A Progressive Motion-Planning Algorithm and Traffic Flow Analysis for High-Density 2D Traffic

Yanchao Liu

1. Introduction

As a major social function, transportation accounts for 8.7% of U.S. gross domestic product (Bureau of Transportation Statistics 2017). It also takes up a lot of precious space, which could alternatively be used for farming, mining, housing, providing room for manufacturing, entertainment, and other activities. Nowadays, the growing demand for mobility challenges the traditional modes and systems of transportation. People expect to reach places faster and in more comfortable means, and expect goods and services to be delivered within a short period of time after the need arises. Meanwhile, the rapid development of new types of vehicles, such as autonomous vehicles and unmanned aerial vehicles (UAVs; or drones), provides great opportunities for a major revamp in our transportation infrastructure. There is an impending call for new systems to provide safe, flexible, and space-efficient means of transportation.

For any service system, increasing the capacity without expanding its physical size means elevating the service density, that is, packing more value-adding operations into a limited space. High density has been a major point of innovation in areas such as material science, manufacturing, communication, agriculture, and service systems that has created tremendous social benefits. The modern computer central processing units (CPUs), high-capacity cell phone batteries, high-speed wireless networks, and even high-rise buildings are all examples of high-density artifacts that pack more value in less space. Thanks to the rapid advancements in control, wireless communication, and artificial intelligence (AI) technologies, high-density transportation is also becoming a reality. Various new concepts and pilot systems have emerged in recent years, including automated highway systems in which vehicles travel in platoons of minimal intervehicle distance to increase the overall road capacity (Bergenhem et al. 2012; Turri, Besselink, and Johansson 2017), hyperloop systems that shoot transporting pods at high frequency and high speed along a vacuum tube connecting two metropolitan centers (Musk 2013, Jenkins 2017), and flying car technology (Hetzner 2018) that multiple automakers around the world are actively exploring as a way to escape congested roads in megacities.

Shifting more transport activities into low-altitude airspace is an attractive option given today’s technological readiness. In particular, using UAVs for inner-city...
cargo delivery will provide a promising solution to the last-mile shipment problem that is costly on the ground. Battery-powered delivery drones are quiet, fast, and clean, and unlike ground-based vehicles, they are not slowed by and do not contribute to traffic congestion on roads, allowing for more efficient operations. Enabling technologies including power, control, sensing, navigation, and communication subsystems are well developed, and several prototype aerial delivery systems have already been field tested. For instance, Amazon (2018) has tested its Prime Air program, a small drone system that can deliver packages up to five pounds, in England. In 2016, 7-Eleven completed a fully autonomous drone delivery test in Reno, Nevada, the first such test in the U.S. airspace approved by the Federal Aviation Administration (Glaser 2016). In China, JD.com launched a trial program for drone delivery services, testing air drop-offs in Beijing and several provinces (Parmar 2016). Similar developments and tests are underway at Google’s, Walmart’s, and NASA’s research arms (Barr and Bensinger 2014, Malcolm and Weise 2015, National Aeronautics and Space Administration 2015).

Given the limited carrying capacity of individual vehicles, the benefit of aerial delivery systems will materialize only when a large number of vehicles is deployed. The capacity for flight for individual units is necessary, but far from sufficient for establishing a practical aerial delivery system in which fleets of drones simultaneously traverse a shared three-dimensional (3D) space in all possible directions and at various speeds. Traffic rules and protocols along with the corresponding enforcement, monitoring, and contingency response mechanisms must be set up, understood, and agreed upon by all parties involved before any deployment at scale is possible. Figure 1 illustrates the complexity of traffic flow in an open space. In a dynamic two-dimensional (2D) traffic scenario, as shown in the figure, vehicles’ on-board AI alone, that is, the ability to sense and avoid obstacles, is insufficient to support a safe and efficient traffic order. The traffic flow will be chaotic, inefficient, and conflict prone when each vehicle determines its own motion without coordination. Because of the high dimension, high density, and autonomy of this mode of transportation, neither the road network models nor the traditional air traffic control procedures can directly apply in this space. New research is needed.

In this paper, we address the problem of routing a large fleet of vehicles in highly dense and congestive traffic. The grand goal is to make all vehicles that come under dispatch reach their respective destinations quickly and efficiently while maintaining a safe intervehicle separation at all times. This infinite-horizon operational problem is formulated as a nonlinear nonconvex optimization model that must be solved in a progressive, receding-horizon fashion. To ensure feasibility and overcome path deadlocks attributed to local optima, a series of heuristic measures is developed and validated via computational experiments. The target application includes, but is not limited to, an aerial delivery system in which heterogeneous drones, each taking off at a random time and having arbitrary destinations, traverse an urban sky at a given altitude. A set of new metrics is also proposed to characterize the 2D traffic flow and space efficiency. As an integral part of an intelligent dispatch software, this work will positively contribute to the technological preparedness for bringing modern transportation to a higher density, be it airborne, ground based, marine, or submersible.

1.1. Related Literature
Mobility system planning has been actively researched in recent years. On one end of the spectrum is the “free flight” type of path planning, where only a list of waypoints is computed. It relies on the moving object’s own navigational and mobile ability to move from one waypoint to the next. Examples include GPS-based direction assistance for automobiles, vehicle routing for delivery trucks (Crevier, Cordeau, and Laporte 2007; Eksioglu, Vural, and Reisman 2009; Drex1 2012), and navigation and piloting practices in...
civil aviation (Bilimoria and Lee 2001) and remote-controlled UAV tasks (Murray and Park 2013). In these systems, human operators’ judgment, intervention, or direct action usually plays a critical role, although more and more routine operations are aided by computer systems (Hoekstra et al. 1998, McNally and Gong 2007). On the other end of the spectrum is the closed-loop end-to-end control method, determining the control inputs to drive the controlled object from a start configuration (e.g., location, orientation, speed, etc.) to a desired end configuration. Examples include planning the motion paths for robotic arms in a manufacturing process, planning paths for automatic guided vehicles in warehouses, controlling the positions and movements of spacecraft or satellites, etc. Books by Laumond (1998) and LaValle (2006) provide comprehensive technical details in robotics motion planning (MP) and control.

Conflict resolution, or collision avoidance, is the primary focus in air traffic management (Vranas, Bertsimas, and Odoni 1994; Durand, Alliot, and Chansou 1995; Kuchar and Yang 2000; Visintini et al. 2006; Cellier, Cafieri, and Messine 2013; Wu and Du 2014; Sunil et al. 2017). When multiple aircraft enter the same volume of airspace, a path conflict may threaten the loss of safe separation among aircraft. In this situation, a resolution strategy or tactic is needed to find trajectories that satisfy separation constraints while minimizing the overall trip disruption for all aircraft involved. Mathematical programming and optimization is an oft-used technique for modeling and solving this problem. The seminal paper by Pallottino, Feron, and Bicchi (2002) formulated the multiaircraft conflict-avoidance problem as a mixed integer linear program (MILP) to minimize the total flight time. Integer variables were used for modeling the forbidden cone of velocity (due to collision avoidance) by disjunctive constraints. The model essentially produced a one-shot alteration to aircraft’s speed or heading at the beginning configuration to make sure the resulting trajectories are conflict-free for the rest of the time. This strategy relied on abundant free space in the initial configuration to accommodate the maneuvers, a luxury that is typically unavailable in dense traffic. A similar treatment for collision avoidance was used by Ny and Pappas (2010), whereas the disjunctive constraints were handled by geometric programming techniques. Christodoulou and Costoulakis (2004) proposed a mixed integer nonlinear programming (MINLP) formulation for multiaircraft conflict resolution. This model was highly nonconvex because of the use of trigonometric functions and bilinear terms to express the speed and heading angle changes. Small-scale instances involving two and three aircraft were solved using optimization solvers within the GAMS modeling software. Frazzoli et al. (2001) employed a nonconvex quadratically constrained quadratic program to model the planar, multiaircraft conflict resolution problem. The overall scheme was a mix of centralized decision making for safety and decentralized preference optimization for efficiency, with a cost function chosen to minimize the deviation between desired and conflict-avoiding heading for each aircraft. The authors furthermore presented a semi-definite programming relaxation scheme and a randomized search approach for resolving various local conflict patterns. Dell’Olmo and Lulli (2003) proposed a two-level optimization model to plan airway usage schedule over a large area of airspace. The upper level was a commodity flow problem in a graph where airports were modeled as nodes, airways connecting airports were arcs, and aircraft were commodities. The lower-level model treated a single airway as a tube in which four-dimensional collision-free aircraft trajectories were computed. Although a novel approach, the model relied on discretization of many parameters. For instance, a binary variable \( x_k(t) \) was defined as “1 if aircraft \( k \) at time \( t \) is at flight level \( z \) with velocity \( v \), 0 otherwise” (Dell’Olmo and Lulli 2003, p. 184), and lived in a huge combinatorial space formed by the Cartesian product of four large discrete sets. This would inevitably present a hurdle to the model’s scalability. By discretizing time into 20-second intervals, Islami, Chaimatnan, and Delahaye (2016) formulated a large-scale MILP for planning the four-dimensional trajectories of aircraft in a nationwide and continent-scale airspace. The authors proposed a hybrid metaheuristics consisting of simulated annealing and local search algorithms to find conflict-free trajectories for thousands of aircraft. However, the solution process was time-consuming; up to 45 hours of CPU time was needed for a single run.

Apart from using centralized optimization techniques to resolve conflicts, agent-based distributed coordination methods have also been widely studied (Eby and Kelly 1999; Šišlák et al. 2007; Šišlák, Samek, and Pěchouček 2008; Pechoucek and Sislak 2009). An extensive review of this important thread of research is forgone here. Instead, the readers are referred to a recent survey paper on collision avoidance for UAVs by Mahjri, Dhraief, and Belghith (2015), in which the authors organize the collision-avoidance system into sensing, detection, and resolution functions and provide a review of literature and technologies of each function.

Path-planning algorithms are also extensively studied in the robotics literature under the term of kinodynamic motion planning (Donald et al. 1993, LaValle and Kuffner 2001, Hsu et al. 2002). For instance, Earl and D’Andrea (2007) formulated a wheeled robot control problem as an MILP, utilizing big-M constraints to facilitate the collision-avoidance constraints. Abichandani et al. (2015) proposed an MINLP
formulation to model the spline path curves for underwater vehicles, in which the cubic spline curves gave rise to nonconvex nonlinear constraints. The solution algorithm was executed sequentially so that each vehicle planed its own movement by taking as input the planned movements of other vehicles of a higher priority order. Sunil et al. (2017) generated local flight plans for solving real-time geofencing and traffic constraints based on the rapidly exploring random tree technique in discrete time. Desaraju and How (2012) studied path planning for multiagent teams under complex constraints and introduced a cooperation strategy that allowed an agent to modify its teammates’ plans to select paths that reduce their combined cost. Cohen et al. (1995) proposed an algorithm to perform quick collision detection between pairs of objects of polyhedral shapes. It was a two-level approach based on the idea of pruning pairs of objects that were far from each other using bounding boxes. A similar exact method is also used in this paper for pruning unnecessary collision-avoidance constraints in the motion-planning model.

