Scalable Multi-Agent Computational Guidance with Separation Assurance for Autonomous Urban Air Mobility Operations

Xuxi Yang* and Peng Wei†
Iowa State University, Ames, Iowa, 50011

Electrical vertical take-off and landing (eVTOL) vehicles are becoming promising for on-demand air transportation in Urban Air Mobility (UAM). However, successfully bringing such vehicles and airspace operations to fruition will require introducing orders-of-magnitude more aircraft to a given airspace volume. Although there are existing solutions for communication technology, onboard computing capability, and sensor technology, the computation guidance algorithm to enable safe, efficient, and scalable flight operations for dense self-organizing air traffic still remains an open question. In this paper, a message-based decentralized computational guidance algorithm is proposed and analyzed for multiple cooperative aircraft by formulating this problem using Multiagent Markov Decision Process and solving it by Monte Carlo Tree Search algorithm. A novel coordination strategy is introduced by using the logit level-k model in behavioral game theory. To achieve higher scalability, we introduce the airspace sector concept into the UAM environment by dividing the airspace into sectors, so that each aircraft only needs to coordinate with aircraft in the same sector. At each decision step, all of the aircraft will run the proposed computational guidance algorithm onboard, which can guide all the aircraft to their respective destinations while avoiding potential conflicts among them. For validation and demonstration, a free-flight airspace simulator that incorporates environment uncertainty is built in an OpenAI Gym environment. Numerical experiment results over several case studies including the roundabout test problem show that the proposed computational guidance algorithm has promising performance even with the high-density air traffic case.

I. Introduction

A. Motivation

The increase in road traffic congestion in urban areas is a major concern for commuters and a burden on both the environment and the economy. This leads to the interest in Urban Air Mobility (UAM), where the electric vertical

*Graduate Research Assistant, Department of Aerospace Engineering, xuxiyang@iastate.edu. Student Member AIAA.
†Assistant Professor, Department of Aerospace Engineering, pwei@iastate.edu. Senior Member AIAA.

take-off and landing (eVTOL) aircraft is able to alleviate transportation congestion by utilizing 3D airspace efficiently for passenger transport in personal commute or on-demand air taxi. A growing community of interest is forming for the concept of UAM including NASA, Uber, Airbus, Honeywell, and many other entities around the globe [1–5]. Over a dozen companies including Airbus, Bell, Embraer, Volocopter and Aurora Flight Sciences are building and testing their eVTOL aircraft to make it a reality.

However, successfully bringing the scalable UAM operations to fruition will require introducing orders-of-magnitude more aircraft to a given airspace volume, and it is estimated there will be 23,000 aircraft flying major routes within the UAM network by 2035 [6]. Thus the technical challenge is to provide concepts, technologies, and procedures that enable safe and efficient flight operations for the large number of eVTOL aircraft in the given airspace [7]. The Federal Aviation Administration's (FAA) Next Generation Air Transportation System (NextGen) program aims to modernize Air Traffic Control (ATC) and aircraft systems in order to increase the capacity of the airspace [8]. However, the projected capacity increases over the 20-year period are expected to be no more than 50% [9, 10], which is sufficient for the increased demand of commercial aircraft but far below the requirement for the UAM air transportation system. In Unmanned Aircraft Systems (UAS) Traffic Management (UTM), research efforts to increase the airspace capacity typically rely on aircraft being sufficiently equipped and automated that they can operate relatively independently from the existing ATC system and are therefore not subject to its capacity limits [11, 12].

In UAM, researchers have proposed structured airspace where the eVTOL aircraft will fly according to fixed routes [13]. In this paper, we consider a free flight airspace framework [14] since it was shown in previous work [15–18] that free flight with airborne separation is able to handle a higher air traffic density even in the presence of various uncertainties and delays. In addition, free flight can also increase fuel and time efficiency [19]. In a free flight framework, it is implied that aircraft will be responsible for self-separation assurance and conflict resolution [20–22]. Removing the airway structure may make the process of detecting and resolving conflicts between aircraft more complex. However, previous studies [23] show that free flight is potentially feasible because of enabling technologies such as Global Positioning Systems (GPS), data link communications such as Automatic Dependence Surveillance-Broadcast (ADSB) [24], the Next-Generation Airborne Collision Avoidance System (ACAS) [25], and powerful onboard computation.

B. Related Work

Decades of research have explored a variety of approaches for designing collision avoidance systems for both manned and unmanned aircraft, large commercial aircraft and small unmanned aircraft. For conflict detection and resolution of commercial aircraft, there exist several surveys: [26] gives a comprehensive survey of air traffic conflict detection and resolution systems. [27] provides a high-level outline in safety risk analysis, and a recent survey [28] presents a unified mathematical framework for air traffic conflict and collision definitions and methods to estimate the probability of conflict and collision.

Currently, the Traffic Alert and Collision Avoidance System (TCAS) is the only widely-deployed aircraft collision avoidance system, which is required on all large transport aircraft in the world. If the system predicts that the intruder will penetrate a predefined safety buffer, the system will issue a resolution advisory to the pilot to adjust the vertical speed of the aircraft [29]. Recent work on formulating the problem of collision avoidance as a Partially Observable Markov Decision Process (POMDP) has led to the development of the ACAS X family of collision avoidance systems [25, 30]. The version for manned aircraft, ACAS Xa, is expected to become the next international standard for large commercial transport and cargo aircraft. Both TCAS and ACAS are designed to resolve one on one conflicts between aircraft with vertical maneuvers. The difference between them is TCAS uses fixed rules to resolve the conflicts while ACAS uses a probabilistic model (solving a Markov Decision Process (MDP) problem with discrete state space), which leads to a better performance than TCAS. For multiple aircraft case, the Autonomous Operations Planner (AOP) developed by NASA [31] is a flexible and powerful prototype of a flight-deck automation system to support self-separation of aircraft while en route. It incorporates a variety of algorithms that provide flight crew support for strategic and tactical conflict resolutions and conflict-free trajectory planning while meeting route constraints and avoiding airspace hazards. In this paper, we investigate how to resolve conflicts for multiple aircraft and guide the aircraft to their destinations through a series of actions by formulating this problem as a MDP and solving it using the Monte Carlo Tree Search (MCTS) algorithm.

For UAS Traffic Management, NASA's UTM project aims to enable the increasing number of low-altitude, small UAS operations in uncontrolled airspace [32], specifically to enable safe and efficient en route UAS operations for civilian and public applications. The Integrated Configurable Algorithms for Reliable Operations of Unmanned Systems software architecture (ICAROUS), being developed as part of the UTM project, will provide highly assured core software modules for building safety-centric autonomous unmanned aircraft applications [33]. A recent implementation known as DAIDALUS (Detect and Avoid Alerting Logic for Unmanned Systems) [34] is the basis for the ICAROUS software architecture. The core logic of DAIDALUS consists of: (1) definition of self-separation threshold (SST) and well-clear violation volume, (2) algorithms for determining if there exists a potential conflict between aircraft pairs within a given lookahead time, and (3) a determine-processing functionality that provides maneuver guidance and alerting logic.

While research on a UTM system [32] for small UAS (sUAS) operating at low altitudes is relevant for UAM, it provides services appropriate for small UAS that do not always easily extend to the UAM environment [7]. For example, the risk of human injuries in the collision of two sUAS is very low [35], while larger vehicles with humans onboard will present significantly higher risks and the safety standards will have to be significantly enhanced for eVTOL aircraft in UAM over sUAS in UTM. Also, sUAS have the freedom to take off and land nearly anywhere, while eVTOL aircraft in UAM will be restricted to a network of vertiports and therefore require scheduling and spacing services in vertiport terminal airspace.

In this paper, we mainly focus on guidance and conflict resolution systems for these new entrants (such as eVTOL urban air mobility aircraft) in the metropolitan airspace. In the applications for these new entrants, the existing work can be categorized based on the following criteria. We will discuss the related work based on the following categories:

- Centralized/Decentralized [36]: whether the problem is solved by a central supervising controller (centralized) or by each aircraft individually (decentralized).
- Planning/Reacting [37]: the planning approach generates feasible or even optimal paths ahead of execution; whereas the reacting approach typically uses an onboard collision avoidance system to respond to obstacles and other vehicles.
- Cooperative/Non-cooperative: whether there exists communication among aircraft, or between aircraft and the central controller.

In centralized methods, the conflicts between aircraft are resolved by a central supervising controller where most of them require planning ahead of time. Under such scenarios, the state of each aircraft, the obstacle information, and the trajectory constraint (such as required times of arrival and restricted airspace area) are known to the central controller (thus centralized methods are always cooperative), and the central controller in return designs the whole trajectory for each aircraft pre-departure or en route, typically by formulating it as an optimal control problem. These methods can be based on semidefinite programming [38], nonlinear programming [39, 40], mixed integer linear programming [41–44], mixed integer quadratic programming [45], sequential convex programming [46, 47], evolutionary techniques [48, 49], and particle swarm optimization [50]. Besides formulating this problem using optimal control framework, computational geometry methods such as visibility graph [51] and Voronoi diagrams [52] can also handle the path planning problem for aircraft without modeling detailed vehicle dynamics. However, calculating the exact solution for such computational geometry based robot motion planning will become intractable [53] when the state space becomes large or high-dimensional. To address this issue, sample-based planning algorithms are proposed, such as probabilistic roadmaps [54], RRT [55], and RRT* [56, 57]. These centralized methods often generate the whole trajectories for agents. However, as the number of aircraft grows (in multi-agent case), the computation time of these methods typically scales exponentially. Moreover, these centralized planning approaches typically need to be re-run, as new information in the environment is updated (e.g., a new aircraft enters the airspace, or one aircraft failed to execute its planned trajectory).

On the other hand, decentralized methods scale better with respect to the number of agents and are more robust since they are not vulnerable to a single point of failure. However, since the agents act only on local information, global optimality of a decentralized control policy is often hard to achieve [58]. In decentralized methods, all the conflicts are resolved by each aircraft individually. Decentralized methods can be cooperative and non-cooperative. Researchers have proposed several algorithms under the case where the communication between aircraft can be successfully established (cooperative with communication) [59]. Algorithms in [60, 61] are based on message-passing schemes, which resolve

local (e.g., pairwise) conflicts without needing to form a joint optimization problem between all members of the team. In [36], every agent is allotted a time slot to compute a dynamically feasible and collision-free path using mixed integer linear programming. In [62], the authors recast the global optimization problem as several local problems, which are then iteratively solved by the agents in a decentralized way. In the Decentralized Model Predictive Control approach [63], the aircraft solve their own sub-problem one by one and send the action to other subsystems through communication.

There are also scenarios where communication cannot be reliably established (non-cooperative) and the aircraft will take action at each time step based on the sensor information. Many works fall in this category: Model Predictive Control [64, 65] can be used to solve the collision avoidance problem but the computation load is relatively high. Potential field method [66] is computationally fast. However, a navigation function [67] is required to deal with the local minima problem and make it a complete path planner [68, 69], which involves discretizing the state space. With the help of machine learning and reinforcement learning [70–74], collision avoidance algorithm (without trajectory re-planning to the destination) can have a promising performance, but the data collection and model training part are expensive. Using the Monte Carlo Tree Search algorithm to solve this problem [75] does not need model training and the algorithm can finish in any predefined computation time, but the aircraft can only adopt several discretized actions at each time step. In [76] the authors proposed a sample-based POMDP approximation algorithm that can run onboard for continuous state and observation spaces, which can find the optimal action using the branch and bound method. However, computation time was the most limiting factor in this work. Geometry based algorithms [77–80] can be also applied for collision avoidance problem and the computation time only grows linearly with the increasing number of aircraft. The drawback of these geometric approaches is that they cannot look ahead for more than one step (they only pay attention to the current action and do not take account of the effect of subsequent actions) and the outcome can be local optimal in the view of the global trajectory.

