapping architecture shown in Fig. 1. Agent 1 plans its trajectory to the required fidelity and adds it to the set of partial solutions. Agent 2 plans its own trajectory while avoiding Agent 1. This continues down the priority list of agents until the group trajectory is complete. This architecture allows for any number of agents and graduated optimization steps according to individual fidelity requirements.

V. Numerical Examples

Time optimal trajectories generated by the LS-PS method are shown for two scenarios: Fig. 2 shows 8 heterogeneous quadcopters navigating a 26 obstacle maze, and Fig. 3 shows 32 Hummingbird quadcopters crossing in traffic without obstacles. Video animations of these examples are available online at https://goo.gl/mZ1gmJ .[1]

The presented scenarios are for coordinated time of arrival while minimizing time of flight for latest departure of each agent. We use 45 collocation nodes with a virtual node multiplier of 5 for a total of 225 constraint enforcement points and a safety margin of 1 centimeter added to the vehicle safety radius of 34 centimeters. LS initializations are run at 20%, 30%, and 40% of $v_{\max}$. Derivative constraints are representative of a Hummingbird quadcopter's nominal capability: $v_{\max}$= 15 m/s, $a_{\max}$= 26.4 m/s^2, and $s_{\max}$= 2784 m/s^4. Constraints on jerk were made very large because quadcopters are capable of high pitch and roll rates that are indirectly limited by angular acceleration. The velocity limit is also large enough to be inactive for most examples.

A. Solution Quality

The example solutions have the expected smoothness, bounded derivatives, and accurately matched terminal conditions. Time of flight of individual vehicle trajectories is within 3-5 milliseconds (less than 0.2%) of locally optimal. This is based on a cost comparison with reference trajectories produced by incrementally increasing the number of collocation points while observing convergence of 1) cost, and 2) the time evolution of the Hamiltonian to a constant value as explained in Sections III-C1 and III-C5. In these examples the quadcopters are allowed to touch so that the precision of the solution is obvious. In practical scenarios, the agent's safety radius would be inflated to account for uncertainty. The quadcopters graze but do not collide with obstacles and other agents because any extra clearance would indicate missed potential time savings. High priority trajectories take a more direct path while low priority trajectories maneuver to avoid upstream agents. Trajectories are intuitive except for the case where low priority vehicles leave early to avoid congested traffic and then must loiter in order to arrive in unison. The resulting behavior can appear erratic, but it is in fact locally, if not globally, time optimal, since no later departure time would allow the agent to reach the goal in time.

B. Success Rate and Computational Efficiency

The 32 vehicle example took 112 seconds total, or an average of 3.5 seconds per vehicle. This includes both LS and PS solve times and is based on an Ubuntu Linux laptop running MATLAB 2017b on an Intel Core i7-6700K CPU @ 4.0 GHz with 64 GB of RAM. Success and runtime statistics with 100 samples per scenario were captured for empty square rooms of side length between 4 and 10 meters for up to 30 agents. The starting and ending locations were sampled uniformly, except

[1]Readers on secure networks should copy & paste the full link: youtube.com/playlist?list=PLwh1aTdBMRuelo9ZRESiwVIW4eiqrDx0s