Most models with kinematics-level details have focused on resolving conflicts in an area separated from the rest of the traffic space. This approach is sufficient for scenarios in which the traffic density is low in most parts of the system and path conflicts, especially ones involving multiple aircraft, occur only sporadically. In high-density scenarios where congestion is a common occurrence (as illustrated in Figure 1), a segregated local approach may cause a chain reaction, in which the resolution of one conflict may lead to other immediate or imminent conflicts (Kuchar and Yang 2000; Jardin 2004). In such circumstances, it is important to adopt a holistic, global optimization framework that coordinates the entire traffic flow across a broad space. Furthermore, traffic coordinated under a coherent theme will provide insights into the airspace capacity and efficiency.

1.2. Main Contributions

In both optimization-based and agent-based robotics motion and aircraft trajectory planning models, temporal discretization is almost ubiquitously used (Richards and How 2002; Dell’Olmo and Lulli 2003; Raghunathan et al. 2004; Šišlák, Samek, and Pechouček 2008; Desaraju and How 2012; Cellier, Cafieri, and Messine 2013; Islami, Chaimatanan, and Delahaye 2016; Sunil et al. 2017). However, the loss of fidelity in using discrete-time models to approximate continuous motions has been largely unattended, perhaps because it is not a predominant concern in low-density applications where maneuvering space is abundant. In fact, collision-avoidance constraints modeled in discrete time will translate to a less secure situation in continuous time, and in the worst case, a head-on collision can happen even when the discrete-time collision-avoidance constraints are perfectly satisfied. Conversely, if the safety distance is set in an overly conservative way, the system capacity will be sacrificed. To address this issue, analytical bounds on the quality of the discrete time approximation are derived, which provide critical guidance for accurately modeling vehicle movements in a dense traffic.

It is customary in the literature to formulate the motion-planning problem as an MILP for its superior solvability over the alternative nonlinear formulations. However, linearization of kinematic laws in the Euclidean space, for example, using polygons to approximate circular sectors, may lead to a loss of accuracy, which matters in high-density scenarios but is often left unquantified. A solution architecture that is both aligned with the Euclidean distance metric and suitable for real-time applications is still lacking in the literature. This paper presents a nonlinear programming (NLP) model that authentically models vehicle motion in a 2D space, and develops an algorithm that enables uninterrupted motion planning with feasibility guarantee and high solution quality. The author is not aware of any other solution approach for similar formulations that is able to produce attractive feasible solutions for large and dense systems.

With the advent of the AI era and the prospect of mass deployment of unmanned aircraft systems (UASs) in the low-altitude airspace (Hall 2016; Ghariibi, Boutaba, and Waslander 2016), understanding traffic flow behavior in 2D (and 3D) space is critical for the safe and efficient use of the airspace. Neither traditional road-based traffic flow metrics nor the existing air traffic models are adequate for this dense, high-dimensional, and unmanned space. In this regard, a set of new metrics is proposed to quantify the trip performance, traffic flow, and system efficiency for coordinated air traffic systems. These metrics are algorithm agnostic; therefore, they can facilitate performance comparisons among a wide range of system designs, models, and implementation schemes. Some of these metrics are used for evaluating the algorithm’s performance in different density conditions and presenting insights in system design.

The model, algorithm, and metrics developed in this paper are, in theory, extensible to air traffic in the 3D Euclidean space. However, we focus on the 2D space here because (1) vertical separation of aircraft is managed differently in practice (i.e., aircrafts’ safety buffer volume is cylindrical rather than spherical; see, e.g., Lehouillier et al. 2017), as horizontal motion accounts for the dominant transportation activity, and (2) the proposed metrics will be comparable in scope to the leading air traffic flow literature in which 2D flow is the predominant focus. Extension to 3D space is briefly discussed as future work in Section 6.
1.3. Organization
The remainder of this paper is organized as follows. Section 2 derives the exact bounds for collision-avoidance constraints and proposes a spatial decomposition strategy. Section 3 presents the metrics for quantifying 2D traffic efficiency. Section 4 develops the main mathematical model and solution algorithms and discusses their properties. Section 5 reports numerical experiments and analyzes the results. Finally, Section 6 concludes this paper and mentions some future work.

2. Collision Avoidance
Safety is the top priority when it comes to moving people or goods from one place to another, and crash is the biggest threat to transportation safety. To avoid crashes, vehicles in the field must be sufficiently separated. A minimum intervehicle distance that allows for an “escape route” in case of a contingency should be maintained. The choice of the safety distance in a transportation system depends on the vehicles’ maneuverability, motion accuracy, and reliability. Setting the safety distance prudently to meet the underlying safety requirements without wasting space is an important task in the design of a transit system.

2.1. Exact Bounds for Collision Avoidance
Collision avoidance must be enforced at all times, whereas collision detection can occur only at discrete time points. This calls for a meticulous discretization scheme along the time dimension so that (1) feasibility at contiguous time points can guarantee feasibility at every time point in the interval formed by the checkpoints and (2) computational efficiency is maximally preserved, meaning that unnecessarily fine discretization should be avoided. This section will derive bounds on the accuracy of temporal discretization and present their practical usage. The notion and result will be used in subsequent sections.

Let $u_{ij} \in \mathbb{R}^n$ be the location of vehicle $i$ at time $t$, where $n$ is 2 or 3 depending on the dimension of space under study. Assume that in any unit interval of time, say, between $t$ and $t + 1$, each vehicle $i$ moves at a fixed velocity $v_{ij} \in \mathbb{R}^n$ with $||v_{ij}|| \leq R_i$, where $R_i$ is the maximum distance vehicle $i$ can travel in a unit time interval. This gives $u_{ij+1} = u_{ij} + v_{ij}$ for each $i$ and $t$. As implied by the notation, a vehicle’s velocity is assumed to be unchanged (i.e., linear movement) within the same unit interval of time, although it can vary across different time intervals. Collision avoidance is concerned with the relative position between two moving vehicles, say, $i$ and $j$. Let $\Delta_t = u_{ij} - u_{ij}$ be the relative location of $i$ to $j$ at time $t$, and let $\delta_t = v_{ij} - v_{ij}$ be the relative velocity of $i$ to $j$; then we have $\Delta_{t+1} = \Delta_t + \delta_t$. Triangle inequality indicates that $||\delta_t|| \leq R$, where $R = R_i + R_j$. When time is discretized in an algebraic modeling system, the collision-avoidance constraint takes the form $||\Delta_t|| \geq S$, $t = 0, 1, 2, \ldots$, requiring the intervehicle distance to be greater than a preset constant $S$ at each discrete time point $t$. In the literature, most work concerning collision avoidance employed this constraint or a close variant of it. However, this constraint says nothing about the intervehicle distance during the time interval. In effect, all-time separation is approximated by discrete-time separation. It is critical to understand the property of this approximation as well as its implications for safety.

This in regard, an immediate inquiry is, do $||\Delta_t|| \geq S$ and $||\Delta_{t+1}|| \geq S$ guarantee $||\Delta_{t+s}|| \geq S$, for any $s \in [0,1]$? Or, equivalently, what is the minimum value of $||\Delta_t + a\delta_t||$ for $0 \leq a \leq 1$, given $||\Delta_t|| \geq S$, $||\Delta_{t+1}|| \geq S$, and $||\delta_t|| \leq R$? The following theorem establishes bounds on $||\Delta_{t+s}||$ for the case when $|| \cdot ||$ is taken to be the Euclidean norm. A similar result with the Manhattan distance is also obtainable, which is outside the scope of this paper.

**Theorem 1.** Given constants $S, R \geq 0$, let $\Delta_t, \delta_t \in \mathbb{R}^n$ with $||\delta_t|| \leq R$ for $t = 0, 1, 2, \ldots$, and define $\Delta_s = \Delta_{t+s} + (s - [s])\delta_{[s]}$ for $s \in \mathbb{R}^+$. Then the algebraic constraints $||\Delta_s|| \geq S, t = 0, 1, 2, \ldots$ imply that the continuous-time separation distance $||\Delta_s||_2$ satisfies

$$||\Delta_s||_2 \geq \begin{cases} \sqrt{S^2 - (1/4)R^2} & \text{when } R \leq 2S, \\ 0 & \text{when } R > 2S, \end{cases}$$

for all $s \in \mathbb{R}^+$.

The proof is given in Online Appendix A. The theorem shows that the minimum separation distance stipulated in the series of discrete-time algebraic constraints does not translate to the same level of separation in continuous time, but to a reduced and hence less secure level of separation. Moreover, if the maximum closing speed between two vehicles is greater than two times the stipulated separation radius, the discrete-time separation constraints will have no effect in preventing collision in real time, and the two vehicles may collide head-on.

The loss of fidelity from substituting discrete-time constraints for continuous-time constraints is not surprising. Theorem 1 exactly quantifies this loss of fidelity and provides useful bounds on the actual separation distance. The bounds allow for an accurate calculation for the parameter values (i.e., $S$ and $R$) needed in discrete-time models. Let us look at a numerical example.

**Example 1** (Figure 2). Vehicle 1 (red) travels from coordinate (100, 100) to (100, 400), whereas vehicle 2 (blue) travels from (200, 400) to (200, 100). For convenience of exposition, let us assume the distance is measured in meters and the natural time unit is the second. The all-time separation distance is set to $S' = 140$
meters, shown as the diameter of the solid circles in the figure. Suppose both vehicles have a maximum speed of 10 meters per second. If time is discretized into 10-second intervals, then the maximum closing speed $R$ would be 200 per unit of time. Theorem 1 would require \[
S - \frac{1}{4}R^2 \geq S',
\] hence, $S$ must be set to 172.1 or above to ensure a safe separation, shown as the diameter of the dotted circle in Figure 2(c). Panel (a) shows a significant loss of separation during time 2 to 3 when only 140 meters of separation was enforced on discrete time points. Using a more fine-grained time discretization, for example, one-second intervals, as shown in panel (b), could mitigate (but not completely eliminate) the loss of accuracy, which would come with a higher computational cost. Panel (d) demonstrates temporal discretization by four-second intervals with an exact separation bound of 148.7 (dotted circle), which approximates the real situation well with security assurance.

In practice, it is more intuitive to specify the separation margin in terms of time, in which case the actual distance of separation is simply a function of speed. Let $h$ be the minimum time distance between any pair of on-route vehicles $i$ and $j$ with maximum speeds $s_i$ and $s_j$, respectively. Then the separation distance $S_{ij}$ to be enforced at discrete time points is given by

\[
S_{ij} = (s_i + s_j)\sqrt{h^2 + 0.25}.
\] (1)

This general formula leaves great flexibility for instantiating the length and time units to match different application scenarios. Given the same set of values for $h$, $s_i$, and $s_j$, for instance, in civil aviation, the length unit can be set to 0.1 nautical miles and the time unit to 5 seconds, whereas in a UAV delivery system.

Figure 2. (Color online) Can a Discrete Time Constraint Ensure Continuous Time Separation?

Notes. Two vehicles start in opposite directions along parallel paths, and the required separation distance is wider than the interpath distance. Panel (a) shows that a significant loss of separation occurs during $t = 2$ to 3. Panel (b) shows that the loss is mitigated (but not completely eliminated) by an overly fine-grained temporal discretization. Panel (c) shows that the loss is eliminated by using exact discrete-time separation bound in accordance with the temporal scheme. Panel (d) shows a finer-grained discretization with an exact separation bound.
consisting of small drones, the unit length and time can be set to 1 meter and 1 second, respectively, to adjust for the physical dimension and operating characteristics of systems.