In this paper, under the free flight framework, we propose a computational guidance algorithm with a separation assurance capability, which is a message-passing based decentralized, reacting, and cooperative algorithm. We formulate this computational guidance problem as a Multiagent MDP (MMDP) and solve the formulated MMDP using the MCTS algorithm that can run onboard the aircraft. In fact, the proposed algorithm in this paper can be either centralized (where a centralized controller is responsible for gathering aircraft state information and issuing action advisories to all of the aircraft) or decentralized (where the algorithm runs onboard the aircraft and aircraft can coordinate with each other through wireless communication). We described both centralized and distributed cases in detail in our recent paper [81]. In this paper, we focus on the distributed case. There are similar works using MDP formulation which solve this problem offline in the pre-departure phase [25, 71, 82] or online during the en route phase [83]. Offline solvers require large computation time up front to compute the optimal policy for the full state space and discrete MDP formulations. Offline methods are typically not adaptive to changes in the environment because the policy is determined ahead of time. Also, the state space of many problems is too large to adequately represent as a finite set of enumerable states. Comparing

with offline methods, by using longer onboard computation time, onboard methods address the shortcomings of offline methods by planning only for the current state and a small number of possible plans. Since onboard algorithms only need to plan for the current state that can take any continuous value, state discretization of MDP formulation is not required. Onboard algorithms are also able to account for changes in the environment because they are executed once at each decision point, allowing for updates between these points. There is also POMDP formulation for this problem which aims to consider the state uncertainty due to the sensor noise [76], where the authors use a search algorithm to solve this POMDP. The major difference between this paper and our work is that they use depth-first search and do not consider the state transition uncertainty, while we use the robust MCTS algorithm to handle the dynamical model uncertainty through simulations.

C. Airspace Sectorization

Airspace sectorization is a concept widely used in commercial aviation. According to the FAA [84], the airspace sector is defined to be an airspace area with predefined horizontal and vertical dimensions for which a controller or group of controllers has air traffic control responsibility, normally within an air route traffic control center or an approach control facility. Sectors are established based on predominant traffic flows, altitude strata, and controller workload.

Airspace sectors can be created to deal with the high demand for aircraft traffic [85]. For example, in times when there are high levels of air traffic, more sectors may be opened with more controllers allocated to manage the aircraft within an area of airspace. This is done to maintain safety as a controller can only manage a certain number of aircraft at one time. In this paper, we use airspace sectors to reduce the computation time for the proposed computational guidance algorithm.

While the airspace sectorization can help reduce the computation time by distributing the workload to several sectors, it also introduces complexity by requiring additional coordination between aircraft and sector controllers, especially when an aircraft flies across more than one airspace sector.

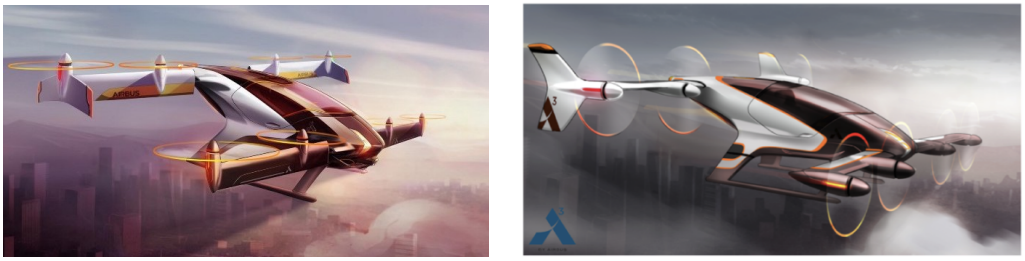
Note that the airspace sectorization also introduces additional "hand-off" conflicts between aircraft near the boundary between (at least) two sectors [86], since the adjacent sectors might have different plans to resolve the conflict and not able to see the traffic situation in the neighboring sector. This type of loss of separation deserves special attention because of the hand-off between two sectors. Several typical scenarios that could cause hand-off conflicts include poor or missing coordination/communication between sectors, aircraft flying along the sector boundary, and different minimum separation standards used in adjacent sectors. Also, adverse weather avoidance, communication equipment failure, high controller workload, transfer of control too early/too late, and sector skipping can contribute to conflicts between aircraft as well. For more details, readers can refer to [86].

In this paper, we will design a novel mechanism for airspace sectorization to resolve the hand-off conflict mentioned above.

D. Contributions

In this paper, we propose a message-passing based decentralized computational guidance algorithm with a separation assurance capability that can scale to multiple cooperative aircraft, where we formulate this problem as a Multiagent Markov Decision Process (MMDP) and solve this formulated MMDP using Monte Carlo Tree Search (MCTS) algorithm. We use a logit level-k model for the multi-aircraft coordination based on the message-passing mechanism through wireless communication. To achieve higher scalability, we introduce the airspace sector concept into the UAM environment by dividing the airspace into sectors, so that each aircraft only needs to coordinate with aircraft in the same sector. A high-density free-flight airspace simulator in the OpenAI Gym environment is built to validate and demonstrate the performance of the proposed algorithm. Through numerical experiments over several case studies in the free flight simulator with environment uncertainty, the proposed algorithm shows promising performance. Additionally, we also performed a stress test on the roundabout test problem, which consists of making a certain number of aircraft fly to the diametrically opposed point at a common speed on an annulus.

In the free flight airspace simulator, the aircraft dynamics are modeled based on the tandem tilt-wing eVTOL (Airbus Vahana) from Airbus A^3 [87] which has flown over 80 full-scale test flights [88]. Fig. 1 shows the take-off and landing phase and cruise phase of Vanaha aircraft. In this paper, we restrict our scope to the cruise phase of this aircraft in en route airspace. For the scheduling and spacing services in the vertiport terminal airspace (arrival and departure management), readers can refer to [89–91].



(a) take-off and landing phase **(b) cruise phase**

Fig. 1 Airbus Vahana with tandem tilt-wing configuration during the take-off phase and cruise phase [92].

Overall, this research proposes a potential solution to integrate the power of onboard aircraft autonomy and the free flight concept for airspace operations to enable safe and efficient flight operations in high-density urban air traffic.

The structure of the paper is as follows: in Section II, the background of MDP, MCTS and Multiagent MDP will be introduced. In Section III, the description of the problem and its mathematical formulation of Multiagent MDP are presented. Section IV presents the designed MCTS algorithm to solve this multiagent problem. The numerical experiment and results are shown in Section V. Section VI is the conclusion.

II. Background

In this section, we briefly review the background of the Markov Decision Process and Monte Carlo Tree Search, as well as the Multiagent Markov Decision Process.

A. Markov Decision Process (MDP)

Since the 1950s, MDPs [93] have been well studied and applied to a wide area of disciplines [94–96], including robotics [97, 98], automatic control [99], economics, and manufacturing [100]. In a MDP, the agent may choose any action a that is available based on current state s at each time step. The process responds at the next time step by moving into a new state s' with certain transition probability, and gives the agent a corresponding reward r.

More precisely, the Markov Decision Process (MDP) includes the following components:

- The state space $\mathcal{S}$ which consists of all the possible states.
- The action space $\mathcal{A}$ which consists of all the actions that the agent can take.
- Transition function $\mathcal{T}(s_{t+1}|s_t, a_t)$ which describes the probability of arriving at state s_{t+1}, given the current state s_t and action a_t.
- The reward function $\mathcal{R}(s_t, a_t, s_{t+1})$ which decides the immediate reward (or expected immediate reward) received after transitioning from state s to state s', due to action a. In general, the reward will depend on the current state, current action, and the next state. However, the reward function may only depend on the current state s_t, which will be the case in this paper.
- A discount factor $\gamma \in [0, 1]$ which decides the preference on immediate reward versus future rewards. Setting the discount factor less than 1 is also beneficial for the convergence of cumulative reward.

In a MDP problem, a policy π is a mapping from the state to one specific action (known as deterministic policy)

$$\pi : \mathcal{S} \to \mathcal{A} \tag{1}$$

The goal of MDP is to find an optimal policy π^* that, if followed from any initial state, maximizes the expected cumulative rewards over all the future steps:

$$\pi^* = \underset{\pi}{\operatorname{argmax}} \mathbb{E}[\sum_{t=0}^{T-1} R(s_t, a_t, s_{t+1})|\pi] \tag{2}$$

Q-function and value function are two important concepts in MDP. The optimal Q-function $Q^*(s, a)$ means the expected cumulative reward received by an agent starting in state s and picks action a, then will behave optimally afterwards. Therefore, $Q^*(s, a)$ is an indication of how good it is for an agent to pick action a while being in state s. The optimal value function $V^*(s)$ denotes the maximum expected total reward when starting from state s, which can be

expressed as the maximum of $Q^*(s,a)$ over all possible actions:

$$V^*(s) = \max_a Q^*(s,a) \quad \forall s \in \mathcal{S} \tag{3}$$

B. Monte Carlo Tree Search (MCTS)

Monte Carlo Tree Search (MCTS) is a method for finding optimal decisions in a given domain by taking random samples in the decision space and building a search tree according to the results [101, 102]. It has already had a profound impact on Artificial Intelligence (AI) approaches for domains that can be represented as trees of sequential decisions, particularly games and planning problems [103–105], including the current state-of-art computer program AlphaZero in the Game of Go [106].

The basic MCTS process is conceptually easy to understand, where a tree is built in an incremental and asymmetric manner, as shown in Fig. 2 (from [102]). For each iteration of the algorithm, a tree policy is used to find the most urgent node of the current tree. The tree policy attempts to balance considerations of exploration (look in areas that have not been well sampled yet) and exploitation (look in areas which appear to be promising). A simulation is then rolled out from the selected node and the search tree updated according to the result. This involves the addition of a child node corresponding to the action taken from the selected node and an update of the statistics of its ancestors. Moves are made during this simulation according to some default policy, which in the simplest case is to make uniformly random moves. A great benefit of MCTS is that the values of intermediate states do not have to be evaluated, as for depth-limited minimax search, which greatly reduces the amount of domain knowledge required. Only the value of the terminal state at the end of each simulation is required.

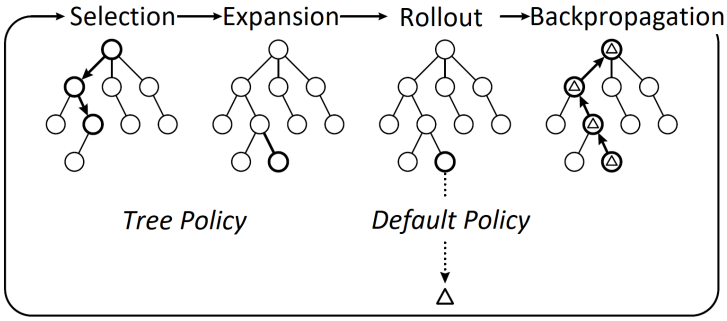


Fig. 2 One iteration of general MCTS approach [102].

C. Multiagent Markov Decision Process

Comparing with previous work [75] where the algorithm can only control one single aircraft, in this paper we focus on how the MCTS algorithm can scale to multiple cooperative agents.

In order to extend MDPs to multiagent settings, Boutilier [107–109] has introduced Multiagent Markov Decision Processes (MMDPs), which allow for representing sequential decision-making problems in cooperative multiagent settings. Similar to MDP, MMDP is defined as a tuple $< n, \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R} >$ [110] where

- n is the number of agents in the whole system.
- $\mathcal{S}$ is the set of states s.
- $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$ denotes the set of joint actions, where $\mathcal{A}_i$ is the set of individual actions of agent i.
- $\mathcal{T}$ is the transition function which gives the probability $\mathcal{T}(s_{t+1}|s_t, a_t)$ that the process moves to state s_{t+1} when the agents execute the joint action $a \in \mathcal{A}$ from state s_t.
- $\mathcal{R}(s_t, a_t, s_{t+1})$ is the reward obtained when the process state changes from s_t to s_{t+1} under the influence of joint action a_t.