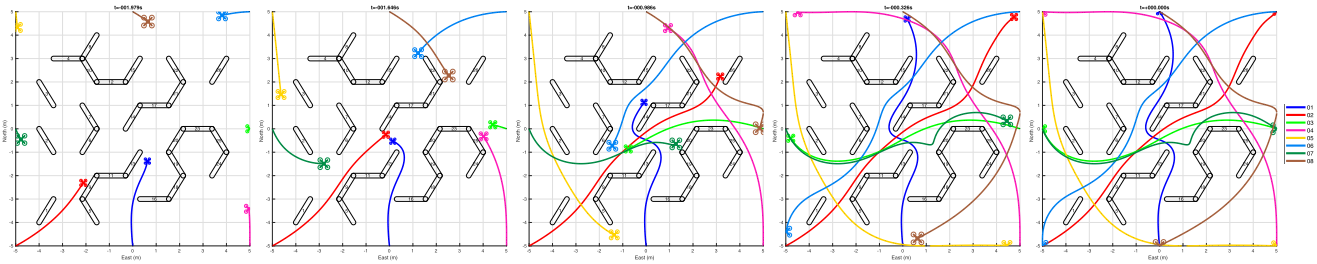


Fig. 2. 8 heterogeneous quadcopters in a maze. The largest is the size and agility of a Hummingbird quadcopter and the smallest proportionally less agile. Priority is slowest first in order of agility. Goal locations are opposite start locations.

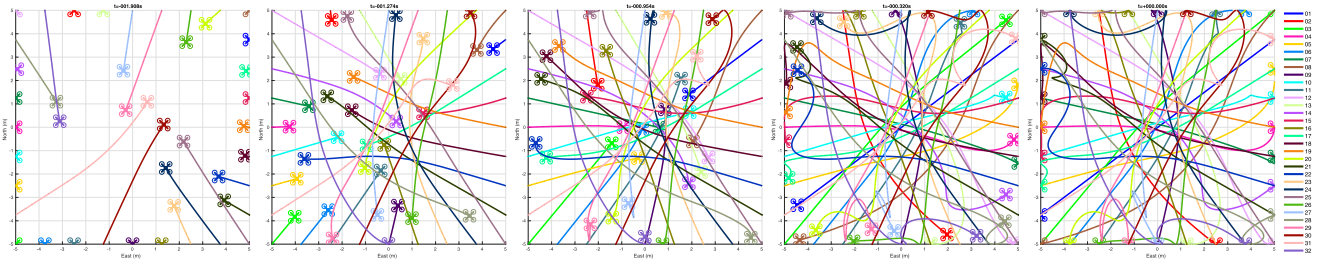
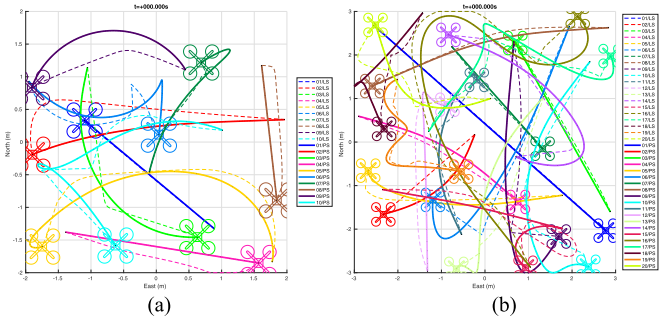
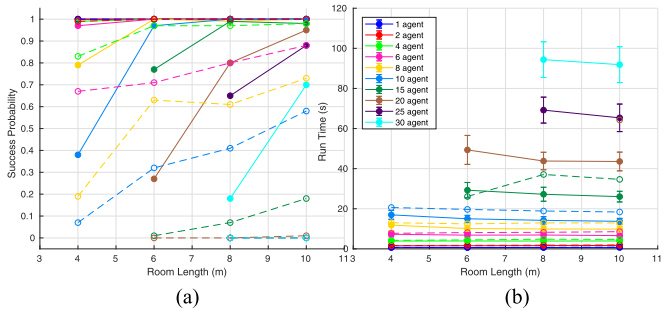


Fig. 3. 32 quadcopters in a 10 meter $\times$ 10 meter space. Trajectory snapshots range from $t = -1.908$ seconds to t $= 0$ seconds.



(a) (b)

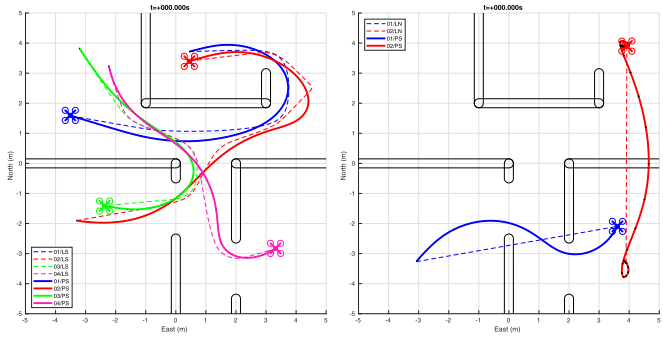
Fig. 4. Room scenario. (a) 10 agents in 4m room. (b) 20 agents in 6m room.