### 2.2. Spatial Decomposition by Collision-Cautious Clusters

When the airspace under management has an uneven traffic density distribution, it is possible to dynamically decompose the motion-planning problem into smaller independent problems, one for each cluster of vehicles linked by the collision-avoidance constraints. Let \( \mathcal{C}_{t_0} \subseteq \mathcal{A} \times \mathcal{A} \) be the set of collision-cautious vehicle pairs whose separation constraint must be enforced in the planning horizon covering time periods \( t_0 \) to \( t_0 + T \). Intuitively, when two vehicles \( i \) and \( j \) are sufficiently far away from each other at the current time \( t_0 \), their separation is not an immediate concern in a model that plans vehicle motion for the next \( T \) time steps. Hence, the pair \( (i,j) \) does not need to appear in the set \( \mathcal{C}_{t_0} \) when solving the model. We will derive the distance threshold below which a vehicle pair must be included in the set \( \mathcal{C}_{t_0} \) for collision-avoidance constraints.

Consider the worst case and assume that the two vehicles travel head-to-head toward each other at maximum speeds during the entire planning horizon of \( T \) time intervals. By time \( t_0 + T \), their distance can be bridged by no more than \( T(s_i + s_j) \). If their distance is still beyond the minimum separation distance \( S \), then it is safe to conclude that their initial distance is large enough to warrant a safe separation at any time during the planning horizon. Hence, \( (i,j) \) does not need to be included in \( \mathcal{C}_{t_0} \). The reasoning is formalized in the following theorem.

**Theorem 2.** It is sufficient to set \( \mathcal{C}_{t_0} = \{ (i,j) \in \mathcal{A} \times \mathcal{A} : \|u_{i,t_0} - u_{j,t_0}\| \leq (T + \sqrt{h^2 + 0.25})(s_i + s_j) \} \). In other words, if a pair of vehicles is not included in \( \mathcal{C}_{t_0} \) determined by the above formula, then the pair is guaranteed to be at least \( h \) time units apart during the entire time period from \( t_0 \) to \( t_0 + T \).

Theorem 2 is a straightforward corollary of Theorem 1, so its proof will be omitted. It provides a conservative (large enough) distance threshold to account for any possible conflict. It does not indicate, however, that all vehicle pairs included in \( \mathcal{C}_{t_0} \) will ever enter a direct conflict. As observed in numerical experiments, many collision-avoidance constraints remain inactive in the solution process, suggesting a highly sparse problem structure.

One way to exploit the sparse structure is to decompose the motion-planning problem over the entire airspace by clusters of vehicles. Specifically, if vehicles in the airspace are modeled as nodes in a graph, and an edge is placed between nodes \( i \) and \( j \) if \( (i,j) \in \mathcal{C}_{t_0} \), then each connected subgraph will represent an independent portion of the problem that can be solved separately and in parallel. The idea is illustrated in Figure 3 and in the following observation.

**Observation 1.** The complexity of a system-wide motion-planning problem is of the same order as the complexity of planning for the densest region of the entire space.

Spatial decomposition for large air transportation networks was also exploited in Wei, Spiers, and Sun (2014), in which airports were clustered based on geographic distance. This paper focuses on solving and analyzing the traffic flow in dense and highly congestive airspace environments; therefore, the decomposition and parallel computing schemes are not implemented in the prototyping code.

### 3. Traffic Flow Metrics for Dense 2D Traffic

From a traffic manager’s perspective, planning the motion of vehicles that arbitrarily enter and utilize the space under management is a continuous, open-ended, and (in theory) infinite-horizon control problem, for which a “globally optimal” course of action is both difficult to describe and not very relevant for practice. Therefore, algorithmic performance measures alone, such as the optimality gap and solution time, fall short of providing practical insights. To amend this shortage, one needs a set of algorithm-agnostic metrics to characterize the traffic flow.
3.1. Inadequacy of Existing Frameworks

There are numerous theories, models, and empirical studies to characterize traffic flow in road transport systems; see, for example, Wardrop (1952), Gerlough and Huber (1975), Anderson (1978), Ceder (2007), and Lieu (1975). However, the models and findings do not apply to 2D and 3D scenarios. The added dimension(s) of freedom dramatically increase the traffic complexity. On a road segment, effective separation among vehicles is straightforward to implement by simple rules, for instance, keeping the following distance vehicles is straightforward to implement by simple rules, for instance, keeping the following distance above a minimum threshold will do the job, and this can be performed instinctively by drivers. Many traffic models take this instinctive collision-avoiding behavior as a fundamental assumption, based on which theoretical and observational characterizations of the system are then derived. In contrast, when vehicles traverse a 2D space in all possible directions, maintaining effective separation becomes much more complicated, and no simple rule will work without significantly sacrificing capacity and efficiency. Furthermore, conventional definitions of speed, flow, and density for fixed-route systems will no longer work in characterizing high-dimensional systems. For example, to avoid collision, an aerial vehicle may travel in a direction unaligned with, or even opposite to, the desired destination-pointing direction. In this case, high speed does not translate to high efficiency. Similarly, if most vehicles are constantly maneuvering because of pathway congestion, the overall corridor efficiency is actually quite low even if speed and density are both at high levels.

Existing experience in air traffic management is also inadequate to provide a full understanding of the traffic behavior. First, traditional air traffic management deals with open-loop, manned systems in which pilots’ skill and judgment, as well as the air-to-ground and air-to-air communication protocols, play a major role in safe operations (Federal Aviation Administration 2009, Lehouillier et al. 2014). In an unmanned system, these elements will become obsolete. Second, the traffic density is usually quite low for the most part in a long-haul air trip. In contrast, a practical low-altitude, short-range aerial delivery system is expected to have much denser traffic, because of the limited flight range and carrying capacity of each vehicle in the system. Third, the airport-to-airport flight routes are almost fixed for bulk air transportation, which reduces the airspace management (two- or three-dimensional) problem to a corridor management (one-dimensional, spatially) problem (Vranas, Bertsimas, and Odoni 1994; Dell’Olmo and Lulli 2003; Bertsimas, Lulli, and Odoni 2011) or to a network optimization problem (Ma, Cui, and Cheng 2004; Wei, Andrisani, and Sun 2011; Yang, Mao, and Wei 2016). In contrast, the origin and destination (OD) and flight plan of a delivery drone can be highly variable. For instance, a lightweight drone can be launched from the top of a delivery truck, which serves as the moving depot and battery stop for the drone (Murray and Chu 2015).

For air traffic flow modeling, Eulerian network modeling (Menon, Sweriduk, and Bilimoria 2004) has been a particularly popular approach. One-dimensional Eulerian models aggregate air traffic into line segments (called links) and use partial differential equation–based difference equations to drive the flow dynamics (Sun, Strub, and Bayen 2007). Two-dimensional models (Menon et al. 2006) partition the airspace into cells and study traffic flows on the cellular network. By imposing such structures, Eulerian models are able to attain remarkable tractability even in continent-scale traffic analyses, and have been used for, for example, density prediction and flow optimization (Bayen, Raffard, and Tomlin 2006; Work and Bayen 2008). Nonetheless, in dense UAS traffic, the vehicle motion cannot be simply driven by difference equations because of nonconvex collision-avoidance constraints and arbitrary new vehicle entrances. Therefore, traffic flow must be measured independent of the underlying equations of motion (EOMs) or planning algorithms.

3.2. New Metrics and Operational Implications

We present below a set of new metrics for quantifying traffic efficiency in a 2D transit space. Let $A$ denote the total size of the transit area under study. Adopting the Euclidean distance, each vehicle $i$ exclusively occupies a circular area of size $\pi (\sqrt{h^2 + 0.25s_i})^2$, where $h$ is the required time of separation between any two vehicles. This area is called the vehicle’s buffer zone. At any given time, no two buffer zones will overlap because of the collision-avoidance constraint. The instantaneous traffic density $p_i$ at time $t$ is defined as the ratio between the sum of all vehicles’ buffer zone areas and the total area $A$, $p_i = \sum_{j=1}^{N_i} \pi (\sqrt{h^2 + 0.25s_j})^2 / A$, where $N_i$ is the number of active vehicles at time $t$. For each vehicle $i$ at time $t$, the instantaneous speed efficiency $\eta_{spd}^i$ is defined as the ratio between its current speed and its maximum speed, the heading efficiency $\eta_{hag}^i$ as the cosine similarity between its current heading and the destination-pointing direction, and the flow efficiency $\eta_{flow}^i$ as the product of the speed efficiency and the heading efficiency. The rationale behind these definitions is that congestion effect will exhibit itself in two forms, to lower a vehicle’s speed, hence causing delay, or to divert a vehicle’s heading, hence causing detour. In congestion-free traffic, a vehicle will move to its destination along the straight-line path at the maximum speed, in which case all efficiency factors will be 1. This can be used as a baseline for comparisons. Furthermore, the granular instantaneous metrics can be aggregated at the trip level. For instance, the trip delay factor of a vehicle is defined as the actual trip time...
divided by the *ideal trip time*, which is the time it would take to travel from origin to destination along a straight line at maximum speed. Similarly, the *trip detour factor* is defined as the actual distance traveled divided by the straight-line distance between the origin and destination. Road transportation, for example, typically results in a much higher detour factor than air transportation because vehicles are confined to travel on road segments, which do not form the point-to-point shortest path for most trips. Detour factor and flow efficiency both imply the energy efficiency of the system, that is, how much useful transit per unit of energy consumed by the vehicles. On the system level, *throughput*, denoted by $V_t$, is the total effective transit made by all active vehicles per unit of time. It measures the *temporal efficiency* of the system. The more vehicles the system accommodates, the higher the throughput will be. However, there is clearly a limit to this positive correlation. If density is too high to be conducive to effective transit, congestion will curb the system throughput. The metrics discussed above are summarized in Table 1.

System regulators need to beware of the trade-off between temporal efficiency and energy efficiency. The optimal operating point of the system should depend on the time value of delivery and the energy cost of the transit method. It is also important to note the functional dependence of density on $h$, the time headway. Vehicle count and space size held equal, systems with a higher accuracy in sensing, control, and mobility can tolerate a smaller $h$ and hence achieve a higher throughput. The relationships between density and different performance metrics are explored via simulation in Section 5.4.

4. UAV Motion-Planning Model and Algorithms

Motion planning involves making decisions about vehicles’ kinematic states over time to achieve certain goals. Our grand goal is to make safe and efficient use of the 2D transit space, which loosely translates to an operational objective of maintaining a feasible yet effective traffic flow. Here, “feasible” means that safe separation is maintained at all times, and “effective” means that the traffic keeps moving in a way to eventually get every vehicle to its destination quickly and with minimal detour.

4.1. A Nonlinear Formulation and Its Challenges

Let us first formulate the task of directing a set $\mathcal{A}$ of vehicles to their respective destinations within a set $\mathcal{T} = \{1, \ldots, T\}$ of time intervals, though ultimately our algorithm is not confined to the scope of any fixed set of vehicles or of a fixed time horizon. Relevant notations in addition to those defined in Table 1 are summarized in Table 2.