Although MMDP can model the interactions among multiple agents, there are also many challenges in MMDP [111]:

1) The curse of dimensionality will cause exponential growth of the discrete state-action space in the number of state and action variables. For example, assume we have 10 agents, and each agent has 3 actions at each time step, then there will be 3^{10} different action combinations to consider at each time step, and each action could be the optimal one.
2) Nonstationarity arises in multiagent systems because each agent is facing a moving-target learning problem: the best policy changes as the other agents' policies change. For example, as shown in Fig. 3, without knowing the action of the other aircraft, we cannot tell which action is optimal since each action could lead to a LOS event between them. And even if we get the optimal action for both aircraft (e.g., both take action to turn right), the optimal action of one aircraft will be affected if the other aircraft changed their actions.

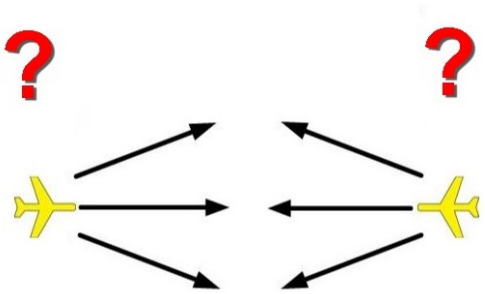


Fig. 3 The optimal action selected by one aircraft also depends on the action selected by the other aircraft.

Most of the literature studied multiagent systems in stochastic environments with a focus on Nash equilibrium [112] or long-term stable behaviors. But when we are dealing with real-time decision-making systems, information about

Nash equilibrium is not always helpful [113]. First of all, it may be unclear which equilibrium to adopt if there are many different equilibria in the system. For systems with only one equilibrium, it may be difficult to compute the Nash equilibrium when the computation time is limited [114]. An area known as behavioral game theory [115] aims to model agents that are limited in the number of steps of strategic look-ahead when the decision time is limited. Many different behavioral models exist, but the logit level-k model [116, 117] has become popular recently and tends to work well in practice. In the logit level-k model, an agent at logit level-k assumes all of the other agents follow logit level-$(k-1)$ strategy. In this way, the actions of other agents become fixed thus the tree search process can avoid the action explosion issue in the multi-agent case.

In this paper, we use a variant of the logit level-k model to make decisions for all of the agents in a fast, dynamic and uncertain environment where real-time onboard decisions are needed.

III. Problem Formulation

A. Problem Statement

The goal of this research is to develop a distributed algorithm that runs on each aircraft to provide tactical guidance commands. These commands will help each aircraft arrive at their respective destinations while avoiding potential LOS events between them during the flight. In this paper we assume a free flight concept of operations (ConOps), where each flight's trajectory is free of airspace constraint. Here we investigate the feasibility of the free flight ConOps and push its safety limits with new tools from MMDP and MCTS. However, we would like to identify other approaches and ConOps such as pre-departure flight planning for strategic deconfliction, and/or implementing structured airspace to ensure higher level of safety. In the MMDP formulation, at each decision point, the vehicle's action is decided directly from the state, which incorporates all the information (the position and velocity of all the intruder aircraft and the destination/goal of ownship) to decide which action is optimal for the corresponding state.

When controlling the aircraft, only horizontal actions are considered in this paper, which means all the aircraft will be flying at the same altitude and this problem can be solved in two dimensions. This assumption makes it possible to incorporate multiple flight levels to deal with the high-density air traffic in UAM.

The objectives for this specific MMDP problem are two-fold: the first is to avoid potential LOS events among all the moving aircraft, the second is to guide all the aircraft to their destinations as quickly as possible during the flight. Therefore, the reward function needs to capture both two objectives.

Based on the above description, this problem will be mathematically formulated as a MMDP problem in the next subsection.

B. MMDP Formulation

The MMDP formulation in this paper is similar to our previous work [75] and here we will briefly introduce it.

1. State Space

The state includes the necessary information for the algorithm to issue actions to all the aircraft: the position, speed, heading angle, and goal position for all of the aircraft. More specifically, the state for one aircraft is $(x, y, v, \psi, g_x, g_y)$, where $(x, y), v, \psi$ are the position, speed, and heading angle for the aircraft, and (g_x, g_y) is the goal position for this aircraft. Stacking the information for all of the controlled aircraft, the state space for the MMDP becomes a $n \times 6$ matrix, where n is the number of aircraft and each row of the matrix represents the information for one aircraft.

Note here the state space is continuous (e.g., all the variables of a state can take continuous values). In general, for a MDP with continuous state variables, it is not clear how to best represent the policy, since it is impossible to enumerate all possible state-action mappings. For previous MDP-based algorithms to solve conflict avoidance problems, some possible approaches to represent the policy include using a grid-based discretization of the state space $\mathcal{S}$ and the action space $\mathcal{A}$ [30, 71] or using policy compression techniques [118, 119]. The advantage of the MCTS algorithm is that it does not need to discretize the state space. For each state, the MCTS algorithm will generate action onboard for the aircraft to follow in real-time.

2. Action Space

At each time step (5 seconds), the aircraft can choose to turn its heading at a certain rate. More precisely, the advisory of heading angle for each aircraft constitutes the action set $\mathcal{A} = \{-5^\circ/s, 0^\circ/s, +5^\circ/s\}$ where positive corresponds to right turn and negative to left turn. The changing rate of heading angle is determined assuming the aircraft is flying with cruise speed at $190 km/h$ [88] and banking angle at 25° (the banking angle limit is chosen to be 25° for the passenger comfort consideration). At each time step, the proposed algorithm will run onboard to issue one action from the action set for the aircraft based on the current state. After the algorithm running, the aircraft will maintain the issued action during this time step.

It's natural to consider extending the set of actions (conflict resolution advisories) to include more options such as vertical resolution and speed resolution. However, using the MCTS algorithm to calculate the optimal action will be more time consuming with the extended action space since the tree size will grow exponentially with the number of actions. Because computation time is an important factor for the real-time onboard algorithm, some techniques are necessary for extending action space in future steps, such as truncated Monte Carlo search algorithms [120], progressive strategies for MCTS [121–124], or using a policy network to narrow down the search to high-value actions [103]. In this paper, we use this defined action space to keep our scope more focused.

3. Dynamical Model

Based on the current state and current action, Dubin's kinematic model will be used to compute state transition for each aircraft:

$$\dot{x} = v\cos\psi \tag{4}$$

$$\dot{y} = v\sin\psi \tag{5}$$

$$\dot{\psi} = a_{\psi} \tag{6}$$

where v is the cruise speed, ψ is the heading angle, and a_{ψ} is the selected action describing the changing rate of heading angle for one aircraft. Following the aircraft performance data from Airbus Vahana [88], the cruise speed of the aircraft is set to $190km/h$, and we restrict the speed to be in between $v_{\min} = 165km/h$ and $v_{\max} = 220km/h$ [87, 88] since there is uncertainty in the speed.

After an aircraft executes an advisory, the aircraft speed is held constant between decision stages with uncertainty, which is modeled as a Gaussian distribution centered on the aircraft's cruise speed with a standard deviation of $5m/s$. The changing rate of heading angle distribution is centered on the resolution advisory with a standard deviation of $2^{\circ}/s$. The noises here aim to account for the uncertainties in the environment and aircraft dynamics. In previous works [71, 125], the uncertainties are modeled as Gaussian distributions with a standard deviation of $2m/s$ for speed and 3° for heading angle when the banking angle is less than 25°. In this paper we increase the standard deviation for speed to $5m/s$ and lower the standard deviation for change of heading angle to $2^{\circ}/s$ to better model the uncertainties since Vahana aircraft is flying at a higher speed.

4. Reward Function

The focus of our computational guidance system is to achieve the dual objectives of maintaining safety while guiding the aircraft to their destinations as quickly as possible. These objectives are captured in a reward function composed of a sum of the reward function for each individual aircraft.

For the consideration of safety, the LOS event is defined to be when the distance of two aircraft is less than a minimum separation distance $r^{min} = 0.5$ nautical miles [126]. This separation standard was chosen using the definition of well clear for UAS according to Cook and Brooks [127]. For large UAS in high-altitude airspace, the Horizontal Miss Distance (HMD) is defined to be 0.66nmi. For small UAS (55lbs vehicle or less) in low-altitude controlled airspace around airports, the horizontal separation is set to be a HMD of 0.36nmi. Using those values as reference, the nominal spatial separation standards picked for this UAM application are set to 0.5nmi horizontally. This value is tighter than UAS standards because it is assumed that enhanced equipage capabilities will be installed on the eVTOL aircraft [126].

Based on the above separation requirements, we define the following two different types of states:

- The distance between two aircraft is less than r^{min} (referred to as a LOS state in the following);
- The aircraft reaches the goal position (referred to as a goal state in the following).

With the two types of states defined above, the reward function for one aircraft is defined as follows:

$$r(s) = \begin{cases} 1, & \text{if } s \text{ is goal state,} \\ 0, & \text{if } s \text{ is LOS state,} \\ 1 - \frac{d(o,g)}{\max d(o,g)}, & \text{otherwise.} \end{cases} \tag{7}$$

where $d(o, g)$ denotes the distance from the aircraft to its goal position. $\max d(o, g)$ is the maximum distance from an aircraft to its goal position, which is the diagonal distance of the map (if the map has an irregular convex shape, we can use the diameter of this convex shape). In this way, if an aircraft is not at LOS state (which has reward 0), this aircraft will get a positive reward between 0 and 1, depending on how far this aircraft is from the goal state.

The reward function for one aircraft is normalized to range $[0, 1]$ for the following reason. In MCTS, the algorithm will keep selecting the action with max value:

$$\bar{r}_j + 2C\sqrt{\frac{2\ln n}{n_j}} \tag{8}$$

where $\bar{r}_j$ is the mean reward value for action j, n is the number of times the current state has been visited, and n_j is the number of times action j has been selected during the process for building the search tree. In Eq. (8), the first term describes the average value for an action from previous information, the second term measures the uncertainty of this action, and the coefficient C can balance these two terms. It should be noted that the value of C depends on the reward scale $\bar{r}_j$. The reward function is normalized to range $[0, 1]$ since for this reward range, the value of $C = 1/\sqrt{2}$ was shown by Kocsis and Szepesvari to satisfy the Hoeffding inequality [128]. With a reward range different than $[0, 1]$, a different value of C will be needed.

After we have the reward function definition for one aircraft, the reward function for the MMDP is defined to be the sum of reward for each aircraft:

$$R(s) = \sum_i r_i(s) \tag{9}$$

where $r_i(s)$ is the reward function for aircraft i defined in Eq. (7).

With this reward setting, when we solve the formulated MMDP by maximizing the reward, all of the aircraft will try to select action leading to a state that is closer to the goal position (which has positive reward) and avoid any LOS states (which has 0 reward). Since non-LOS states are always preferred than LOS states in the above reward setting, we do not need to introduce a penalty (such as a negative reward) for LOS states.

IV. Solution Method

In this paper, we will use the most popular algorithm in the MCTS family, the Upper Confidence Bound for Trees (UCT) to solve the MMDP problem formulated above. The details of UCT algorithm implementation for the single aircraft case can be found in [75]. As formulated in Section III, computing the global optimal solution is impractical for more than a few aircraft since the state space will grow linearly and the action space will grow exponentially with the increasing number of aircraft. Therefore we use a variant of logit level-k model [116, 117] to solve the above issue. In the logit level-k model, an agent at logit level-k assumes all of the other agents follow logit level-$(k-1)$ strategy. In this way, the actions of other agents become fixed thus the tree search process can avoid the action explosion issue in the multi-agent case.

Cooperative Sequential Decision Making

In the logit level-k model, a level-0 agent selects actions randomly by following a uniform distribution. A level-1 agent assumes that all the other agents adopt level-0 strategies and selects actions according to the logit distribution

$$P(a_i) \propto e^{Q_1^*(s,a_i)} \tag{10}$$

where $Q_1^*(s, a_i)$ is the Q-function for the state-action pair (s, a_i) of a level-1 agent assuming other agents follow level-0 strategies. A level-k agent assumes that the other agents adopt level-$(k-1)$ strategies and select their own actions according to Eq. (10).