(a) (b)

Fig. 5. 1–30 agents in a square room. (a) Success probability. (b) Run time.

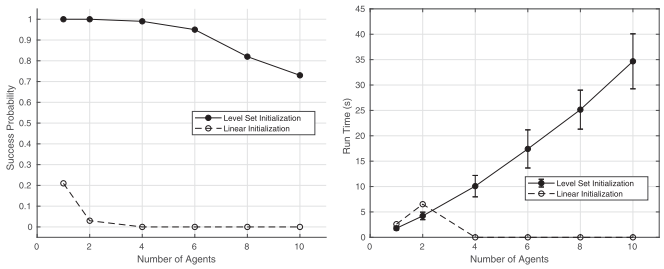
that the distance between all starting (ending) locations was required to be at least 125% of the the safe separation distance, and the ending location was required to be at least 30% of the room's diagonal distance from the starting location. These two requirements were introduced to avoid infeasible initialization and trivial solutions, respectively. Example solutions of 10 and 20 agents in rooms with side length of 4 and 6 meters, respectively, are shown in Fig. 4, with both LS and PS trajectories shown for comparison. Fig. 5(a) shows the success probability using Level Set initialization (solid line) versus trivial linear initialization (dashed lines). LS initialization results in much higher success rates than PS alone, especially for the most difficult cases. The limitations of sequential planning are apparent in the cases where the LS-PS success rate drops rapidly for very crowded scenarios. In these cases, the LS solver fails to find a feasible solution for a low priority vehicle because all previously existing solutions have been eliminated by motion of higher priority vehicles, as discussed in Section IV. Fig. 5(b) shows the total run time mean and standard deviation; the LS-PS method is faster than PS alone even with the added computational expense of the LS method.



(a) (b)

Fig. 6. Building scenario. (a) Successful LS initialization. (b) 1 successful, 1 failed linear init.

Statistics were also computed for a non-convex building-like space with 1–10 agents. Starting and ending locations were required to be at least 1 meter from the building walls. Fig. 6(a) shows a successful example scenario with 4 agents. Note that the LS initialization naturally avoids the walls, thereby allowing



(a) (b)

Fig. 7. 1–10 agents in a building. (a) Success probability. (b) Run time.

the PS optimization to find the locally optimal solution. Fig. 6(b) shows a rare success case for straight line initialization in blue, and a common failure case in red, where the local gradient information did not have enough global information to allow the solver to escape the local infeasibility. Fig. 7(a) shows the success probability using LS initialization (solid line) versus linear initialization (dashed lines) with LS-PS success rates ranging from 72% to 98% higher than PS alone. Clearly, rich initialization is absolutely critical for trajectory generation in maze-like environments. LS-PS success rates less than 100% are due to limitations inherent in sequential planning. Fig. 7(b) shows the total run time mean and standard deviation indicating complexity that scales slightly greater than linearly with the number of vehicles. Note that, since there were no successes (out of 100 samples) using linear initialization with 4 or more agents, these are indicated with zero run time.

VI. Conclusion

The proposed LS-PS trajectory planning method produces time optimal trajectories for high-dimensional dynamics with nonlinear constraints in congested non-convex scenarios for a given priority sequence. Reliably solving problems of this level of difficulty in a few seconds per vehicle in such crowded scenarios represents significant progress in single agent and multi-agent motion planning. The resulting success rates for the presented scenarios are beyond what has been demonstrated in the current literature not because of any novelty in the sequential planning methodology but because of the unique synergism of the underlying LS-PS trajectory planner. We have focused the work on time optimal trajectories; however, the presented framework is flexible enough to support other optimization objectives such as maximizing efficiency or range of flight.