Given initial speed $v_{i,0} = 0$ and initial location $u_{i,0}$ of each vehicle $i$ at $t = 0$ [with $||u_{i,t} - u_{i,0}|| \geq S_{ij}$; for all $(i,j) \in \mathcal{E}$], we formulate the MP model as follows:

$$\min_{u_{i,t}} \sum_{i \in \mathcal{A}, t \in \mathcal{T}} ||u_{i,t} - D||_2$$

s.t. $u_{i,t} = u_{i,t-1} + (v_{i,t} + v_{i,t-1})/2$, $\forall i \in \mathcal{A}, t \in \mathcal{T}$,

$$||v_{i,t} - v_{i,t-1}|| \leq s_i, \forall i \in \mathcal{A}, t \in \mathcal{T}$$

$$||u_{i,t} - u_{j,t}|| \leq S_{ij}, \forall (i,j) \in \mathcal{E}, t \in \mathcal{T}$$

The objective (2) is to minimize the distance to destination summed over all vehicles and all time points. Constraints (3) to (5) are discretized EOMs, treating each vehicle as a point mass. Specifically, (3) stipulates a vehicle’s locations between successive time periods, assuming the vehicle makes uniform acceleration during each unit time interval. Note that in the last section, we adopted a slightly simpler EOM (assuming linear uniform-speed motion) for ease of exposition, although the analytical results obtained therefrom are applicable here. Constraints (4) and (5)
limit the acceleration and the top speed of each vehicle. Constraints (6) are the collision-avoidance constraints, imposing a separation distance of at least $S_{ij}$ between vehicles $i$ and $j$ for all pairs $(i, j)$ close enough to be collision cautious. These constraints couple different vehicles together and also make the solution space nonconvex.

The MP model serves as the baseline model. The first question one might ask is why the objective (2) is a good choice with regard to achieving the grand goal. Indeed, in a continuous traffic operation, it is hard to tell what constitutes a global optimal motion plan, particularly when new information (e.g., new vehicle entrance) is gradually revealed and has to be accounted for dynamically over time. We present an indirect argument for the soundness of the objective function and resort to the metrics developed in Section 3 to measure the practical performance of motion plans.

Let us look at a reduced motion planning (RMP) model, defined by (2)–(5). RMP corresponds to congestion-free traffic in which collision avoidance is not a constraint. In this condition, Theorem 3 says that the objective function (2) would induce a straight-line path at full speed from the origin to destination for each vehicle, which is the best possible result with regard to the grand goal.

**Theorem 3.** The solution to RMP gives a minimum-distance and minimum-travel-time path from origin $D_i$ to destination $O_i$ for each vehicle $i$.

In congested traffic, however, low travel time and short travel distance are not necessarily aligned. To avoid collision, a vehicle may take an action anywhere between the following two extremes: (a) take a long detour at the maximum speed to bypass the conflict area and (b) stop and wait until the collision risk is resolved by the other vehicle(s) involved in the situation and then proceed along the shortest path toward the destination. Apparently, by staying on the shortest path, option (b) might require more en-route time than option (a). We argue that in such cases, the objective function (2) serves to mitigate delay and detour on the system level, which will be assessed separately using metrics developed in the last section.

### 4.1.1. Remarks on Simplifying Assumptions

To serve methodological development, the MP model embodies a plethora of assumptions that greatly simplify the real world. Several practical elements absent from the model are worth mentioning: wind effects, interactions with the payload, and energy consumption are significant factors in UAS control and routing practices; flight plan disruption and maneuvering fuel costs are common considerations in conflict resolution; and anticipating upcoming trip demand could alleviate the myopia in a deterministic planning model.

### 4.1.2. Remarks on the Implicit Right-of-Way Priority

Because model MP minimizes the total distance to destination, in the case of a two-vehicle conflict, for example, it will favor the vehicle farther away from its destination, because a delay in that vehicle would contribute a greater value (i.e., its distance to destination) to the objective function than would the other. This is a byproduct of using the objective function (2) to induce shortest paths. A simple case illustrating this phenomenon is provided Online Appendix B. At first sight, this resolution may seem unfairly discriminatory against vehicles closer to their destinations. Nevertheless, absent further knowledge about the specific system, the best assumptions one can have are that all zones are equally likely to become congested and that all vehicles’ origins and destinations are uniformly distributed across the space, and therefore, on average, the collision-avoidance cost (delay or detour) is shared evenly across all vehicles in the field. In fact, such a discriminatory effect is negligible when the look-ahead horizon $T$ is limited, as is the case in the receding-horizon algorithm developed in Section 4.2. Moreover, the priority order can be easily adjusted by adding different weights to the objective terms, among other means, without sacrificing other desirable properties of the model.

### 4.1.3. Challenges in Solving the MP Model

The form of the MP model as presented in (2) to (6) is appropriate for conceptual exposition, but not suitable for computational implementation. First, given a set of trip requests $(O_i, D_i)$, for $i \in \mathcal{A}$, as well as vehicle configurations $a_i$ and $s_i$, for $i \in \mathcal{A}$, there is no way to exactly determine a priori the planning horizon $T$ needed to complete all trips. In other words, the end time $t = \min\{t : u_{ij}^* = D_i, \forall i \in \mathcal{A}\}$ exists in the solution of MP only when it is smaller than $T$, and its value is not known until a solution that contains it is found. The size of the problem grows linearly with the value of $T$, so blindly trying an unnecessarily large $T$ is computationally nonviable. Second, when it comes to air traffic management, path planning is not a

---

**Table 2.** List of Notations for the MP Model

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{A}$</td>
<td>Set of vehicles</td>
</tr>
<tr>
<td>$\mathcal{T}$</td>
<td>Set of time intervals</td>
</tr>
<tr>
<td>$\mathcal{C}$</td>
<td>$\subset \mathcal{A} \times \mathcal{A}$, set of collision-cautious vehicle pairs</td>
</tr>
<tr>
<td>$a_i$</td>
<td>Maximum acceleration between successive time points for vehicle $i$</td>
</tr>
<tr>
<td>$S_{ij}$</td>
<td>Minimum separation distance between vehicles $i$ and $j$</td>
</tr>
<tr>
<td>$u_{ij}$, $v_{ij}$</td>
<td>Abscissa and ordinate components of $u_{ij}$, respectively, in 2D space</td>
</tr>
<tr>
<td>$x_{ij}$, $y_{ij}$</td>
<td>Abscissa and ordinate components of $v_{ij}$, respectively, in 2D space</td>
</tr>
<tr>
<td>$\tilde{x}<em>{ij}$, $\tilde{y}</em>{ij}$</td>
<td>Abscissa and ordinate components of the destination vector $D_i$</td>
</tr>
</tbody>
</table>
one-shot deal, but a dynamic, evolving process without a definite time frame. Vehicles that have reached their destinations should be promptly excluded from the planner’s scope, so that they will not block the pathways of other vehicles or prevent other vehicles from landing in the same area. It is not possible to recognize whether a vehicle has reached its destination without significantly complicating the model formulation, for instance, by using integer variables. In addition, as existing trips are being optimized, new trip requests will continue entering the system and must be promptly accommodated. For these reasons, the motion-planning problem needs to be solved on a rolling basis in practice.

4.2. A Receding-Horizon Progressive Heuristic for Motion Planning

Let us cast the finite-horizon MP model into an infinite-horizon traffic management context. For simplicity of exposition, a planar model \((n = 2)\) is adopted, and it is straightforward to extend the model and algorithm to a three-dimensional space. In a planar area, the location and speed vectors are specified by the \(x\) and \(y\) coordinates. Thus, let us define \(u_{i,t} = (x_{i,t}, y_{i,t})\), \(D_i = (\hat{x}_i, \hat{y}_i)\), and \(v_{i,t} = (x'_{i,t}, y'_{i,t})\). At a given time point \(t_0\), the system state is represented by the following parameters: the set of active (en-route) vehicles \(\mathcal{A}_t = \{i \in \mathcal{A} : x_{i,t} \neq \hat{x}_i \) or \(y_{i,t} \neq \hat{y}_i\}\), the current location \((x_{i,0}, y_{i,0})\) and velocity \((x'_{i,0}, y'_{i,0})\) of each active vehicle \(i \in \mathcal{A}_t\), the length of the planning horizon \(T\) with the corresponding time index set \(\mathcal{T}_t = \{t_0, t_0 + 1, \ldots, t_0 + T\}\), and the set of collision-cautious vehicle pairs \(\mathcal{C}_{1,t}\).

The algorithm is designed to work as follows. At time \(t_0\), vehicle motions are planned for the next \(T\) periods. A feasible and nontrivial solution of such a model will recommend an optimal \(T\)-step movement path for each vehicle that obeys all the physical constraints, including the acceleration limits, speed limits, and the collision-avoidance constraints. Instead of following the entire \(T\)-step paths, the vehicles will move only \(M (M \leq T)\) steps according to the recommended paths. By treating the time point \(t_0 + T\) as the new \(t_0\) for the next iteration, the system state is reevaluated and updated. Specifically, during the \(M\)-step move, if a vehicle has reached its destination, it will be eliminated from the active set \(\mathcal{A}_{t_0}\) in the upcoming iteration. The set of collision-cautious vehicle pairs \(\mathcal{C}_{1,t}\) will also be updated, by Theorem 2. The above process then repeats itself until \(\mathcal{A}_{t_0} = \emptyset\), that is, all vehicles have reached their respective destinations. The whole solution process progresses at a step length of \(M\) time periods.

The success of the algorithm depends critically on how this question is addressed: Continuing from location \((x_{i,0}, y_{i,0})\) and velocity \((x'_{i,0}, y'_{i,0})\), which represent the end state of the \(M\)-step move from the previous iteration, how can one ensure that the next \(T\)-step MP model always has a feasible solution? If at any time the model becomes infeasible, the chain of progression will break and the algorithm fail.

In the very beginning of time, when all vehicles are assumed to start from still with acceptable separation distance, the model MP always has a trivial feasible solution which is to dictate all vehicles to remain still, that is, \(v_{i,t} = v_{i,0} = 0\) and \(u_{i,t} = u_{i,0}\) for all \(i \in \mathcal{A}\) and \(t \in \mathcal{T}\). However, such a convenient feasible solution is no longer available when vehicles are in motion, as it is not possible to instantaneously brake moving vehicles to a complete stop (as an attempt to preserve feasibility of collision-avoidance constraints for upcoming time periods).

To reinstate the availability of a feasible solution in each progressive solution of the MP model, an auxiliary constraint is introduced to fix the speed at the end of the look-ahead horizon to zero. It is critical to note here that this treatment will not really bring the solution which is to dictate all vehicles to remain still, as it significantly complicating the model MP, parameterized by \(t_0\). Constraint (13) forces the speed at the end time \(t_0 + T\) to be zero. \(\mathcal{A}_t\) denotes the set of active vehicles at time \(t\), and \(\mathcal{C}_t\) denotes the set of collision-cautious vehicle pairs at time \(t\), for \(t \in \mathcal{T}_t\). From here to Section 4.3 it is convenient to assume \(\mathcal{A}_t = \mathcal{A}_{t_0}\) and \(\mathcal{C}_t = \mathcal{C}_{t_0}\) for all \(t \in \mathcal{T}_t\), for which the choice of \(\mathcal{C}_{1,t}\) is prescribed in Theorem 2. In Section 4.4,
where traffic merging is discussed, $\delta_t$ and $\epsilon_t$ will deviate from the above setting for certain $t \in \mathcal{F}_t$.
Let us see how the progressive solution works via an example.