In this paper we use a deterministic variant of the logit level-k model. More specifically, a level-0 aircraft selects the action to fly straight, and a level-1 aircraft assumes other aircraft adopt the level-0 strategy and selects action a^* with max Q value:

$$a^* = \underset{a}{\operatorname{argmax}} Q_1^*(s, a_i) \tag{11}$$

where the Q-function $Q_1^*(s, a_i)$ is approximated from the MCTS algorithm.

The deterministic logit level-k model presented above updates the action for the next level in a synchronous way (e.g., the aircraft at the same level do not know the actions among each other). As shown in Fig. 3, knowing the action of other aircraft is helpful for the action selection, therefore in this paper we update the action for the next level in an asynchronous way.

More specifically, suppose currently we have n aircraft flying in the air. Then at the beginning of the search algorithm when all the aircraft are at level 0, the joint actions $\mathbf{a}$ for all aircraft are initialized to 0 at first:

$$\mathbf{a} = \{a_1, a_2, \cdots, a_n\} \tag{12}$$

where a_i is the action for aircraft i and $a_1 = a_2 = \cdots = a_n = 0^\circ/s$.

Then starting from the first aircraft, each aircraft will run the algorithm onboard by building a search tree, assuming all of the other aircraft will take action according to the joint action **a** and follow the dynamical model described in Section III. B. 3. The searching process is similar to the process described in [75] and the difference is all aircraft can turn according to the joint action.

Next, assume the tree search result for the first aircraft is a_1^*, then the first aircraft will send this action information a_1^* to all the remaining aircraft through wireless communication and the joint action will be updated as follows:

$$\mathbf{a} = \{a_1^*, a_2, \cdots, a_n\} \tag{13}$$

Note here only the first aircraft is at level 1 and all the other aircraft are still at level 0. Then after receiving the action information from the first aircraft, the second aircraft begins running the MCTS algorithm onboard assuming all of the aircraft are following the most recent joint action. This process will iterate over all the aircraft until all the $a_1, a_2, \cdots, a_n$ are updated, when all the aircraft are at level 1. We can keep updating the actions for all of the aircraft to higher levels. Since numerical experiment results do not show much performance improvement for level 2 comparing with level 1, for the consideration of computation time, in this paper we stop this process after reaching level 1. After we have the level 1 joint action, all of the aircraft will execute this joint action for 5 time steps, after which all of the aircraft will run the algorithm again to generate new joint action.

V. Numerical Experiments

A. Simulator

To test the performance of the proposed algorithm, a simulator was built in Python where multiple aircraft can fly freely in the two dimensional en route airspace above New York City. We envision there will be multiple altitude levels where the eVTOL aircraft are operated. In the scope of this paper, we only focus on one altitude level (a two dimensional environment).

To validate the performance of this algorithm in real world applications, we will simplify the UAM network by following the generic city model presented in [129, 130]. In this generic city model, seven vertiports are distributed in a "six around one" hexagonal pattern. As shown in Fig. 4, vertiport 1 is at in the center of the hexagon and located equidistant from the other six vertiports at a distance of 16km, which will cover the main congestion area of New York City. Overlays of the vertiport network are shown on a Google map image of New York in Fig. 4, which shows typical New York traffic on a Friday at 5 pm [131].

In real world scenarios, the arrival and departure routes of large, conventional aircraft in terminal airspace near airports [132], the presence of flight restrictions and restricted airspace [133], and the noise of the eVTOL vehicles [2]

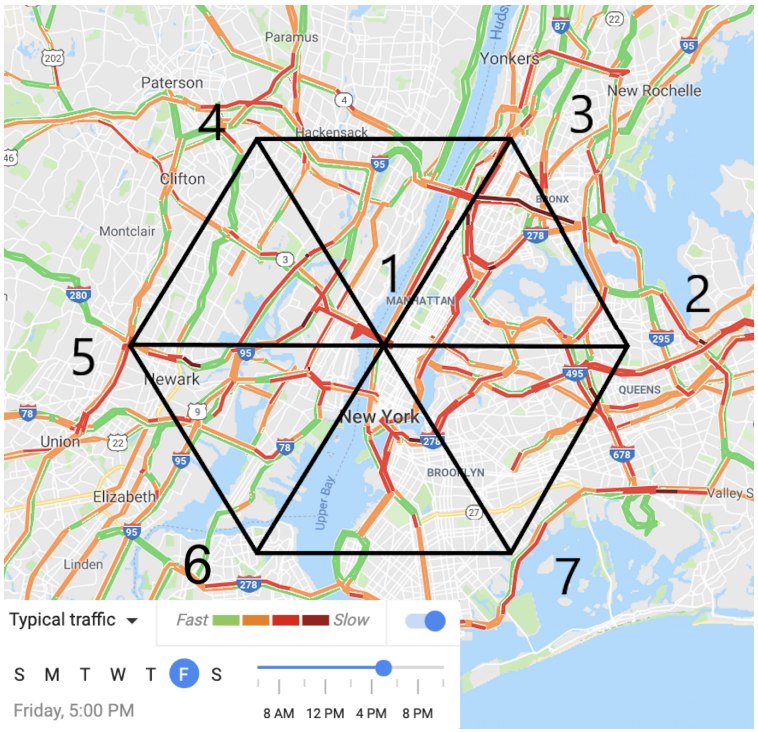


Fig. 4 Network of seven vertiports overlaid on New York city with segment length 16km.

may all limit the UAM airspace shape and restrict the eVTOLs operations between vertiports. The proposed algorithm in this paper does not work under such scenarios yet, especially when the restricted airspace is in an irregular shape. A direction of future work would be to adapt the proposed algorithm to incorporation various airspace restrictions to make it more practical.

B. Airspace Sectorization

In the numerical experiment, the result shows the computation time for the algorithm grows linearly with the increasing number of aircraft, which makes it intractable for real-time guidance when the number of aircraft is over 15. To mitigate this issue, in this paper we introduce the concept of airspace sectorization to reduce computation time for the proposed algorithm by distributing the coordination workload to different sectors. As shown in Fig. 5, the airspace with 7 vertiports is divided into 7 hexagonal sectors and the center of each hexagonal sector is one vertiport. One advantage of this hexagonal sector configuration is that hexagons can form a tessellation of a two dimensional airspace, which means this sector configuration can be easily extended to larger airspace.

In this sector setting, each aircraft only needs to coordinate with other aircraft in the same sector. More specifically, the aircraft in one sector only gather information of other aircraft in the same sector and the decision-making process only depends on the gathered information. Every 5 seconds, aircraft in the same sector will simulate action and communicate the action information according to the default order (which can be based on the time when the aircraft enters this

sector). With this sector setting, ideally we can reduce the computation time by a factor of 7.

While the sector configuration can help reduce the computation time for the proposed algorithm, it also introduces a new type of conflict called hand-off conflict: when the aircraft is crossing/flying near a boundary of the sector, there is higher chance for conflicts to happen since the aircraft does not know the aircraft information from other sectors. To resolve this problem, we introduce one-directional gates on the boundary of sectors, which is denoted as orange rectangles in Fig. 5 and the direction of the gate is denoted using arrows. The aircraft is only allowed to exit through the available gates and exiting from other locations will be regarded as a LOS state with reward 0, which the algorithm will try to avoid. The width of the gates is designed to be 1,800m, so as to allow two aircraft passing the gates simultaneously. When an aircraft enters a new sector or takes off from a vertiport, it will be assigned an exit gate for this aircraft (if the goal vertiport of this aircraft is not in the current sector) by minimizing the total path:

$$\min\{\sqrt{(p_x - e_x)^2 + (p_y - e_y)^2} + \sqrt{(e_x - g_x)^2 + (e_y - g_y)^2}\} \tag{14}$$

where $(p_x, p_y), (e_x, e_y), (g_x, g_y)$ are the position for aircraft, exit gate, and the goal of the aircraft. After assigning the exit gate for this aircraft, the onboard algorithm will guide the aircraft toward its assigned exit gate.

To further reduce the risk of conflicts, the aircraft in one sector can also sense the aircraft information from other sectors that are close to this sector (if the distance between the aircraft and the sector is smaller than 1,500m), but the aircraft does not receive any action information (or flight intention) from them. In summary, each aircraft will receive full information (state and action information) from other aircraft in the same sector, partial information (only state information) from other aircraft that are close to its own sector, and no information from all the remaining aircraft.

C. Parameter Settings

In the proposed algorithm there are two parameters that can impact the performance of the MCTS algorithm: the number of simulations and the search depth. The number of simulations means the number of roll-outs to simulate during the tree search process. Since there is uncertainty in the environment and aircraft dynamics, more roll-outs will cover more cases and make the algorithm more robust. Search depth means how many steps to look ahead. A more detailed definition can be found in [75]. Typically for the MCTS algorithm, a larger search tree (more simulations and deeper search depth) can lead to better performance of the algorithm but need longer computation time. In previous work [75], it is found that setting number of simulations to 100 and search depth to 3 in this problem can have decent performance in terms of the number of LOS events/NMACs, average en route flight time, and average onboard computation time. Thus in this paper we adopt the same parameter setting. We also noticed that when an aircraft is far from the other aircraft, a smaller tree is enough to find the good action. So to speed up the algorithm, when the distance of an aircraft to its closest aircraft is larger than 2 km, we set the number of simulations to 30 and search depth to 2.

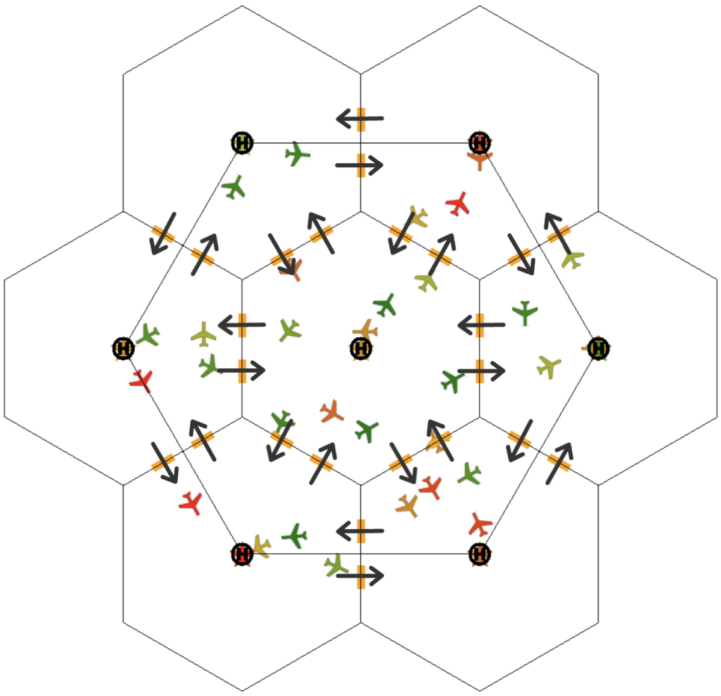


Fig. 5 The airspace is divided into 7 sectors to reduce the computation time for the proposed algorithm.

D. Case Studies and Results

Based on the above simulator setting, we conducted the following three case studies to illustrate the performance of the proposed algorithm.

1. Case study I - On-demand air transportation

In the first case study, based on the airspace configuration we define a demand model that generates flight requests stochastically. At each vertiport, after the taking off of the previous aircraft, the time interval for next aircraft to take off is uniformly distributed between 1 minute and 2 minutes and the newly generated flight request will choose a random vertiport as its destination. Then the simulator will be kept running until 10,000 aircraft have been generated. During the running of this simulator, the number of LOS events and NMACs (short for near mid-air collision) and the average computation time to make each decision will be recorded and compared. Here the NMAC standard is defined to be 500 feet by the Aeronautical Information Manual (7-6-3) [134].

Then we conducted 5 independent experiments and there are 10,000 aircraft generated in total in each experiment. The code implementation of this algorithm is available on GitHub * and a short video demo for this case study can be found on YouTube †.

