Multi-Agent Autonomous On-Demand Free Flight Operations in Urban Air Mobility

Conference Paper · June 2019

DOI: 10.2514/6.2019-3520

3 authors, including:

Xuxi Yang
Iowa State University
14 PUBLICATIONS **65** CITATIONS
SEE PROFILE

Peng Wei
George Washington University
89 PUBLICATIONS **553** CITATIONS
SEE PROFILE

Multi-Agent Autonomous On-Demand Free Flight Operations in Urban Air Mobility

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

In urban air mobility (UAM), flying with electrical vertical takeoff and landing (eVTOL) aircraft will bring fundamental changes to city infrastructures and daily commutes. In order to enable safe and efficient autonomous on-demand free flight operations for the eVTOL aircraft in UAM, a centralized computational guidance algorithm is proposed and analyzed for multi cooperative aircraft. The approach proposed in this paper is to formulate this problem as a Markov Decision Process (MDP) and solve it using an online algorithm Monte Carlo Tree Search (MCTS). A coordination mechanism is designed to manage multiple cooperative aircraft. By generating real-time actions for all the cooperative aircraft to follow, the algorithm can guide all the aircraft to their respective destinations while avoiding potential conflicts between them. For the sake of illustration, a free flight airspace simulator is created to test the performance of this algorithm. Results show that this algorithm can help all the aircraft reach their trip destinations while only having 0.2% conflicts during the flights.

I. Introduction

A. Motivation

URBAN air mobility, an emerging transportation concept which becomes popular in recent years, encompasses the movement of people and cargo around metropolitan areas in vehicles that fly within and/or at low altitudes over urban areas. NASA, Uber, and Airbus have been exploring this exciting concept of Urban Air Mobility (UAM) [1–6], and companies includes Airbus, Bell, Embraer, Joby, Zee Aero, Pipistrel, Volocopter, and Aurora Flight Sciences have been working to build and test electric vertical takeoff and landing (eVTOL) aircraft. In UAM, the eVTOL aircraft may be either human piloted or autonomous for passenger transport in personal commute or on-demand air taxi [7–10]. The UAM operations are expected to fundamentally change cities and people's lives to reduce commute time and stress. The vehicle technology and airspace operation concepts are critical for UAM realization. Through close collaborations with our colleagues from Airbus, in this paper we investigate how to combine the power of onboard aircraft intelligence (vehicle technology) and the advantage of the free flight idea (airspace operation concept) to enable safe and efficient flight operations in on-demand urban air transportation.

The concept of "Free Flight" was proposed primarily for future air transportation applications because it has the potential to cope with the ongoing congestion of the current ATC system. It was shown in previous work [11, 12] that free flight with airborne separation is able to handle a higher traffic density comparing with structured airspace. Besides, free flight can also bring fuel and time efficiency [13]. In a free flight framework, it is implied that aircraft will be responsible for their own separation assurance and conflict resolution. The loss of an airway structure may make the process of detecting and resolving conflicts between aircraft more complex. However, previous study [14] shows that free flight is potentially feasible because of enabling technologies such as Global Positioning Systems (GPS), data link communications like Automatic Dependence Surveillance-Broadcast (ADSB) [15], Traffic Alert and Collision Avoidance Systems (TCAS) [16], and powerful on-board computation. Also, automated conflict detection and resolution tools [17] will be required to aid pilots and/or ground controllers in ensuring traffic separation and conflict resolution.

In this paper, a computational guidance algorithm with collision avoidance capability is proposed to help guide the aircraft to their respective destinations, where the aircraft dynamics is modeled based on the tandem tilt-wing eVTOL (Airbus Vahana) from Airbus A^3 [18], as shown in Fig. 1. The proposed algorithm in this paper uses Markov Decision Process and Monte Carlo Tree Search, where the input of this algorithm is the current position and velocity for all the aircraft, and the position of their respective target vertiport. Based on this information and with a designed coordination

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

mechanism, the aircraft will perform online sequential decision making to select actions in real time with onboard avionics computation. The series of actions will guide all the aircraft to reach its goal and avoid potential conflicts. The proposed algorithm provides a potential solution framework to enable autonomous on-demand free flight operations in urban air mobility.

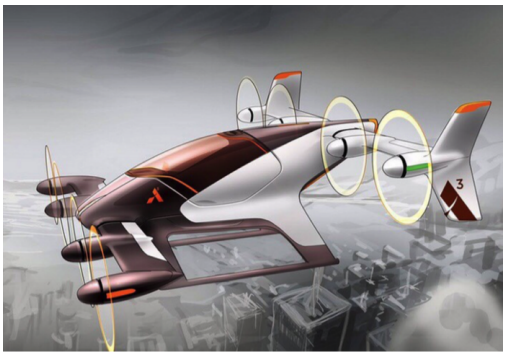

Fig. 1 Airbus Vahana with tandem tilt-wing configuration during the cruise phase [19].

B. Related Work

There have been many important contributions to the topic of guidance algorithm with collision avoidance capability for small unmanned aerial aircraft. And these approaches can be roughly categorized based on the following criteria:

- Centralized/Decentralized [20]: whether the problem is solved by a central supervising controller (centralized) or by each aircraft individually (decentralized).
- Planning/Reacting [21]: The planned approach generates feasible paths ahead of time; whereas the reactive approach typically uses an online collision avoidance system to respond to dangerous situations.
- Cooperative/Non-cooperative: whether there exists online communication between aircraft or between aircraft and the central controller.

In the following, we will briefly discuss the related work based on the first criteria: centralized method and decentralized method. And the remaining criteria will be discussed under the first criteria.