**Example 2 (Look-Far-Move-Small Planning Process).** Suppose we choose $T = 10$ and $M = 2$, and suppose model $M\hat{P}(t_0)$ has a local optimal solution $(u^{*}_{t_0}, u^{*}_{t_0+1}, \ldots, u^{*}_{t_0+10})$ and $(v^{*}_{t_0}, v^{*}_{t_0+1}, \ldots, v^{*}_{t_0+10})$ for $i \in \mathcal{A}$. Then, the time will be advanced by $M = 2$ steps with the vehicles following the first two steps in the solution and moving to $u^{*}_{t_0+2}, i \in \mathcal{A}$. The next MP solution will take $t_0 + 2$ as the starting time, fix the starting vehicle locations to $u^{*}_{t_0+2}$, and solve for optimal paths for the subsequent $T = 10$ time steps, from $t_0 + 2$ to $t_0 + 12$. In other words, the model instance $M\hat{P}(t_0 + 2)$ will be solved. In this solution, $(u^{*}_{t_0+2}, u^{*}_{t_0+3}, \ldots, u^{*}_{t_0+10}, u^{*}_{t_0+10}, u^{*}_{t_0+10}, u^{*}_{t_0+10})$ and $(v^{*}_{t_0+2}, v^{*}_{t_0+3}, \ldots, v^{*}_{t_0+10}, 0, 0)$ will be supplied as the starting point for variables $(u_{t_0})_{t_0+12}^{t_0+2}$ and $(v_{t_0})_{t_0+12}$, respectively. This starting point is actually a feasible solution for the model $M\hat{P}(t_0 + 2)$. To see this, let us assume, without loss of generality, $\mathcal{A}_{t_0+2} = \mathcal{A}_{t_0}$, and also assume, for now, $\epsilon_{t_0+2} = \epsilon_{t_0}$ (this will be generalized later). Then the constraints (8)–(12) for $t \in \{t_0 + 2, \ldots, t_0 + 10\}$ are identical in both $M\hat{P}(t_0)$ and $M\hat{P}(t_0 + 2)$, and therefore, feasibility of the solution points $(u^{*}_{t_0})_{t_0+10}$ and $(v^{*}_{t_0})_{t_0+10}$ carries over. Furthermore, the starting point essentially “freezes” and extends the still state (recall $v^{*}_{t_0+10} = 0$) of $t = t_0 + 10$ to the next $M = 2$ periods, $t_0 + 11$ and $t_0 + 12$, in which the feasibility of all constraints are retained. The above method of setting starting points is generalized below.

Strategy 1. Given the solution of $M\hat{P}(t_0)$, that is, $(u^{*}_{i,j})_{t_0+10}$ and $(v^{*}_{i,j})_{t_0+10}$ for $i \in \mathcal{A}_{t_0}$, set the starting point for variables $(u_{i,j})_{t_0+10+T}$ and $(v_{i,j})_{t_0+10+T}$, $i \in \mathcal{A}_{t_0+M}$ for model $M\hat{P}(t_0 + M)$ as follows:

- For $t \in \{t_0 + M, \ldots, t_0 + T\}$, set $u_{i,j} \leftarrow u^{*}_{i,j}$ and $v_{i,j} \leftarrow v^{*}_{i,j}$.
- For $t \in \{t_0 + T + 1, \ldots, t_0 + T + M\}$, set $u_{i,j} \leftarrow u^{*}_{i,j+T}$ and $v_{i,j} \leftarrow 0$.

This strategy ensures that each time the $M\hat{P}(t_0)$ model is solved, the solver is supplied with a feasible starting point, which is of critical importance for the solvability of nonconvex nonlinear programs.

There are three important technical issues to be addressed: (1) the side effect of constraint (13), (2) the appropriate choice of $T$ and $M$, and (3) the determination of $\epsilon_{t_0}$ in each iteration.

### 4.2.1. Removing the Side Effect of (13) by Choosing $T$ and $M$

The sole purpose of the requirement that all vehicles come to a stop at the end of each look-ahead horizon is to provide a feasible starting point for the next solution, thereby ensuring that each successive NLP solution in the algorithm chain starts with a feasible solution. We now analyze the effect of this constraint and show that by choosing the parameters $T$ and $M$ prudently, restrictive effects on the actual vehicle motion can be removed.

As reflected in constraint (10), it might take a few time intervals to reduce a vehicle’s speed from maximum to zero. Therefore, the effect of constraint (13) is not limited to the time point $t_0 + T$, but may extend backward along the time line to exert artificial speed limits [that override the constraint (11)] for time points, for example, $t_0 + T - 1, t_0 + T - 2$, etc. How far back this effect extends depends on the time to full stop from presumably the maximum speed, which, on the system level, is upper bounded by $\max_{i \in \mathcal{A}}|s_i|/a_i$. Accordingly, the look-ahead step length $T$ and move-ahead step length $M$ must follow the following two rules:

\[
T \geq 2 \max_{i \in \mathcal{A}}|s_i|/a_i, \quad (14)
\]

\[
M \leq T - \max_{i \in \mathcal{A}}|s_i|/a_i. \quad (15)
\]

The lower bound on $T$ in (14) ensures that in each planning cycle the vehicles have enough time to accelerate to the maximum speed as in free traffic even when starting from still. The upper bound on $M$ in (15) stipulates that the move-ahead step should be conservative enough so as not to land in the forced deceleration stage. Following these rules is a minimum requirement that will remove the definite, foreseeable, and significant restriction on the solution space brought about by (13). Going beyond the minimum (i.e., setting a larger $T$ and smaller $M$ than the required bounds, respectively) will further expand the solution space of each NLP solution and might help improve the overall solution quality, but the marginal benefit will be thin compared with the effort made to barely satisfy the minimum. This argument is corroborated by experimental results in Section 5.1.

Intuitively, the more steps one looks and plans into the future, the sooner one can detect possible deadlocks in future time periods, and thus the more headsup time one has to start making corrections for them. On the other hand, a large $T$ also results in a large MP model to solve in each iteration, which takes more computing time. A similar trade-off goes for the choice of $M$. Having conservative move steps leaves more time for adjustments to avoid future deadlocks, at the cost of wasting much of the motion computation by not fully following the computed paths.

### 4.3. Avoiding Deadlocks Due to Local Optima

In each iteration of the progressive algorithm, the nonconvex MP model that drives all vehicles toward their final destination is solved. Each MP solution always starts with a feasible solution and ends with a locally optimal solution. Part of the solution (vehicle motion planned for the first $M$ time intervals) is used for setting the actual movement of vehicles, and the remaining part of the solution serves as the building block.
of a feasible starting point for the next iteration. Time advances by $M$ intervals in each iteration. All seems to work well. However, there is one question remaining to be answered: how and when does this process terminate?

After each iteration, the active set of vehicles $\mathcal{A}_t$ is updated with the current vehicle location. If a vehicle reaches its destination, it will be excluded from $\mathcal{A}_t$ from the next iteration onward. Therefore, a natural termination point is when all vehicles have reached their destinations and $\mathcal{A}_t = \emptyset$. However, there exists another possibility: the solution is stalemated and stops changing over time, in which case the feasible starting point becomes the only feasible solution (hence, locally optimal) for model MP($t$) for all $t$ beyond a certain time point. Such a solution is marked by an unchanging location vector and a zero speed vector. In other words, a cluster of active vehicles stop moving and cannot advance even a tiny bit toward their respective destinations. This phenomenon is called a deadlock. Figure 4 demonstrates a deadlock involving two vehicles. The two vehicles’ destinations are closer to each other than the required separation radius, so when they reach the vicinity of their destinations and prepare for landing at the same time, they get stuck.

Deadlocks are attributed to the solution getting trapped in a local optimum. In a deadlock, any feasible moving direction, if followed, will cause an instantaneous increase in the total distance to the destination. It is possible but very costly to rectify this situation by attempting to solve the MP model to global optimality. Furthermore, as mentioned in the beginning of Section 4.2, it is also quite complex to determine a large enough planning horizon $T$ so that a global optimum of MP will end up being deadlock-free.

In practice, deadlocks are avoided by assigning different right-of-way priorities or issuing direct resolution orders to competing vehicles. Such priority orders may be simple enough to be articulated in traffic rules, such as the right-of-way rule at a four-way stop intersection, or they may be determined by the traffic controller on a case-by-case basis, such as the landing sequence assignment for multiple aircraft attempting to land on the same runway. Long-term planning, such as the deliberate planning of delayed takeoffs to minimize delay in the landing stage (Vranas, Bertsimas, and Odoni 1994), is also typically employed by airlines to minimize fuel costs.

To automatically circumvent deadlocks in the algorithmic chain, we use the heuristic tactic of assigning disparate priorities in the objective function to vehicles deemed to be forming a deadlock. Fortunately, in the look-far-move-small algorithmic paradigm, one does not need to wait until all involved vehicles have fully moved into a deadlock to be able to detect and react to it. As soon as a stalemate arises in the far end of the planning horizon, actions will be taken. A stalemate in a look-far solution is recognized by the occurrence that any two vehicles stop moving and their collision-avoidance constraint becomes active. Specifically, let $\mu^*_i,j$ denote the optimal Lagrangian multiplier associated with the constraint (12) in the local optimal solution of model MP($t_0$). A positive Lagrangian multiplier indicates that the corresponding constraint is active and binding at the solution (Nocedal and Wright 2000). Given an integer parameter $L \geq 1$, which denotes the number of time points to check, we determine that an impending deadlock involving at least vehicles $i$ and $j$ has arisen if the following conditions hold:

\begin{align}
\mu^*_i,j > 0, & \text{ for } t \in \{t_0 + T - L, \ldots, t_0 + T\}, \\
\|u_j^t - u_{j-1}^t\| & \leq \epsilon, \text{ for } t \in \{t_0 + T - L, \ldots, t_0 + T\}, \\
\|u_{j,t} - u_{j-1,t}\| & \leq \epsilon, \text{ for } t \in \{t_0 + T - L, \ldots, t_0 + T\},
\end{align}

where $\epsilon$ is a small positive number. In each iteration, the algorithm checks in the MP($t_0$) solution the above conditions for all vehicle pairs in $\mathcal{E}_{t_0}$. If the conditions are met for a vehicle pair $(i,j)$, $\alpha_i$ is elevated by a factor of $\gamma$, and $\alpha_j$ is kept unchanged. In numerical instances, $\gamma$ should be set large enough, commensurate with the numeric scale of $\max_{x_{0\text{end}}}(\|O_i - D\|)$, to unambiguously arbitrate the priorities. In our numerical experiments, $\gamma$ is set to 100. Such a disparity in priority will elicit the lower-priority vehicle $j$ to yield and make way for the higher-priority vehicle $i$, therefore circumventing a deadlock situation. An instance of deadlock-breaking is illustrated in Figure 5.

Note that satisfaction of the above conditions does not indicate strictly a pairwise deadlock involving only two vehicles. A cluster of multiple vehicles may be involved in a deadlock. It is clearly a feasible strategy.