Table 1 shows the average number of LOS events and NMACs per flight hour and standard error from the simulation

*https://github.com/xuxiyang1993/Multi_MCTS_Guidance_Separation_Assurance
†https://www.youtube.com/watch?v=2cbRUig4G_I

results over 5 independent experiments with and without sector configuration (referred to as sectored airspace and unsectored airspace in the following). From this table we see that the LOS event happens around 1×10^{-2} per flight hour and NMAC happens around 1×10^{-4} per flight hour. This table shows the proposed algorithm has promising performance for guidance and separation assurance, and the introduction of the airspace sector help reduce the LOS/NMAC risk for aircraft. Note that the recorded number of NMACs during simulation is in the absence of a collision avoidance system such as TCAS or ACAS-X. A direction of future work would be to integrate our separation assurance model with a collision avoidance system as the final layer of protection. Furthermore, all results shown in this work have not been integrated with any strategic flight plan deconfliction, traffic flow management, or flow control. We expect better safety performance once integrating these components.

Table 1 The simulation results of MCTS algorithm averaged over 5 independent experiments.

	average LOS per flight hour	average NMACs per flight hour
sectored airspace	$(1.05 \pm 0.34) \times 10^{-2}$	$(1.45 \pm 2.90) \times 10^{-4}$
unsectored airspace	$(1.31 \pm 0.22) \times 10^{-2}$	$(2.91 \pm 3.57) \times 10^{-4}$

Fig. 6 plots the computation time needed for all the aircraft to finish running the onboard algorithm to decide the joint actions. This shows the computation time for both unsectored airspace and sectored airspace are growing with the increase of the number of aircraft, and the sectored airspace greatly reduces the computation time to real-time levels. For the 30 aircraft case, the algorithm with airspace sectorization only takes less than 500 ms to issue the actions for all the aircraft. Note here we record the longest computation time among the 7 sectors. Since most of the time the aircraft are not evenly distributed in the 7 sectors, the computation time does not achieve 7 times faster and the variance among 5 independent experiments is larger than that in unsectored airspace.

Note here that the MCTS algorithm is a statistical anytime algorithm [102]. This means the algorithm can stop running at anytime by returning the current best action, and longer computation time generally leads to better action advisory. This is beneficial for the case when an aircraft needs a maximum acceptable run time.

Although the introduction of airspace sectorization help increase the safety level and reduce the computational time of the proposed algorithm, it also requires metering the aircraft to the exit gate of each sector which will force the vehicles to deviate from their preferred optimal route and result in suboptimal trajectories. Thus in the numerical experiment we also studied the en route flight time in sectored airspace and unsectored airspace. Here we divided the route option into 3 categories based on the distance between the start vertiport and target vertiport. Recall in Fig. 4 we labeled the vertiports with different ID numbers. In this vertiport map, route 1 is the shortest (e.g., from vertiport 1 to vertiport 2), route 2 has the medium length (e.g., from vertiport 3 to vertiport 7), and route 3 has the longest path (e.g., from vertiport 3 to vertiport 6). The three different routes are plotted in Fig. 7. Table 2 shows the en route flight time of sectored airspace and unsectored airspace for different route options assuming there is no intruder aircraft. From this

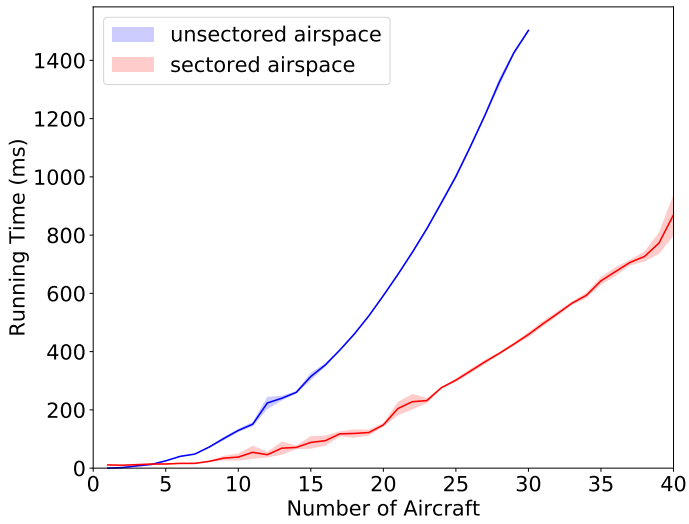


Fig. 6 The computation time with increasing number of aircraft for the unsectored airspace and sectored airspace.

table we can see for route 1, 2, and 3, the airspace sectorization increases the flight time by 6s, 20s, and 7s respectively, which is acceptable in trade of safety level and computation time. For route 1 and route 3, the difference is small since the deviated trajectory in sectored airspace is close to the preferred optimal trajectory in unsectored airspace.

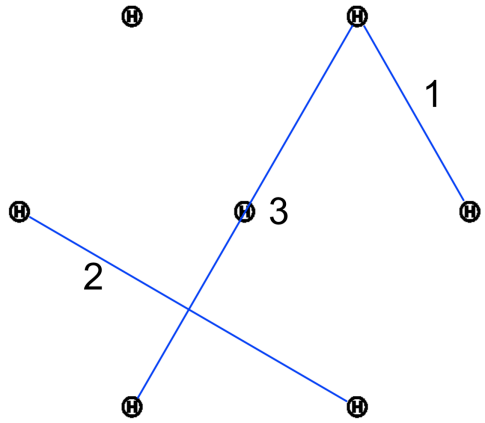


Fig. 7 Three different routes in the above vertiport setting.

Table 2 Flight time (en route air time) in seconds for sectored airspace and unsectored airspace.

route	1	2	3
sectored airspace	340.60 ± 7.91	606.56 ± 10.58	685.71 ± 10.81
unsectored airspace	334.46 ± 7.48	586.43 ± 9.88	678.81 ± 10.75

In Fig. 8, we recorded the number of en route aircraft at each time step and plotted the resulting histogram. In this plot, x-axis is the total number of en route aircraft and y-axis denotes their frequency. From this figure we can see most of the time, there are $25 \sim 40$ aircraft flying in the air, which is approximately $22.55 \sim 36.08$ aircraft per 10,000 km^2.

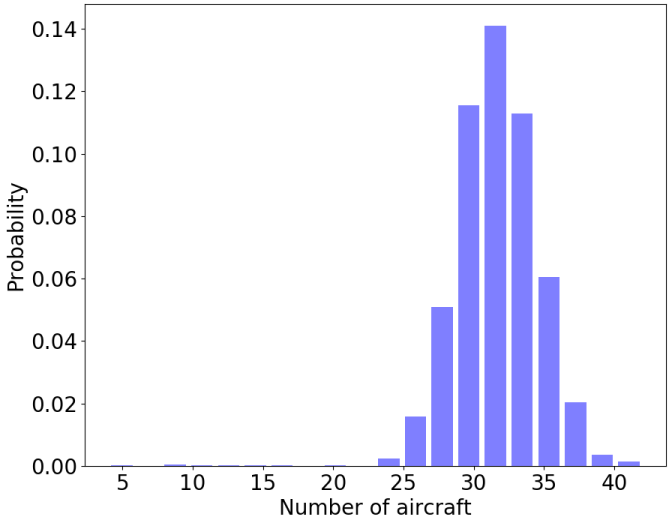


Fig. 8 The histogram for the total number of en route aircraft.

2. Case study II - eVTOL with different priority

In the second case study, we further analyze the en route flight time for the aircraft with priority, since in some cases we may have an emergency flight request which needs to reach its destination in the shortest possible time (e.g., a law enforcement eVTOL, or an ambulance eVTOL). This case can also happen when an aircraft flying in the air has a fault in its electric propulsion system. To let these emergent aircraft arrive goal in a shorter time, in this experiment we divide all the aircraft into two categories: each aircraft is classified into high/low priority, which is used to denote the emergency level for an aircraft. When an aircraft takes off, the priority will be randomly assigned to the aircraft in the numerical experiment. In real world applications, an eVTOL will be identified and approved with different priorities depending on its trip purpose.

When the algorithm makes decisions for the aircraft, it will first generate actions for all the aircraft with high priority, by only considering the high priority aircraft information. Then it will next generate actions for all the remaining aircraft, given the actions made for the high priority aircraft.

Specifically, suppose in the current sector we have n aircraft, out of which p aircraft are with high priority. Then the algorithm will first initialize the actions for these p aircraft:

$$\mathbf{a}_{\text{high}} = \{a_1, a_2, \cdots, a_p\} \tag{15}$$

where a_i is the action for aircraft i and $a_1 = a_2 = \cdots = a_p = 0°/s$. The algorithm will generate actions based on the high priority aircraft information $[s_1, \cdots, s_p]$ where s_i denotes the state information for aircraft i. After generating the optimal actions for these p high priority aircraft we get

$$\mathbf{a}^*_{\text{high}} = \{a^*_1, a^*_2, \cdots, a^*_p\} \tag{16}$$

The algorithm will then next initialize the joint action for all the remaining aircraft as

$$\mathbf{a}_{\text{low}} = \{a^*_1, \cdots, a^*_p, a_{p+1}, \cdots, a_n\} \tag{17}$$

where $a_{p+1} = \cdots = a_n = 0°/s$. Then the algorithm will generate actions for these aircraft with all of the aircraft information $[s_1, \cdots, s_n]$.

In this experiment, we compare the average en route flight time of aircraft with different priorities for the 3 different routes, which is recorded in Table 3. In this table we also added the optimal en route flight time where there is no intruder aircraft, from which we can see the flight time of high priority aircraft is closer to the optimal path flight time comparing with low priority aircraft, and high priority aircraft can save 14s, 34s, 43s en route time for each route comparing with low priority aircraft.

Table 3 Flight time (en route air time) in seconds for high priority aircraft and low priority aircraft.

route	1	2	3
clear path	340.60 ± 7.91	606.56 ± 10.58	685.71 ± 10.81
high priority	344.93 ± 0.32	619.98 ± 0.70	703.14 ± 1.31
low priority	358.80 ± 0.83	653.38 ± 0.49	746.51 ± 2.32

3. Case study III - Roundabout stress test

In this case study, the proposed algorithm is evaluated against a stress-test set of multi-threat scenarios randomly generated from an encounter model, similar to the experiments presented in [71]. This case study is helpful since a large part of current commercial interest surrounding UAM stems from the potential to accommodate a large number of eVTOL aircraft and it is estimated there will be 23,000 aircraft flying major routes within the UAM network by 2035 [6].

In each encounter scenario, the number of aircraft in the multi-threat encounters is ranging from 10 to 20, distributing uniformly on an annulus. Specifically, the annulus had inner and outer radii of 10km and 15km, and if a new aircraft added is closer to other aircraft than 2km, we resample the new aircraft position to avoid initializing aircraft already in the LOS state. The goal position of each aircraft is set to be the symmetric point in the annulus with respect to this aircraft's position, so the headings of each aircraft are initialized to point straight towards the annulus center to ensure

that all aircraft would have potential conflicts. Figure 9 shows an example stress test scenario with 10 aircraft where the positions were initialized uniformly in the annulus.

Fig. 10 illustrates the performance of the proposed computational guidance algorithm in the stress test scenarios as the number of aircraft increases, where each point denotes the average result in 1,000 independent encounter scenarios. It plots the probability that an aircraft has LOS/NMAC with another aircraft in one scenario. From Fig. 10a we can see for the MCTS algorithm the LOS event probability is less than 1% and the NMAC probability is less than 0.1% for each aircraft. We also show the result of the baseline where no actions are taken for all of the aircraft (e.g., all of the aircraft are flying straight towards their respective goals), and plot the average result over 1000 independent experiments. To ensure the aircraft can arrive at their respective goals, we remove the uncertainty for the change of heading angle and only keep the uncertainty of speed. From Fig. 10b we can see when no actions are taken, over 90% of the aircraft will have at least one LOS event and over 40% of the aircraft will have NMAC with other aircraft in each stress test encounter scenario. The comparison of the two plots in Fig. 10 shows the promising performance of the proposed computational guidance algorithm even with high-density air traffic.