We plan to extend this work to 3-D physics-based dynamics in preparation for flight demonstration.

References

[1] Y. Kuwata and J. P. How, "Cooperative distributed robust trajectory optimization using receding horizon milp," *IEEE Trans. Control Syst. Technol.*, vol. 19, no. 2, pp. 423–431, Mar. 2011.

[2] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 2520–2525.

[3] S. Spedicato and G. Notarstefano, "Minimum-time trajectory generation for quadrotors in constrained environments," *IEEE Trans. Control Syst. Technol.*, vol. PP, no. 99, to be published.

[4] D. Mellinger, A. Kushleyev, and V. Kumar, "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2012, pp. 477–483.

[5] F. Augugliaro, A. P. Schoellig, and R. D'Andrea, "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 1917–1922.

[6] Y. Chen, M. Cutler, and J. P. How, "Decoupled multiagent path planning via incremental sequential convex programming," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2015, pp. 5954–5961.

[7] M. A. Hurni, "An information-centric approach to autonomous trajectory planning utilizing optimal control techniques," Ph.D. dissertation, Dept. Mech. Aerosp. Eng., Naval Postgraduate School, Monterey, CA, USA, 2009.

[8] O. Turnbull and A. Richards, "Collocation methods for multi-vehicle trajectory optimization," in *Proc. Eur. Control Conf.*, Jul. 2013, pp. 1230–1235.

[9] A. Rucco, G. Notarstefano, and J. Hauser, "An efficient minimum-time trajectory generation strategy for two-track car vehicles," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 4, pp. 1505–1519, Jul. 2015.

[10] A. J. Häusler, A. Saccon, A. P. Aguiar, J. Hauser, and A. M. Pascoal, "Energy-optimal motion planning for multiple robotic vehicles with collision avoidance," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 3, pp. 867–883, May 2016.

[11] A. P. Aguiar *et al.*, *Constrained Optimal Motion Planning for Autonomous Vehicles Using PRONTO*. Cham, Switzerland: Springer, 2017, pp. 207–226.

[12] M. A. Patterson and A. V. Rao, "GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming," *ACM Trans. Math. Softw.*, vol. 41, pp. 1–37, Oct. 2014.

[13] D. Garg, "Advances in global pseudospectral methods for optimal control," Ph.D. dissertation, Univ. Florida, Gainesville, FL, USA, 2011.

[14] M. A. Patterson and A. Rao, "Exploiting sparsity in direct collocation pseudospectral methods for solving optimal control problems," *J. Spacecraft Rockets*, vol. 49, pp. 354–377, Mar. 2012.

[15] F. Fahroo and I. M. Ross, "Costate estimation by a legendre pseudospectral method," *J. Guidance, Control, Dyn.*, vol. 24, pp. 270–277, 2001.

[16] M. Chen and C. Tomlin, "Exact and efficient hamilton-jacobi reachability for decoupled systems," in *Proc. IEEE Conf. Decis. Control*, Dec. 2015, pp. 1297–1303.

[17] M. Chen, J. F. Fisac, S. Sastry, and C. J. Tomlin, "Safe sequential path planning of multi-vehicle systems via double-obstacle Hamilton-Jacobi-Isaacs variational inequality," in *Proc. Eur. Control Conf.*, Jul. 2015, pp. 3304–3309.

[18] M. Gashler, D. Venture, and T. Martinez, "Manifold learning by graduated optimization," *IEEE Trans. Syst., Man, and Cybern., B, Cybern.*, vol. 41, no. 6, pp. 1458–1470, Dec. 2011.

[19] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces* (Applied Mathematical Sciences 153), S. S. Antman, J. E. Marsden, and L. Sirovich, Eds. New York, NY, USA: Springer-Verlag, 2003.

[20] A. V. Rao, "A survey of numerical methods for optimal control," *Adv. Astronaut. Sci.*, vol. 135, pp. 497–528, Aug. 2009.