**Figure 4.** (Color online) Traces of Two Vehicles with Equal Priority

![Figure 4](image_url)

Notes. The planning horizon is $T = 30$, but the plot shows the traces for $t = 1, \ldots, 18$. Both vehicles are unable to reach their respective destinations (marked by the square points), as their movements are stalemated by a local optimum at $t = 18$. The gray dashed lines indicate the distances to the destinations at different time points. The circles indicate the separation constraints at $t = 1$ and $t = 18$, respectively.
to deal with them one by one, making one of the vehicles pass quickly as a first step to alleviate the congestion. Correspondingly, one needs to ensure that only one vehicle in a deadlock cluster has a higher priority and all other vehicles in the cluster remain at the base priority level, to avoid ambiguity and cycling. This can be achieved by a simple bookkeeping rule, that is, maintaining a set $\mathcal{DL} \subset \mathcal{L}_0$ for vehicles that are in the deadlock resolution mode, and a set $\mathcal{DL} \subset \mathcal{L}_0$ for vehicle pairs that are in the deadlock resolution mode. Elements in these sets are taboo, that is, excluded from consideration for further priority adjustments. The priority-setting mechanism is summarized in Algorithm 1. The algorithm checks for impending deadlocks and, if any are detected, elevates the priority of an arbitrary vehicle (determined by the lexicographic ordering of the vehicles’ index names) involved in the deadlock, and then augments the sets $\mathcal{DL}$ and $\mathcal{DL}$ for global bookkeeping.

**Algorithm 1 (Deadlock Detection and Priority Adjustment)**

1. procedure $\text{SetPriority}(\mathcal{L}_0, \mathcal{DL})$, Solution of $\text{MP}(t_0)$
2. for $(i, j) \in \mathcal{L}_0$
3. if $i \notin \mathcal{DL}$ and $j \notin \mathcal{DL}$ then
4. Check $(i, j)$ for conditions (16)–(18)
5. if Conditions hold then
6. $\alpha_i \leftarrow \gamma \alpha_i$
7. $\mathcal{DL} \leftarrow \mathcal{DL} \cup \{i, j\}$
8. end if
9. end if
10. end for
11. end procedure

The priority factor in the objective function is intended only for breaking a deadlock that would otherwise materialize in the default configuration where all vehicles have the same priority. As soon as an impending deadlock is successfully circumvented, the elevated priority should be reset to the base level. This ensures that the motion-planning model continues to function on an equitable basis, and also releases affected vehicles from the taboo state in case their priority needs to be adjusted for breaking the next impending deadlock. Two vehicles $i$ and $j$ that have previously entered a deadlock cluster are said to be completely disengaged if the following conditions hold at the current MP($t_0$) solution:

$$\mu^{*}_{i, j, t} = 0, \text{ for } t \in \{t_0, \ldots, t_0 + T\}. \quad (19)$$

The procedure of scanning for disengagement and resetting priorities is given in Algorithm 2. Along with resetting priorities to the base level, the procedure also takes care of the bookkeeping sets $\mathcal{DL}$ and $\mathcal{DL}$, that is, releasing appropriate vehicles from the taboo state.

**Algorithm 2 (Postresolution Priority Reset)**

1. procedure $\text{ResetPriority}(\mathcal{DL}, \text{Solution of MP}(t_0))$
2. for $(i, j) \in \mathcal{DL}$ do
3. Check $(i, j)$ for conditions (19)
4. if Conditions hold then
5. $\alpha_i \leftarrow \alpha_j$
6. $\mathcal{DL} \leftarrow \mathcal{DL} / \{i, j\}$, $\mathcal{DL} \leftarrow \mathcal{DL} / \{i, j\}$
7. end if
8. end for
9. end procedure

We have eliminated all undesired termination points of the algorithm, including the occurrence of an infeasible MP($t_0$) caused by the rolling augmentation of $\mathcal{L}_0$, and the motion stagnation caused by local optima of MP($t_0$). Therefore, only one termination point is possible, which is attained when all vehicles have reached their respective destinations. The overall progressive motion planning (PMP) algorithm is presented in Algorithm 3. Inputs to the algorithm include the set of vehicles $\mathcal{L}$; each vehicle’s maximum speed $s_i$, maximum acceleration $a_i$, origin coordinate $O_i$, and destination coordinate $D_i$; the look-ahead step $T$; the move-ahead step $M$; and the minimum separation time $h$. The outputs consist of each vehicle’s location and speed at each time point leading to its destination. This algorithm guarantees a feasible output as long as the trips’ origins are feasible for the separation constraints.

**Algorithm 3 (Progressive Motion Planning)**

1. procedure MP($\mathcal{L}$, $T$, $M$, $h$, $O_i$, $D_i$, $s_i$, $a_i$)
2. $t_0 \leftarrow 1$, $u_{i, t_0} \leftarrow O_i$, $v_{i, t_0} \leftarrow 0$, $\alpha_i \leftarrow 1$, $t_i \leftarrow 1$ for $i \in \mathcal{L}$
3. $\mathcal{L}_0 = \mathcal{L}$, $\mathcal{DL} \leftarrow \emptyset$, $\mathcal{DL} \leftarrow \emptyset$
4. while $\mathcal{L}_0 \neq \emptyset$ do
accommodating new vehicle entrance

In ET(k), the constant $S_{kl}$ is equal to $(s_k + s_l)\cdot\sqrt{h^2 + 0.25}$. The set $V_k$ contains vehicles whose projected locations at time $t_0 + T$ are close enough to $O_k$ so that the separation constraints need to be inspected. It is constructed based on system states and an outlook as of $t_0$ and thus is an exogenous parameter to the model. The constant $R_k$ is the radius of the search area. The rationale behind searching a neighborhood of $O_k$ for an entrance point rather than fixing the entrance point strictly to $O_k$ is rooted in practical considerations. It is practical to assume that in preparation for entrance to the main traffic flow, a vehicle is able to maneuver freely to arrive at a designated (based on system feasibility) entrance point not far too from its launch point $O_k$. The maneuvering can occur, for example, in a lower-altitude space reserved for takeoff where the takeoff traffic is managed in serial mode. By prudently setting $\psi_k$ and $R_k$, ET(k) is practically a small-scale problem and can be solved efficiently by a global solver such as GloMIQO (Misener and Floudas 2013) or BARON (Tawarmalani and Sahinidis 2005).

In dense traffic, ET(k) may be infeasible at $t_0$, meaning that the requested entrance cannot be accommodated at the moment, and the vehicle must wait on the ground or hover in the lower-altitude layer until enough space in the main traffic becomes available. In this case, solution of ET(k) will be attempted again and again in subsequent iterations of the main algorithm until a solution is found.

When a solution, say, $(\hat{x}_k^i, \hat{y}_k^i)$, to ET(k) is found at time $t_0$, the following will be done to incorporate vehicle $k$ into the main planning loop: (1) Add $k$ to $\mathcal{A}_t$ for $t \geq t_0 + T$ and add $(k, i), i \in V_k$, to $\mathcal{E}_t$ for $t \geq t_0 + T$. (2) Fix $(x_{ki}, y_{ki}, y_{ki, t + T})$ to $(\hat{x}_k^i, \hat{y}_k^i)$, and fix $(x_{ki, t + T}, y_{ki, t + T})$ to $(0, 0)$.

With augmented sets $\mathcal{A}_t$ and $\mathcal{E}_t$ at $t = t_0 + T$, MP($t_0$) remains feasible because all newly added constraints [new rows of (11) to (13)] at $t_0 + T$ are satisfied. Therefore, the chain of feasibility is preserved. The above actions inform the main loop of the PMP algorithm that vehicle $k$ is scheduled to enter the main traffic at time $t_0 + T$. Subsequent planning iterations will incorporate the new vehicle, whose actual entrance will take place at time $t_0 + T$.

Figure 6 demonstrates the new vehicle entrance process. In this scenario, vehicles 1 and 2 represent the existing traffic, and vehicle 3 is a new entrant. Similar to the situation when an automobile merges into highway traffic, the space accommodation, as well as the postentrance motion planning, starts way ahead of the actual entrance, which occurs at time $t = 7$. At $t = 1$, a feasibility check is presumably performed and passed, and vehicle 3 appears in the planning
horizon at its desired entrance point. At $t = 3$ and 5, vehicle 3’s postentrance path is being planned along with the existing traffic even though the vehicle is not physically in the traffic yet. At $t = 7$, vehicle 3 enters the traffic and starts to follow its planned path. Note that feasibility of vehicle 3’s entrance is ensured at $t = 1$ by an optimal solution to ET($k$). If ET($k$) was not feasible at $t = 1$, it would be solved again at $t = 3$ and so forth, until a solution became available.

The imposition of a minimum $T$-period pre-entry delay may seem to be a stringent constraint in the context of the on-demand business model that will predominantly exist in the system under consideration. We acknowledge that in periods or regions of low traffic density, faster entries are possible. For instance, an instantaneous entry of a new vehicle in the white area in Figure 3 will cause no problem under the current algorithmic framework. The ET($k$) approach is designed to work in an arbitrary condition and hence can be construed as a worst-case resort.

5. Experiments

The PMP algorithm was implemented in GAMS (Windows version, release 25.0.2), and the core optimization problem MP($t_0$) was solved using the CONOPT solver (Drud 1985, 2018) with default options. The experimental hardware was an HP Z820 workstation with two Xeon E5-6290 v2 CPUs and 96 GB of RAM. We first demonstrate the solution to stylized
instances to understand the solution process, and then run a set of computational experiments to gain more insights about high-density air traffic.

5.1. Effects of $T$ and $M$

The PMP algorithm is inherently a heuristic approach, progressively weaving the overall motion plan with segments of locally optimal plans. We use a simple stylized case to compare PMP’s solution to the global optimum and to demonstrate the effects of parameters $T$ and $M$ on the solution quality, particularly in the context of Section 4.2.1.

The case consists of four vehicles whose OD coordinates, maximum speeds $s_i$, and maximum accelerations $a_i$, as well as the system-level time separation $h$, are given in Table 3. For this case and subsequent numerical cases, the distance unit ($du$) and time unit ($tu$) are kept indefinite, so the units for distance, time, speed and acceleration should be construed as $du$, $tu$, $du/tu$ and $du/tu^2$, respectively. We first solve MP with $T = 40$ using a global solver. At the solution, all vehicles are able to reach their destinations at time 33, and the total transit distance is 951.2. The motion trace at this solution is shown in the upper left plot in Figure 7. We also run the same case using PMP at $T = 2$ to 9 and $M = 1$ to $T$, a total of 44 combinations. The solutions, that is, time to completion and the total transit distance, are given in Table 4. Note that the objective function (2) that couples time and distance is used for inducing good-property solutions, whereas the objective value itself is of little practical meaning to traffic managers. In the interest of space, the objective values are not reported here.