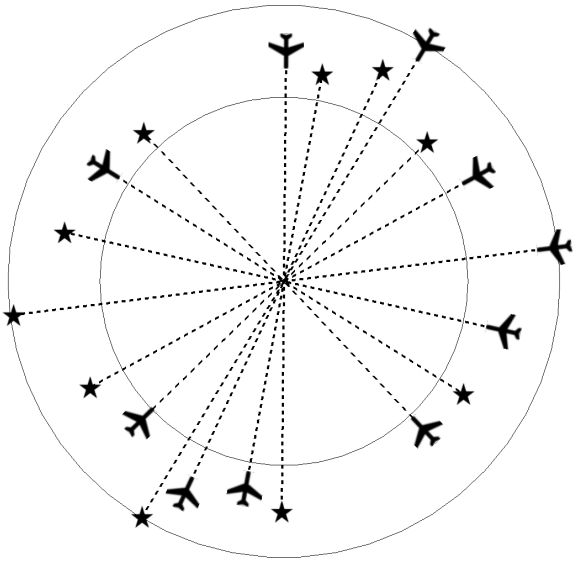
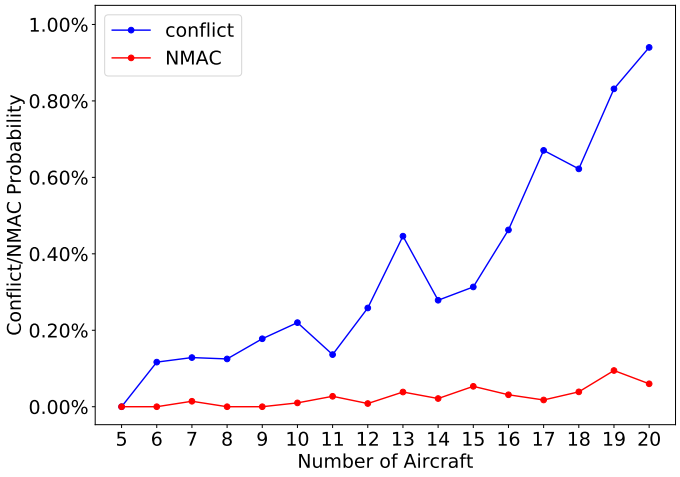


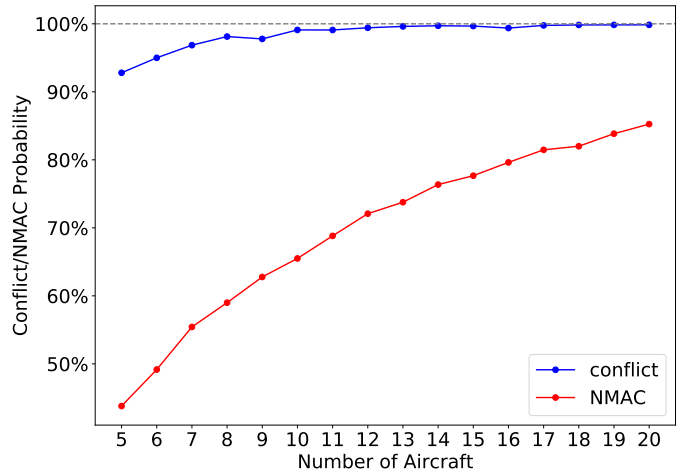
Fig. 9 One randomly generated stress test scenario. Each aircraft and its destination are connected through dashed line.

VI. Conclusion

A message-passing decentralized computational guidance algorithm with a separation assurance capability for multiple cooperative aircraft in urban air mobility is proposed in this paper. The problem is formulated as a Multiagent Markov Decision Process (MMDP) and then solved by Monte Carlo Tree Search (MCTS) algorithm that can run



(a) MCTS algorithm



(b) Baseline

Fig. 10 LOS event/NMAC probability as the number of aircraft increases.

onboard. A message-passing based coordination mechanism was designed to manage multiple cooperative aircraft. The airspace sectorization is introduced to help to achieve better scalability and safety. Numerical experiments over three case studies show that this proposed algorithm has promising performance to help the aircraft reach their destinations and avoid potential LOS events among them even for the high-density air traffic case. The contributions of this paper includes: (1) exploring the feasibility of the free flight concept of operations for urban air mobility; (2) designing the multi-agent computational guidance algorithm with a separation assurance capability; and (3) proposing airspace sectorization in urban air mobility.

The proposed concept of operations and algorithm provide a potential solution for distributed separation assurance

to enable safe, efficient, and scalable flight operations in on-demand urban air transportation with high-density air traffic. We found this framework might be a potential solution for certain airspace such as rural or suburban areas. However, the proposed framework is still in the exploratory phase with some limitations. To make the proposed algorithm more practical in real world applications, the future work should include (1) expanding the action space to altitude changes and speed changes; (2) adapting the proposed algorithm in restricted airspace or structured airspace; (3) incorporating a high-fidelity aircraft dynamics model with more realistic mission profile; and (4) integrating other layers of protection into this framework such as strategic flight planning, flow control, and collision avoidance systems.

Acknowledgments

This research is partially supported by the National Science Foundation Grant No. 1565979.

References

[1] Gipson, L., "NASA Embraces Urban Air Mobility, Calls for Market Study," https://www.nasa.gov/aero/nasa-embraces-urban-air-mobility, 2017. Accessed: 2020-02-15.

[2] Thipphavong, D. P., Apaza, R., Barmore, B., Battiste, V., Burian, B., Dao, Q., Feary, M., Go, S., Goodrich, K. H., Homola, J., et al., "Urban air mobility airspace integration concepts and considerations," *2018 Aviation Technology, Integration, and Operations Conference*, 2018, p. 3676. doi:https://doi.org/10.2514/6.2018-3676.

[3] Moore, M., "Uber elevate: eVTOL urban mobility," *Rotorcraft Business & Technology Summit*, 2017.

[4] Holden, J., and Goel, N., "Fast-Forwarding to a Future of On-Demand Urban Air Transportation," *San Francisco, CA*, 2016.

[5] "Future of Urban Mobility," https://www.airbus.com/newsroom/news/en/2016/12/My-Kind-Of-Flyover.html, 2017. Accessed: 2020-02-15.

[6] Porsche Consulting study, "the future of vertical mobility," https://www.porsche-consulting.com/fileadmin/docs/04_Medien/Publikationen/TT1371_The_Future_of_Vertical_Mobility/The_Future_of_Vertical_Mobility_A_Porsche_Consulting_study__C_2018.pdf, Mar 2018. Accessed: 2020-02-15.

[7] Mueller, E. R., Kopardekar, P. H., and Goodrich, K. H., "Enabling Airspace Integration for High-Density On-Demand Mobility Operations," *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 3086. doi: https://doi.org/10.2514/6.2017-3086.

[8] FAA, "NextGen implementation plan 2016," U.S. Department of Transportation, 2016.

[9] Gawdiak, Y., Carr, G., and Hasan, S., "JPDO case study of NextGen high density operations," *9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) and Aircraft Noise and Emissions Reduction Symposium (ANERS)*, 2009, p. 6918. doi:https://doi.org/10.2514/6.2009-6918.

[10] Timar, S., Hunter, G., and Post, J., "Assessing the benefits of NextGen performance based navigation (PBN)," *10th USA/Europe Air Traffic Management Research and Development Seminar, Chicago, Illinois*, 2013.

[11] Moore, M. D., and Goodrich, K. H., "High speed mobility through on-demand aviation," *2013 Aviation Technology, Integration, and Operations Conference*, 2013, p. 4373. doi:https://doi.org/10.2514/6.2013-4373.

[12] Gawdiak, Y., Holmes, B., Sawhill, B., Herriot, J., Ballard, D., Creedon, J., Eckhause, J., Long, D., Hemm, R., Murphy, C., et al., "Air transportation strategic trade space modeling and assessment through analysis of on-demand air mobility with electric aircraft," *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2012, p. 5594.

[13] Zhu, G., and Wei, P., "Pre-Departure Planning for Urban Air Mobility Flights with Dynamic Airspace Reservation," *AIAA Aviation 2019 Forum*, 2019, p. 3519.

[14] Force, R. T., "Final report of RTCA task force 3: Free flight implementation," Tech. rep., RTCA Inc., Tech. rep, 1995.

[15] Hoekstra, J. M., van Gent, R. N., and Ruigrok, R. C., "Designing for safety: the 'free flight' air traffic management concept," *Reliability Engineering & System Safety*, Vol. 75, No. 2, 2002, pp. 215–232. doi:https://doi.org/10.1016/S0951-8320(01)00096-5.

[16] Bilimoria, K. D., Grabbe, S. R., Sheth, K. S., and Lee, H. Q., "Performance evaluation of airborne separation assurance for free flight," *Air Traffic Control Quarterly*, Vol. 11, No. 2, 2003, pp. 85–102. doi:https://doi.org/10.2514/atcq.11.2.85.

[17] Consiglio, M., Hoadley, S., Wing, D., and Baxley, B., "Safety performance of airborne separation: Preliminary baseline testing," *7th AIAA ATIO Conf, 2nd CEIAT Int'l Conf on Innov and Integr in Aero Sciences, 17th LTA Systems Tech Conf; followed by 2nd TEOS Forum*, 2007, p. 7739. doi:https://doi.org/10.2514/6.2007-7739.

[18] Blom, H. A., and Bakker, G., "Safety evaluation of advanced self-separation under very high en route traffic demand," *Journal of Aerospace Information Systems*, Vol. 12, No. 6, 2015, pp. 413–427. doi:https://doi.org/10.2514/1.I010243.

[19] Clari, M. S. V., Ruigrok, R. C., Hoekstra, J. M., and Visser, H. G., "Cost-benefit study of free flight with airborne separation assurance," *Air Traffic Control Quarterly*, Vol. 9, No. 4, 2001, pp. 287–309. doi:https://doi.org/10.2514/atcq.9.4.287.

[20] David J., W., Richard J., A., Jacqueline A., D., Brain M., L., Bryan E., B., and Donald, M., "Airborne Use of Traffic Intent Information in a Distributed Air-Ground Traffic Management Concept: Experiment Design and Preliminary Results," 2001.

[21] Battiste, V., Johnson, W., Kopardekar, P., Lozito, S., Mogford, R., and Palmer, E., "Distributed Air/Ground Traffic Management-Technology and Concept Demonstration Report," *AIAA's Aircraft Technology, Integration, and Operations (ATIO) 2002 Technical Forum*, 2002, p. 5825. doi:https://doi.org/10.2514/6.2002-5825.

[22] Barhydt, R., Kopardekar, P., Battiste, V., Doble, N., Johnson, W., Lee, P., Prevot, T., and Smith, N., "Joint NASA Ames/Langley experimental evaluation of integrated air/ground operations for en route free maneuvering," 2005.

[23] Tomlin, C., Pappas, G. J., and Sastry, S., "Conflict resolution for air traffic management: A study in multiagent hybrid systems," *IEEE Transactions on automatic control*, Vol. 43, No. 4, 1998, pp. 509–521. doi:https://doi.org/10.1109/9.664154.

[24] Kahne, S., and Frolow, I., "Air traffic management: Evolution with technology," *IEEE Control Systems*, Vol. 16, No. 4, 1996, pp. 12–21. doi:https://doi.org/10.1109/37.526911.

[25] Kochenderfer, M. J., Holland, J. E., and Chryssanthacopoulos, J. P., "Next-generation airborne collision avoidance system," Tech. rep., Massachusetts Institute of Technology-Lincoln Laboratory Lexington United States, 2012.

[26] Kuchar, J. K., and Yang, L. C., "A review of conflict detection and resolution modeling methods," *IEEE Transactions on intelligent transportation systems*, Vol. 1, No. 4, 2000, pp. 179–189. doi:https://doi.org/10.1109/6979.898217.

[27] Netjasov, F., and Janic, M., "A review of research on risk and safety modelling in civil aviation," *Journal of Air Transport Management*, Vol. 14, No. 4, 2008, pp. 213–220. doi:https://doi.org/10.1016/j.jairtraman.2008.04.008.