As indicated by the values of $s_i$ and $a_i$, the time to full stop is 3. Thus, according to the rules (14) and (15), the recommended ranges for $T$ and $M$ are $T \geq 6$ and $M \leq T - 3$. Conforming configurations have their time entries in bold in Table 4. If we consider these

![Figure 7](Color online) Motion Traces of Four-Vehicle Traffic Generated by Four Different Algorithmic Configurations

<table>
<thead>
<tr>
<th>$i$</th>
<th>$O_i$</th>
<th>$D_i$</th>
<th>$s_i$</th>
<th>$a_i$</th>
<th>$h$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-80</td>
<td>-80</td>
<td>80</td>
<td>80</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>80</td>
<td>-80</td>
<td>-80</td>
<td>80</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>80</td>
<td>80</td>
<td>-80</td>
<td>-80</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>-80</td>
<td>80</td>
<td>80</td>
<td>-80</td>
<td>9</td>
</tr>
</tbody>
</table>

Notes. Each vehicle’s safety disc is plotted at $t = 1$. “Time” indicates the number of time intervals needed to complete the routing.
experiments as a “random sample” drawn from the space of all possible data scenarios and algorithmic configurations, we can perform a Welch’s $t$-test for the null hypothesis that the mean time for the conforming configuration is equal to the mean time for the non-conforming configuration, where conforming means following the rules (14) and (15). The $p$-value for the one-sided test is 0.0006, giving strong evidence that the recommended rules are effective at inducing shorter overall delays. Furthermore, the variance in time for the conforming configurations are quite small, indicating that going beyond the minimum recommendation (i.e., further increasing $T$ and decreasing $M$) does not yield significant improvements in transit time. The transit distances, shown in the right part of the table, are not affected as much by artificial speed limits incurred by nonconformance. However, as a general trend, they do benefit from larger look-ahead and smaller move-ahead steps. Three selected configurations of $T$ and $M$ are plotted in Figure 7, along with the global solution of MP. The look-far-move-small strategy as exemplified by the setting $T = 9$ and $M = 1$ approximates the global solution very well, whereas the more frugal and aggressive setting (i.e., the one with $T = 6$ and $M = 3$) generates a slightly curvy motion path. The setting with $T = 2$ and $M = 2$ not only artificially limits the move speed (hence, prolongs the total transit time), but also exhibits a myopic motion plan; that is, the vehicles head straight toward the destination until bumping into a conflict zone, then make sharp detours to avoid collision.

In terms of $T$’s and $M$’s effects on computational cost, the experimental results are aligned with what is postulated in Section 4.2.1, that is, a setting with a large $T$ and small $M$ takes more computing time. In this four-vehicle instance, the setting with $T = 9$ and $M = 1$ took 8.48 seconds to execute (on average, 8.48/33 = 0.26 seconds per major iteration), and the setting with $T = 6$ and $M = 3$ took 2.85 seconds [on average, 2.58/((34 − 1)/3) = 0.23 seconds per major iteration]. Moreover, if we were to assume that a unit time interval in the model corresponds to one second of clock time, then the former setting would receive a per-iteration computing budget of one second and the latter setting a budget of three seconds. The actual computing times would be within budget in both settings under this assumption.

### 5.2. Conflict Resolution of 30 Vehicles

We continue to apply the PMP algorithm to a traffic scene consisting of 30 vehicles starting in a ringed formation, shown in the upper left of Figure 8. The destination of each vehicle is the point symmetric to its origin on the circle. For example, vehicle 1’s destination coincides with the origin of vehicle 16 and vice versa. The radius of the circle is 250; hence, each vehicle’s trip distance is 500, and the total transit distance is 15,000. The speed limit is $s_l = 12$, the acceleration limit $a_l = 4$, and the separation time is $h = 2$. The small circle around each vehicle shows the vehicle’s buffer zone.

An ad hoc feasible solution one might come up with is to have all vehicles move along the perimeter of the circle at the same pace, figuratively rotating the ringed formation by 180 degrees. In this solution, all vehicles refrain from pursuing the shortest path in exchange for time efficiency. The corresponding total trip distance is approximately $30 \times 250 \times \pi = 23,562$, which amounts to a detour factor of 1.57. Another ad hoc solution manageable by a human dispatcher would be to dispatch vehicles in small groups at a time. For example, let vehicle 1 and vehicle 16 traverse first while keeping other vehicles waiting. Such a solution would require less detour but incur much more delay, which would correspond to an inefficient use of the airspace and result in a very low flow efficiency.

We used the PMP algorithm at $T = 6$ and $M = 3$ to produce a more efficient solution with a total trip distance of 19,385. Figure 8 demonstrates snapshots of the traffic scene at four time points, whereas a full animated demonstration is available in the online companion.

In the initial stage ($t = 1$ to 23), all vehicles make greedy moves toward their destinations, ostensibly gravitating to the center zone. The instantaneous flow efficiency initially ramps up to a high level

### Table 4. Time to Route a Four-Drone Case with Different $T$ and $M$ Values

<table>
<thead>
<tr>
<th>$T, M$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>85</td>
<td>58</td>
<td>125</td>
<td>41</td>
<td>56</td>
<td>85</td>
<td>33</td>
<td>34</td>
<td>76</td>
</tr>
<tr>
<td></td>
<td>979</td>
<td>977</td>
<td>980</td>
<td>974</td>
<td>971</td>
<td>991</td>
<td>988</td>
<td>971</td>
<td>973</td>
</tr>
<tr>
<td>Distance</td>
<td>33</td>
<td>34</td>
<td>33</td>
<td>34</td>
<td>34</td>
<td>34</td>
<td>34</td>
<td>34</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>34</td>
<td>33</td>
<td>34</td>
<td>34</td>
<td>33</td>
<td>38</td>
<td>46</td>
<td>34</td>
</tr>
<tr>
<td></td>
<td>966</td>
<td>977</td>
<td>983</td>
<td>977</td>
<td>971</td>
<td>981</td>
<td>955</td>
<td>963</td>
<td>957</td>
</tr>
<tr>
<td></td>
<td>955</td>
<td>963</td>
<td>968</td>
<td>967</td>
<td>982</td>
<td>981</td>
<td>984</td>
<td>959</td>
<td>954</td>
</tr>
<tr>
<td></td>
<td>959</td>
<td>957</td>
<td>957</td>
<td>954</td>
<td>955</td>
<td>953</td>
<td>955</td>
<td>965</td>
<td>963</td>
</tr>
</tbody>
</table>

*Note. (T, M) configurations that conform to the rules (14) and (15) have their time entries in bold.*
The ramping effect during times 1 to 3 as shown in Figure 9 is due to acceleration from the still state and then drops dramatically as a severe traffic jam is formed around time 24. The subsequent traffic movement demonstrates an interesting visual effect, in which the whole cluster of vehicles rotates in coordination so that all vehicles get closer to their destinations. During time $t = 40$ to 50, the major rotation is completed, and the compact cluster starts to disintegrate into smaller clusters. In the final stage, that is, $t = 60$ to 80, most vehicles decelerate to exactly reach their destination points. This explains why the speed efficiency ramps down during this period. The overall process finishes at time 80, and the total distance traveled by all vehicles is 19,385.

Figure 9 tracks the speed, heading, and flow efficiency metrics over time. It can be seen that apart from the ramping stages in the beginning and the end periods, the instantaneous speed and heading efficiencies are for the most part positively correlated. This implies that delay and detour concur at proportional intensities, both of which are related to the local density of traffic. In addition, the instantaneous efficiency metrics vary greatly over time, prompting the use of time-averaged metrics to summarize the overall system performance. We will do this in the subsequent simulation analysis.

5.2.1. Computational Efficiency. The CONOPT solver (single thread) was used to solve the model $MP(t_0)$ in
each iteration of the PMP algorithm, and the whole solution process took less than 9 minutes (2.5 minutes when CONOPT4 was used).

For comparison, a solution was also computed by solving the MP model (2)–(6) directly using a large enough $T$. Given the knowledge that PMP could complete in 80 time steps, $T$ was set to 90 in this run to allow for enough time. Because it was a one-shot solve, all pairs $(i, j)$ with $i < j$ were included in $c$. The resulting NLP model had 47,086 rows, 10,201 columns, and 188,341 nonzeroes. It took CONOPT4 (utilizing all 20 cores) 8.7 hours to find a local optimal solution. The solution converged at time 83 when all vehicles had reached their respective destinations. The total distance traveled was 20,759. This run could also be construed as the first iteration of a run by the PMP algorithm configured with $T = 90$.

The motion trace generated by the direct solution of the MP looks similar to that obtained by the PMP algorithm shown in Figure 8. The full motion trace data and animation are provided in the online companion.

The progressive solution appears to be superior to the direct MP solution in all regards: it takes less time (80 versus 83) to dispatch all vehicles, it leads to a shorter total distance (19,385 versus 20,759), and it takes much less computational time (2.5 minutes versus 8.3 hours). Despite being counterintuitive, it is a legitimate result given the subtle difference between what is being optimized and what is being assessed here as performance metrics. Specifically, the objective function couples time and distance together, whereas the quality of a solution is more conveniently assessed by transit time and transit distance separately.

Table 5. Solution of 20 Randomized Cases of 20-Vehicle Conflicts

<table>
<thead>
<tr>
<th>Case</th>
<th>Time</th>
<th>Detour</th>
<th>R. dist.</th>
<th>CPU</th>
<th>Time</th>
<th>Detour</th>
<th>R. dist.</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>44</td>
<td>1.80</td>
<td>0.62</td>
<td>183</td>
<td>39</td>
<td>1.69</td>
<td>0.92</td>
<td>5,682</td>
</tr>
<tr>
<td>2</td>
<td>44</td>
<td>1.83</td>
<td>0.90</td>
<td>187</td>
<td>52</td>
<td>1.93</td>
<td>0.11</td>
<td>6,485</td>
</tr>
<tr>
<td>3</td>
<td>51</td>
<td>1.84</td>
<td>0.92</td>
<td>202</td>
<td>41</td>
<td>1.70</td>
<td>0.54</td>
<td>5,941</td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>1.86</td>
<td>0.92</td>
<td>176</td>
<td>53</td>
<td>2.29</td>
<td>0.91</td>
<td>10,746</td>
</tr>
<tr>
<td>5</td>
<td>42</td>
<td>1.79</td>
<td>0.38</td>
<td>172</td>
<td>60</td>
<td>2.36</td>
<td>31.46</td>
<td>7,167</td>
</tr>
<tr>
<td>6</td>
<td>50</td>
<td>1.97</td>
<td>0.25</td>
<td>163</td>
<td>46</td>
<td>1.85</td>
<td>0.66</td>
<td>5,981</td>
</tr>
<tr>
<td>7</td>
<td>39</td>
<td>1.72</td>
<td>0.51</td>
<td>131</td>
<td>51</td>
<td>1.94</td>
<td>0.49</td>
<td>5,978</td>
</tr>
<tr>
<td>8</td>
<td>44</td>
<td>2.26</td>
<td>0.72</td>
<td>233</td>
<td>60</td>
<td>2.32</td>
<td>339.60</td>
<td>4,234</td>
</tr>
<tr>
<td>9</td>
<td>40</td>
<td>1.71</td>
<td>0.36</td>
<td>143</td>
<td>35</td>
<td>1.59</td>
<td>0.53</td>
<td>3,815</td>
</tr>
<tr>
<td>10</td>
<td>46</td>
<td>1.68</td>
<td>0.26</td>
<td>230</td>
<td>60</td>
<td>1.96</td>
<td>1.61</td>
<td>7,023</td>
</tr>
<tr>
<td>11</td>
<td>54</td>
<td>1.99</td>
<td>0.31</td>
<td>121</td>
<td>60</td>
<td>2.15</td>
<td>56.37</td>
<td>4,752</td>
</tr>
<tr>
<td>12</td>
<td>43</td>
<td>1.83</td>
<td>0.93</td>
<td>169</td>
<td>59</td>
<td>1.86</td>
<td>0.71</td>
<td>4,438</td>
</tr>
<tr>
<td>13</td>
<td>54</td>
<td>2.53</td>
<td>0.75</td>
<td>231</td>
<td>50</td>
<td>2.47</td>
<td>0.54</td>
<td>12,097</td>
</tr>
<tr>
<td>14</td>
<td>48</td>
<td>1.96</td>
<td>0.39</td>
<td>224</td>
<td>52</td>
<td>2.00</td>
<td>0.48</td>
<td>6,460</td>
</tr>
<tr>
<td>15</td>
<td>55</td>
<td>2.34</td>
<td>0.52</td>
<td>197</td>
<td>60</td>
<td>2.10</td>
<td>1.14</td>
<td>8,693</td>
</tr>
<tr>
<td>16</td>
<td>45</td>
<td>1.85</td>
<td>0.27</td>
<td>153</td>
<td>40</td>
<td>1.92</td>
<td>0.78</td>
<td>5,161</td>
</tr>
<tr>
<td>17</td>
<td>43</td>
<td>1.70</td>
<td>0.22</td>
<td>156</td>
<td>53</td>
<td>1.99</td>
<td>0.51</td>
<td>10,869</td>
</tr>
<tr>
<td>18</td>
<td>39</td>
<td>1.67</td>
<td>0.88</td>
<td>192</td>
<td>60</td>
<td>2.09</td>
<td>2.87</td>
<td>9,697</td>
</tr>
<tr>
<td>19</td>
<td>34</td>
<td>1.32</td>
<td>0.60</td>
<td>140</td>
<td>35</td>
<td>1.32</td>
<td>0.81</td>
<td>3,217</td>
</tr>
<tr>
<td>20</td>
<td>53</td>
<td>1.95</td>
<td>0.97</td>
<td>160</td>
<td>37</td>
<td>1.84</td>
<td>0.33</td>
<td>7,682</td>
</tr>
</tbody>
</table>