[28] Mitici, M., and Blom, H. A., "Mathematical Models for Air Traffic Conflict and Collision Probability Estimation," *IEEE Transactions on Intelligent Transportation Systems*, , No. 99, 2018, pp. 1–17. doi:https://doi.org/10.1109/TITS.2018.2839344.

[29] SC-186, R. F., *Minimum Operational Performance Standards for 1090 MHz Extended Squitter: Automatic Dependent Surveillance-Broadcast (ADS-B) and Traffic Information Services-Broadcast (TIS-B)*, RTCA, 2006.

[30] Kochenderfer, M. J., and Chryssanthacopoulos, J., "Robust airborne collision avoidance through dynamic programming," *Massachusetts Institute of Technology, Lincoln Laboratory, Project Report ATC-371*, 2011.

[31] Karr, D., Vivona, R., Roscoe, D., Depascale, S., and Wing, D., "Autonomous operations planner: A flexible platform for research in flight-deck support for airborne self-separation," *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2012, p. 5417. doi:https://doi.org/10.2514/6.2012-5417.

[32] Kopardekar, P., Rios, J., Prevot, T., Johnson, M., Jung, J., and Robinson, J. E., "Unmanned aircraft system traffic management (UTM) concept of operations," 2016.

[33] Consiglio, M., Muñoz, C., Hagen, G., Narkawicz, A., and Balachandran, S., "ICAROUS: Integrated configurable algorithms for reliable operations of unmanned systems," *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, IEEE, 2016, pp. 1–5.

[34] Muñoz, C., Narkawicz, A., Hagen, G., Upchurch, J., Dutle, A., Consiglio, M., and Chamberlain, J., "DAIDALUS: detect and avoid alerting logic for unmanned systems," 2015, pp. 5A1–1–5A1–12. doi:https://doi.org/10.1109/DASC.2015.7311421.

[35] Jung, J., D'Souza, S. N., Johnson, M. A., Ishihara, A. K., Modi, H. C., Nikaido, B., and Hasseeb, H., "Applying required navigation performance concept for traffic management of small unmanned aircraft systems," 2016.

[36] Schouwenaars, T., How, J., and Feron, E., "Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, p. 5141. doi:https://doi.org/10.2514/6.2004-5141.

[37] Siegwart, R., Nourbakhsh, I. R., and Scaramuzza, D., *Introduction to autonomous mobile robots*, MIT press, 2011.

[38] Frazzoli, E., Mao, Z.-H., Oh, J.-H., and Feron, E., "Resolution of conflicts involving many aircraft via semidefinite programming," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 1, 2001, pp. 79–86. doi:https://doi.org/10.2514/2.4678.

[39] Raghunathan, A. U., Gopal, V., Subramanian, D., Biegler, L. T., and Samad, T., "Dynamic optimization strategies for three-dimensional conflict resolution of multiple aircraft," *Journal of guidance, control, and dynamics*, Vol. 27, No. 4, 2004, pp. 586–594. doi:https://doi.org/10.2514/1.11168.

[40] Enright, P. J., and Conway, B. A., "Discrete approximations to optimal trajectories using direct transcription and nmiscar programming," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992, pp. 994–1002. doi:https://doi.org/10.2514/3.20934.

[41] Schouwenaars, T., De Moor, B., Feron, E., and How, J., "Mixed integer programming for multi-vehicle path planning," *Control Conference (ECC), 2001 European*, IEEE, 2001, pp. 2603–2608. doi:https://doi.org/10.23919/ECC.2001.7076321.

[42] Richards, A., and How, J. P., "Aircraft trajectory planning with collision avoidance using mixed integer linear programming," *American Control Conference, 2002. Proceedings of the 2002*, Vol. 3, IEEE, 2002, pp. 1936–1941. doi:https://doi.org/10.1109/ACC.2002.1023918.

[43] Pallottino, L., Feron, E. M., and Bicchi, A., "Conflict resolution problems for air traffic management systems solved with mixed integer programming," *IEEE transactions on intelligent transportation systems*, Vol. 3, No. 1, 2002, pp. 3–11. doi:https://doi.org/10.1109/6979.994791.

[44] Vela, A., Solak, S., Singhose, W., and Clarke, J.-P., "A mixed integer program for flight-level assignment and speed control for conflict resolution," *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, IEEE, 2009, pp. 5219–5226. doi:https://doi.org/10.1109/CDC.2009.5400520.

[45] Mellinger, D., Kushleyev, A., and Kumar, V., "Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams," *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, IEEE, 2012, pp. 477–483. doi:https://doi.org/10.1109/ICRA.2012.6225009.

[46] Augugliaro, F., Schoellig, A. P., and D'Andrea, R., "Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach," *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, 2012, pp. 1917–1922. doi:https://doi.org/10.1109/iros.2012.6385823.

[47] Morgan, D., Chung, S.-J., and Hadaegh, F. Y., "Model predictive control of swarms of spacecraft using sequential convex programming," *Journal of Guidance, Control, and Dynamics*, Vol. 37, No. 6, 2014, pp. 1725–1740. doi:https://doi.org/10.2514/1.G000218.

[48] Delahaye, D., Peyronne, C., Mongeau, M., and Puechmorel, S., "Aircraft conflict resolution by genetic algorithm and B-spline approximation," *EIWAC 2010, 2nd ENRI International Workshop on ATM/CNS*, 2010, pp. 71–78.

[49] Cobano, J. A., Conde, R., Alejo, D., and Ollero, A., "Path planning based on genetic algorithms and the monte-carlo method to avoid aerial vehicle collisions under uncertainties," *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE, 2011, pp. 4429–4434. doi:https://doi.org/10.1109/ICRA.2011.5980246.

[50] Pontani, M., and Conway, B. A., "Particle swarm optimization applied to space trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 5, 2010, pp. 1429–1441. doi:https://doi.org/10.2514/1.48475.

[51] Hoffmann, G., Rajnarayan, D. G., Waslander, S. L., Dostal, D., Jang, J. S., and Tomlin, C. J., "The Stanford testbed of autonomous rotorcraft for multi agent control (STARMAC)," *The 23rd Digital Avionics Systems Conference (IEEE Cat. No. 04CH37576)*, Vol. 2, IEEE, 2004, pp. 12–E.

[52] Howlet, J. K., Schulein, G., and Mansur, M. H., "A practical approach to obstacle field route planning for unmanned rotorcraft," 2004.

[53] Canny, J., *The complexity of robot motion planning*, MIT press, 1988.

[54] Kavraki, L., Svestka, P., and Overmars, M. H., *Probabilistic roadmaps for path planning in high-dimensional configuration spaces*, Vol. 1994, Unknown Publisher, 1994.

[55] LaValle, S. M., "Rapidly-exploring random trees: A new tool for path planning," 1998.

[56] Karaman, S., and Frazzoli, E., "Sampling-based algorithms for optimal motion planning," *The international journal of robotics research*, Vol. 30, No. 7, 2011, pp. 846–894.

[57] Čáp, M., Novák, P., Vokrínek, J., and Pěchouček, M., "Multi-agent RRT: sampling-based cooperative pathfinding," *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 1263–1264.

[58] Pallottino, L., Scordio, V. G., Frazzoli, E., and Bicchi, A., "Probabilistic verification of a decentralized policy for conflict resolution in multi-agent systems," *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, IEEE, 2006, pp. 2448–2453. doi:https://doi.org/10.1109/ROBOT.2006.1642069.

[59] Wollkind, S., Valasek, J., and Ioerger, T., "Automated conflict resolution for air traffic management using cooperative multiagent negotiation," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, p. 4992. doi:https://doi.org/10.1.1.76.2547.

[60] Purwin, O., D'Andrea, R., and Lee, J.-W., "Theory and implementation of path planning by negotiation for decentralized agents," *Robotics and Autonomous Systems*, Vol. 56, No. 5, 2008, pp. 422–436. doi:https://doi.org/10.1016/j.robot.2007.09.020.

[61] Desaraju, V. R., and How, J. P., "Decentralized path planning for multi-agent teams in complex environments using rapidly-exploring random trees," *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE, 2011, pp. 4956–4961. doi:https://doi.org/10.1109/ICRA.2011.5980392.

[62] Inalhan, G., Stipanovic, D. M., and Tomlin, C. J., "Decentralized optimization, with application to multiple aircraft coordination," *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, Vol. 1, IEEE, 2002, pp. 1147–1155. doi:https://doi.org/10.1109/CDC.2002.1184667.

[63] Richards, A., and How, J., "Decentralized model predictive control of cooperating UAVs," *43rd IEEE Conference on Decision and Control*, Vol. 4, Citeseer, 2004, pp. 4286–4291. doi:https://doi.org/10.1109/CDC.2004.1429425.

[64] Shim, D. H., and Sastry, S., "An evasive maneuvering algorithm for UAVs in see-and-avoid situations," *American Control Conference, 2007. ACC'07*, IEEE, 2007, pp. 3886–3891. doi:https://doi.org/10.1109/ACC.2007.4283147.

[65] Shim, D. H., Kim, H. J., and Sastry, S., "Decentralized nonlinear model predictive control of multiple flying robots," *Decision and control, 2003. Proceedings. 42nd IEEE conference on*, Vol. 4, IEEE, 2003, pp. 3621–3626. doi:https://doi.org/10.1109/CDC.2003.1271710.

[66] Khatib, O., and Mampey, L., "Fonction decision-commande d'un robot manipulateur," *Rep*, Vol. 2, No. 7, 1978, p. 156.

[67] Koditschek, D. E., and Rimon, E., "Robot navigation functions on manifolds with boundary," *Advances in applied mathematics*, Vol. 11, No. 4, 1990, pp. 412–442.

[68] Rimon, E., and Koditschek, D. E., "Exact robot navigation using cost functions: the case of distinct spherical boundaries in E/sup n," *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*, IEEE, 1988, pp. 1791–1796.

[69] Connolly, C. I., Burns, J. B., and Weiss, R., "Path planning using Laplace's equation," *Robotics and Automation, 1990. Proceedings., 1990 IEEE International Conference on*, IEEE, 1990, pp. 2102–2106. doi:https://doi.org/10.1109/ROBOT.1990.126315.

[70] Kahn, G., Zhang, T., Levine, S., and Abbeel, P., "Plato: Policy learning using adaptive trajectory optimization," *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, 2017, pp. 3342–3349. doi:https://doi.org/10.1109/ICRA.2017.7989379.

[71] Ong, H. Y., and Kochenderfer, M. J., "Markov Decision Process-Based Distributed Conflict Resolution for Drone Air Traffic Management," *Journal of Guidance, Control, and Dynamics*, 2016, pp. 69–80. doi:https://doi.org/10.2514/1.G001822.

[72] Chen, Y. F., Liu, M., Everett, M., and How, J. P., "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, 2017, pp. 285–292. doi:https://doi.org/10.1109/ICRA.2017.7989037.

[73] Li, S., Egorov, M., and Kochenderfer, M., "Optimizing Collision Avoidance in Dense Airspace using Deep Reinforcement Learning," *arXiv preprint arXiv:1912.10146*, 2019.

[74] Brittain, M., and Wei, P., "Autonomous Separation Assurance in An High-Density En Route Sector: A Deep Multi-Agent Reinforcement Learning Approach," *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 3256–3262. doi:https://doi.org/10.1109/ITSC.2019.8917217.

[75] Yang, X., and Wei, P., "Autonomous On-Demand Free Flight Operations in Urban Air Mobility using Monte Carlo Tree Search," *8th International Conference on Research in Air Transportation (ICRAT)*, ICRAT, 2018.

[76] Wolf, T. B., and Kochenderfer, M. J., "Aircraft collision avoidance using Monte Carlo real-time belief space search," *Journal of Intelligent & Robotic Systems*, Vol. 64, No. 2, 2011, pp. 277–298. doi:https://doi.org/10.1007/s10846-010-9532-6.