5.3. More Comparisons Between PMP and the One-Shot MP

We also conducted randomized experiments to elaborate the computational comparison between the PMP algorithm and the single-solve MP model. The goal of the comparison is to demonstrate the value of the PMP algorithm in materializing the good properties of the MP model. We will show that, absent the progressive algorithm, the MP model alone is not very useful in terms of computing time and reliability. We created 20 data sets each consisting of 20 vehicles whose OD coordinates are uniformly sampled in a 200 × 200 square area. The vehicle maximum speeds are randomly picked from the set {8, 12, 16}, time headway $h$ is set to 2, and time to full speed is set to 3. To ensure the cases are solvable, we qualified the OD generation process by requiring not only that the 20 origins are adequately distanced (this is needed by both MP and PMP), but also that the 20 destinations are adequately distanced (this is needed only by the one-shot MP method because it is unable to remove completed vehicles, which might cause deadlocks). The data generation (both here and in the sequel) was performed in R with documented random seeds, so the data source is reproducible.

We first ran all cases using PMP with $T = 6$ and $M = 1$ and observed that all of them can be completed in 60 time intervals. We then ran the cases using a single-solve MP by setting $T = 60$. Dispatch was considered completed if the total residual distance to destination summed over all vehicles was less than 1. The total effective transit distance (sum of straight-line
origin-to-destination distances) for these 20 cases ranged between 2,300 and 2,813, so the above distance tolerance was small enough.

The results are summarized in Table 5. For each approach, the time to complete the dispatch, the detour factor (distance traveled divided by the total origin-to-destination distance), the total residual distance (R. dist.) upon algorithm termination, and the computing time in seconds (CPU) are reported. The performances of the two approaches in terms of time and detour are mixed: PMP won 13 cases in time, 13 cases in detour, and 11 cases in both time and detour. Notably in four cases (Cases 5, 8, 11, and 18), the one-shot MP method failed to complete by a significant residual distance margin. Case 8 is visualized in Figure 10 to aid the diagnosis. It turned out that because of MP’s inability to recognize and remove completed vehicles from the model, the density remained quite high all the time, significantly slowing down the traffic progression.

5.4. Simulation Study of Traffic Efficiency

5.4.1. Data Set Generation. Each data set consisted of N vehicles, each having a randomly generated OD pair in the 500 × 500 square area and a speed limit s_i randomly picked from the set {8, 12, 16} to create abundant variation in density. Time headway was h = 2, and acceleration limit a_i = s_i/3. The following acceptance criteria were applied to ensure initial feasibility and that the trips were nontrivial: (a) the location of all origins must satisfy the separation constraint (12), and (b) the origin and destination of each vehicle must be at least 100 distance units apart.

For the number of vehicles N, the range between 5 and 70 with a spacing of 5 was considered, that is, N = 5, 10, 15, . . . , 70, and at each level, 20 random instances were generated. A similar experimental setup can be found in Šišlák, Samek, and Pěchouček (2008) and Ny

Figure 10. (Color online) Demonstration of Case 8

Notes. The left panel shows the residual distance to the destination over time in the two approaches. PMP completed at time 44 and the one-shot MP failed to complete by time 60. The right panel shows a traffic scene at time 60 in the one-shot MP approach. Several vehicles have not reached their destinations and are moving slowly.

Figure 11. (Color online) An Air Traffic Scene Consisting of 70 Vehicles

Notes. In this particular instance, the density is 0.51, and the detour factor is 2.0, meaning that on average, every mile of effective delivery required two miles of actual travel distance because of congestion. In this snapshot, many vehicles are not heading in the destination-pointing direction (marked by gray dashed lines) because of collision-avoidance constraints.
and Pappas (2010). Altogether, there were 280 instances in the experiment.

Figure 11 demonstrates one of the densest traffic scenarios simulated in the experiment. It is a snapshot of the continuous traffic flow at time $t = 5$. The traffic density as defined in Table 1 is 0.51, meaning that about half of the transit space (outlined by the blue dotted and dashed line in the figure) is filled by vehicles’ buffer zones, that is, the nonoverlapping circles. The size of a circle reflects the maximum speed of the corresponding vehicle, as indicated in Equation (1).

5.4.2. Result Analysis. The first thing to note is the algorithm’s reliability: it was able to produce a feasible motion plan for all instances regardless of the traffic density. This ability is attributed to the meticulous design of all components of the algorithm. Finding the global optimal solution is computationally prohibitive for reasons discussed in Section 4.1.3. Feasibility guarantee accompanied by good solution quality is what one would practically aim for in a motion-planning algorithm. Figure 12 demonstrates the solution quality by plotting pairwise scatter plots over selected metrics. A full summary table as well as the detailed motion trace log files are available in the online companion. In Figure 12, each dot represents one of the 280 experimental instances. Density is the instantaneous traffic density $p_{0}$ in the starting configuration, and throughput is the total effective transit made by all vehicles per unit of time. We can see that, in general, a higher density corresponds to a higher system throughput. However, when density is too high, most vehicles will go a long way and spend a long time maneuvering rather than making effective, destination-bound transits, and therefore the system throughput will deteriorate, as exhibited by the downward trend in the high-density end in the density–throughput plot. Flow efficiency, as expected, is negatively correlated with density, and the correlation appears to be quite uniform across the wide range of densities examined here.

Finding local optimal solutions for large-scale nonlinear programs, even starting with a good, feasible solution, is quite time consuming, as shown in Figure 13. The box plot shows the average solution time for each run of MP($t_{0}$) for cases with different numbers of vehicles. Note that it is not the time it took to complete the entire traffic simulation, which was actually one to two orders of magnitude longer. The computing time seems to increase at an increasing rate with the number of vehicles, and becomes more variable for larger-scale instances. Nonetheless, there are a few things to note. First, in these experiments, the size of the airspace under study was fixed; thus, by increasing the number of vehicles, we were essentially increasing the traffic density, with the aim of (a) demonstrating the algorithm’s reliability in handling dense cases and (b) presenting the system behavior across a wide range of density settings to generate insights on

![Figure 12. Empirical Relation Between Density and System Efficiency](image)

Notes. Each point represents an experimental instance. Density and flow efficiency are negatively correlated. Density and throughput are positively correlated in the moderately low range. At low density, more detour indicates a higher throughput, whereas too much detour will start to reduce throughput.
system response. It is not meant to imply that extremely high densities are common in practice. As observed in Section 2.2, the computational complexity of a system consisting of many vehicles is upper bounded by the complexity of its largest collision-cautious cluster. Second, there exists room to shorten the computing time. For instance, if it was not for our experimental assumption that it took up to three time intervals to accelerate to full speed (which stipulated a minimum planning horizon $T$ of 6), we could adopt a shorter horizon in each incremental run to save time, which of course might come with some compromise in flow efficiency. Tuning the algorithmic parameters to achieve the best combination of solution quality and solution time is best performed for a specific target system, which we will leave for future work. Moreover, new NLP solvers such as CONOPT4 are multithreaded, which can dramatically reduce the computing time on a multicore workstation.

6. Conclusion

Going high density is an inevitable route to increasing the transportation volume without dramatically expanding space use. When many vehicles simultaneously traverse a shared space, intervehicle motion coordination is of critical importance for safety and efficiency. Under safety constraints, the foremost objective of a motion-planning algorithm is to ensure a feasible, deadlock-free solution that keeps the traffic moving at any density level. A good algorithm should also mitigate delay and detours to achieve high transit efficiency. In this paper, a motion-planning algorithm that meets these expectations has been developed, and its effectiveness has been validated via comprehensive simulation experiments. In particular, we have performed the following main contributions:

1. resolved a safety loophole typically found in optimization-based motion-planning models via providing analytical bounds on the minimum separation distance between vehicles;
2. proposed a nonlinear optimization model to centrally coordinate the trajectories and resolve conflicts and developed a progressive algorithm based on a series of heuristic measures to dispatch vehicles in high-density traffic with a feasibility guarantee;
3. coined a set of metrics to measure high-density 2D traffic in a general sense, applied these metrics in simulation experiments to evaluate and validate the motion-planning algorithm, and generated useful insights regarding air traffic density and flow.

6.1. Future Work

UAS traffic management in low-altitude airspace is a relatively new area of research, and there are numerous important and challenging questions to address. Summarized below are some future directions that will extend what has been proposed in this paper.

6.1.1. Dynamic Space Layering and Altitude Assignment. Although the models developed in this paper are applicable for an arbitrary dimension in the Euclidean space, the main target application has been traffic on a 2D planar area. Based on the interface for accommodating new vehicle entrance in the PMP algorithm, future work could include the dynamic assignment of vehicles into different altitude layers to further reduce congestion; see Lehouillier et al. (2017) for reference. This would entail, for instance, a broader-scale traffic optimization that takes the 2D motion-planning algorithm as a critical component. The traffic flow metrics proposed herein will continue to be useful and can be further refined.

6.1.2. Hybrid and Distributed Schemes for Vehicle Motion Computing. The large volume of UAS traffic in the future airspace cannot be all managed centrally by one computer. As discussed in this paper, both temporal and spatial decomposition are available in practical deployments. Future work could investigate data processing and edge-computing paradigms for motion computation, as well as hybrid navigation schemes utilizing vehicle’s on-board AI to achieve a scalable implementation of UAS traffic management.

6.1.3. Robustness, Resiliency, and Uncertainty Management. Uncertainty is almost ubiquitous in any complex system. Apart from the inherent variance in the sensing, control, and actuating components, many stochastic factors such as changing weather and electromagnetic environments also heavily affect an aerial vehicle’s mobile stability and accuracy. Therefore, comprehensive geographic information system, weather, and geomagnetic data need to be incorporated and
operational uncertainties need to be accounted for in practical motion-planning software.

Acknowledgments
The author thanks three anonymous referees for comments and suggestions that improved the quality of this paper.

References