[77] Han, S.-C., Bang, H., and Yoo, C.-S., "Proportional navigation-based collision avoidance for UAVs," *International Journal of Control, Automation and Systems*, Vol. 7, No. 4, 2009, pp. 553–565. doi:https://doi.org/10.1007/s12555-009-0407-1.

[78] Park, J.-W., Oh, H.-D., and Tahk, M.-J., "UAV collision avoidance based on geometric approach," *SICE Annual Conference, 2008*, IEEE, 2008, pp. 2122–2126. doi:https://doi.org/10.1109/SICE.2008.4655013.

[79] Krozel, J., Peters, M., and Bilimoria, K., "A decentralized control strategy for distributed air/ground traffic separation," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2000, p. 4062. doi:https://doi.org/10.2514/6.2000-4062.

[80] Van Den Berg, J., Guy, S. J., Lin, M., and Manocha, D., "Reciprocal n-body collision avoidance," *Robotics research*, Springer, 2011, pp. 3–19. doi:https://doi.org/10.1007/978-3-642-19457-3_1.

[81] Yang, X., Deng, L., Liu, J., Wei, P., and Li, H., "Multi-Agent Autonomous Operations in Urban Air Mobility with Communication Constraints," *AIAA Scitech 2020 Forum*, 2020, p. 1839. doi:https://doi.org/10.2514/6.2020-1839.

[82] Temizer, S., Kochenderfer, M., Kaelbling, L., Lozano-Pérez, T., and Kuchar, J., "Collision avoidance for unmanned aircraft using Markov decision processes," *AIAA guidance, navigation, and control conference*, 2010, p. 8040.

[83] Bertram, J., and Wei, P., "Distributed Computational Guidance for High-Density Urban Air Mobility with Cooperative and Non-Cooperative Collision Avoidance," *AIAA Scitech 2020 Forum*, 2020, p. 1371. doi:https://doi.org/10.2514/6.2020-1371.

[84] Federal Aviation Administration, "Control Sector," http://www.faraim.org/aim/aim-4-03-14-618.html, 2019. Accessed: 2020-02-15.

[85] NATS, "Introduction to Airspace," https://www.nats.aero/ae-home/introduction-to-airspace/, 2019. Accessed: 2020-02-15.

[86] SKYbrary, "Loss of Separation at Sector Boundaries," https://www.skybrary.aero/index.php/Loss_of_Separation_at_Sector_Boundaries#, 2019. Accessed: 2020-02-15.

[87] Pradeep, P., and Wei, P., "Energy Optimal Speed Profile for Arrival of Tandem Tilt-Wing eVTOL Aircraft with RTA Constraint," *IEEE CSAA Guidance, Navigation and Control Conference*, 2018. doi:https://doi.org/10.2514/6.2018-2008.

[88] Airbus, "Vahana," https://www.airbus.com/innovation/urban-air-mobility/vehicle-demonstrators/vahana.html, 2018. Accessed: 2020-02-15.

[89] Pradeep, P., and Wei, P., "Heuristic Approach for Arrival Sequencing and Scheduling for eVTOL Aircraft in On-Demand Urban Air Mobility," *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*, IEEE, 2018, pp. 1–7.

[90] Kleinbekman, I. C., Mitici, M. A., and Wei, P., "eVTOL Arrival Sequencing and Scheduling for On-Demand Urban Air Mobility," *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*, IEEE, 2018, pp. 1–7.

[91] Bertram, J., and Wei, P., "An Efficient Algorithm for Self-Organized Terminal Arrival in Urban Air Mobility," *AIAA Scitech 2020 Forum*, 2020, p. 0660. doi:https://doi.org/10.2514/6.2020-0660.

[92] Stoschek, A., "Exploring Sense-and-Avoid Systems for Autonomous Vehicles," https://acubed.airbus.com/blog/vahana/exploring-sense-and-avoid-systems-for-autonomous-vehicles/, Dec 2017. Accessed: 2020-02-15.

[93] Bellman, R., "A Markovian Decision Process," *Indiana Univ. Math. J.*, Vol. 6, 1957, pp. 679–684.

[94] Howard, R. A., "Dynamic programming and Markov processes," 1964. doi:https://doi.org/10.1126/science.132.3428.667.

[95] White, D. J., "A survey of applications of Markov decision processes," *Journal of the operational research society*, Vol. 44, No. 11, 1993, pp. 1073–1096. doi:https://doi.org/10.1057/jors.1993.181.

[96] Feinberg, E. A., and Shwartz, A., *Handbook of Markov decision processes: methods and applications*, Vol. 40, Springer Science & Business Media, 2012. doi:https://doi.org/10.1007/978-1-4615-0805-2.

[97] Koenig, S., and Simmons, R., "Xavier: A robot navigation architecture based on partially observable markov decision process models," *Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems*, 1998, pp. 91–122.

[98] Thrun, S., "Probabilistic robotics," *Communications of the ACM*, Vol. 45, No. 3, 2002, pp. 52–57. doi:https://doi.org/10.1017/S0269888906210993.

[99] Mariton, M., *Jump linear systems in automatic control*, M. Dekker New York, 1990.

[100] Puterman, M. L., *Markov decision processes: discrete stochastic dynamic programming*, John Wiley & Sons, 2014.

[101] Coulom, R., "Efficient selectivity and backup operators in Monte-Carlo tree search," *International conference on computers and games*, Springer, 2006, pp. 72–83. doi:https://doi.org/10.1007/978-3-540-75538-8_7.

[102] Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S., "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, Vol. 4, No. 1, 2012, pp. 1–43. doi:https://doi.org/10.1109/TCIAIG.2012.2186810.

[103] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al., "Mastering the game of Go with deep neural networks and tree search," *nature*, Vol. 529, No. 7587, 2016, p. 484. doi:https://doi.org/10.1038/nature16961.

[104] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al., "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," *arXiv preprint arXiv:1712.01815*, 2017.

[105] Champandard, A. J., "Monte-Carlo tree search in TOTAL WAR: ROME II's campaign AI," *AIGameDev.com: http://aigamedev.com/open/coverage/mcts-rome-ii*, 2014.

[106] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al., "Mastering the game of Go without human knowledge," *Nature*, Vol. 550, No. 7676, 2017, p. 354. doi: https://doi.org/10.1038/nature24270.

[107] Boutilier, C., "Planning, learning and coordination in multiagent decision processes," *Proceedings of the 6th conference on Theoretical aspects of rationality and knowledge*, Morgan Kaufmann Publishers Inc., 1996, pp. 195–210.

[108] Boutilier, C., Dean, T., and Hanks, S., "Decision-theoretic planning: Structural assumptions and computational leverage," *Journal of Artificial Intelligence Research*, Vol. 11, 1999, pp. 1–94.

[109] Boutilier, C., "Sequential optimality and coordination in multiagent systems," *IJCAI*, Vol. 99, 1999, pp. 478–485.

[110] Sigaud, O., and Buffet, O., *Markov decision processes in artificial intelligence*, John Wiley & Sons, 2013.

[111] Busoniu, L., Babuška, R., and De Schutter, B., "Multi-agent reinforcement learning: An overview," *Innovations in multi-agent systems and applications-1*, Vol. 310, 2010, pp. 183–221.

[112] Nash, J. F., et al., "Equilibrium points in n-person games," *Proceedings of the national academy of sciences*, Vol. 36, No. 1, 1950, pp. 48–49.

[113] Kochenderfer, M. J., *Decision making under uncertainty: theory and application*, MIT press, 2015.

[114] Daskalakis, C., Goldberg, P. W., and Papadimitriou, C. H., "The complexity of computing a Nash equilibrium," *SIAM Journal on Computing*, Vol. 39, No. 1, 2009, pp. 195–259.

[115] Camerer, C. F., *Behavioral game theory: Experiments in strategic interaction*, Princeton University Press, 2011.

[116] Stahl II, D. O., and Wilson, P. W., "Experimental evidence on players' models of other players," *Journal of economic behavior & organization*, Vol. 25, No. 3, 1994, pp. 309–327.

[117] Stahl, D. O., and Wilson, P. W., "On players' models of other players: Theory and experimental evidence," *Games and Economic Behavior*, Vol. 10, No. 1, 1995, pp. 218–254.

[118] Julian, K. D., Lopez, J., Brush, J. S., Owen, M. P., and Kochenderfer, M. J., "Policy compression for aircraft collision avoidance systems," *Digital Avionics Systems Conference (DASC), 2016 IEEE/AIAA 35th*, IEEE, 2016, pp. 1–10. doi: https://doi.org/10.1109/DASC.2016.7778091.

[119] Julian, K. D., Kochenderfer, M. J., and Owen, M. P., "Deep neural network compression for aircraft collision avoidance systems," *Journal of Guidance, Control, and Dynamics*, Vol. 42, No. 3, 2018, pp. 598–608.

[120] Tesauro, G., and Galperin, G. R., "On-line policy improvement using Monte-Carlo search," *Advances in Neural Information Processing Systems*, 1997, pp. 1068–1074.

[121] Chaslot, G. M. J., Winands, M. H., HERIK, H. J. V. D., Uiterwijk, J. W., and Bouzy, B., "Progressive strategies for Monte-Carlo tree search," *New Mathematics and Natural Computation*, Vol. 4, No. 03, 2008, pp. 343–357.

[122] Coulom, R., "Computing "elo ratings" of move patterns in the game of go," *Icga Journal*, Vol. 30, No. 4, 2007, pp. 198–208.

[123] Wang, Y., Audibert, J.-Y., and Munos, R., "Algorithms for infinitely many-armed bandits," *Advances in Neural Information Processing Systems*, 2009, pp. 1729–1736.

[124] Couëtoux, A., Hoock, J.-B., Sokolovska, N., Teytaud, O., and Bonnard, N., "Continuous upper confidence trees," *International Conference on Learning and Intelligent Optimization*, Springer, 2011, pp. 433–445.

[125] Julian, K. D., and Kochenderfer, M. J., "Neural network guidance for UAVs," *AIAA Guidance, Navigation, and Control Conference*, 2017, p. 1743.

[126] Bosson, C., and Lauderdale, T. A., "Simulation Evaluations of an Autonomous Urban Air Mobility Network Management and Separation Service," *2018 Aviation Technology, Integration, and Operations Conference*, 2018, p. 3365. doi:https://doi.org/10.2514/6.2018-3365.

[127] Cook, S. P., and Brooks, D., "A Quantitative Metric to Enable Unmanned Aircraft Systems to Remain Well Clear," *Air Traffic Control Quarterly*, Vol. 23, No. 2-3, 2015, pp. 137–156. doi:https://doi.org/10.2514/atcq.23.2-3.137.

[128] Kocsis, L., Szepesvári, C., and Willemson, J., "Improved monte-carlo search," *Univ. Tartu, Estonia, Tech. Rep*, Vol. 1, 2006.

[129] Kohlman, L. W., and Patterson, M. D., "System-level urban air mobility transportation modeling and determination of energy-related constraints," *2018 Aviation Technology, Integration, and Operations Conference*, 2018, p. 3677.

[130] Patterson, M. D., Antcliff, K. R., and Kohlman, L. W., "A Proposed Approach to Studying Urban Air Mobility Missions Including an Initial Exploration of Mission Requirements," 2018.

[131] Google, "Google Maps," https://www.google.com/maps, 2020. Accessed: 2020-02-15.

[132] Syed, N., Rye, M., Ade, M., Trani, A., Hinze, N., Swingle, H., Smith, J. C., Dollyhigh, S., and Marien, T., "Preliminary Considerations for ODM Air Traffic Management based on Analysis of Commuter Passenger Demand and Travel Patterns for the Silicon Valley Region of California," *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 3082. doi:https://doi.org/10.2514/6.2017-3082.

[133] Vascik, P. D., Cho, J., Bulusu, V., and Polishchuk, V., "A Geometric Approach Towards Airspace Assessment for Emerging Operations," 2019.

[134] Federal Aviation Administration, "Near Midair Collision Reporting," http://www.faraim.org/aim/aim-4-03-14-530.html, 2017. Accessed: 2020-02-15.