In centralized methods, the conflicts between aircraft are resolved by a central supervising controller. Under such scenario, the state of each aircraft, the obstacle information, the trajectory constraint as well as the terminal condition is known to the central controller (thus centralized methods are always cooperative), and the central controller in return designs the individual whole trajectory for all aircraft before the flight, typically by formulating it to an optimal control problem. These methods can be based on semidefinite programming [22], nonlinear programming [23, 24], mixed integer linear programming [25–28], mixed integer quadratic programming [29], sequential convex programming [30, 31], second-order cone programming [32], evolutionary techniques [33, 34], and particle swarm optimization [35]. Besides formulating this problem using optimal control framework, roadmap methods such as visibility graph [36] and Voronoi diagrams [37] can also handle the path planning problem for aircraft. However, calculating the exact solutions will become impractical when the state space becomes large or high-dimensional. To address this issue, sample-based planning algorithms are proposed, such as probabilistic roadmaps [38], RRT [39], and RRT* [40]. These centralized methods often pursue the global optimality of the solution. However, as the number of aircraft grows, 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).

On the other hand, decentralized methods scale better with respect to the number of agents and are more robust since they do not possess a single point of failure [41]. 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) [42]. Algorithms in [43, 44] 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 [20], every agent is allotted a time slot in which to compute a dynamically feasible and guaranteed collision-free path using MILP. In [45], the author recast the global optimization problem as several local problems, which are then iteratively solved by the agents in a decentralized way. In Decentralized Model Predictive Control approach [46], the aircraft solve their own sub-problem

one after the other and send the action to other subsystems through communication.

There are also some works focus on scenarios where communication cannot be reliably established (non-cooperative). In this case, the algorithms usually output one action at each time step and by following the issued actions, the aircraft can reach the goal position and avoid conflicts with other aircraft. Many works fall in this category: Model Predictive Control [47, 48] can be used to solve collision avoidance problem but the computation load is relatively high. Potential field method [49, 50] is computationally fast, but in general they provide no guarantees of collision avoidance. With the help of machine learning and reinforcement learning [51–54], collision avoidance algorithm can have a promising performance, but usually needs a lot of time to train. Monte Carlo Tree Search algorithm [55] does not need time to train before the flight and it can finish in any predefined computation time, but the aircraft can only adopt several discretized actions at each time step. Geometric approach [56–59] can be also applied for collision avoidance problem and the computation time only grows linearly as the increase of number of aircraft. DAIDALUS (Detect and Avoid Alerting Logic for Unmanned Systems) [60] is a geometry-based approach developed by NASA. The core logic of DAIDALUS consists of: (1) definition of self-separation threshold (SST) and well-clear violation volume (WCV), (2) algorithms for determining if there exists potential conflict between aircraft pairs within a given lookahead time, and (3) a determine-processing functionality provides maneuver guidance and alerting logic. The drawback of these geometric approaches is that it cannot look ahead for more than one step (it only pays attention to the current action and does not take account of the effect of subsequent actions) and the outcome can be local optimal in the view of the global trajectory.

Our proposed algorithm is an extension of previous work [55], where the authors formulate the computational guidance problem with collision avoidance function as a Markov Decision Process (MDP) problem and solve this MDP using Monte Carlo Tree Search (MCTS) algorithm. Numerical experiments result under high-density airspace scenarios show that when the algorithm controls the aircraft by changing the heading angle, the aircraft can get to the destination and have an average of 0.5 conflicts and 0.05 near mid-air collision along the way from the origin to goal position. A short video demo for this previous work was released [61].

Comparing with previous work [55] where we can only control one aircraft to avoid conflicts with other intruder aircraft, the algorithm proposed in this paper can help multiple aircraft currently flying in the airspace taking actions in a cooperative way by letting them communicate with each other.

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

II. Background

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

A. Markov Decision Process (MDP)

Since the 1950s, MDPs [62] have been well studied and applied to a wide area of disciplines, including robotics, automatic control, economics, and manufacturing. 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' according to the transition probability, and given 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 how much reward the agent will collect after a transition s_t, a_t, s_{t+1}. 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)

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

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

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

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 [63, 64]. 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 [65–67], including the current state-of-art computer program AlphaZero in the Game of Go [68].

The basic MCTS process is conceptually very simple, as shown in Fig. 2 (from [64]). A tree is built in an incremental and asymmetric manner. 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.

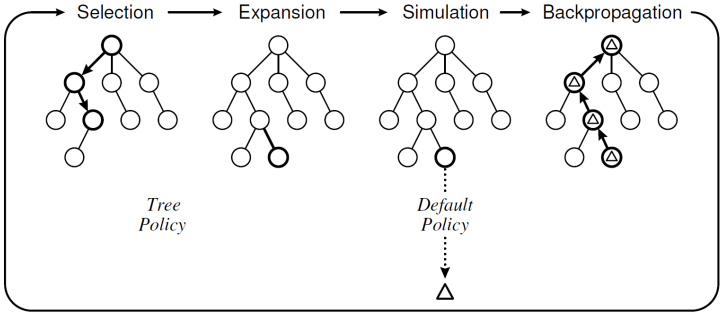

Fig. 2 One iteration of general MCTS approach [64].

C. Multi-Agent Systems

Comparing with previous work [55] where the algorithm can only control one single aircraft, in this paper we will investigate how MCTS algorithm can be used to control multiple cooperative agents. A multi-agent system is a group of autonomous, interacting entities sharing a common environment, which they perceive with sensors and upon which they make decisions and act with actuators [69–74]. Although the multi-agent systems can model many agents, there are also many challenges in multi-agent systems [73]:

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 multi-agent systems because each agent is facing with 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 can't tell what's the optimal action. And even if we get the optimal action for both aircraft (e.g. both turn right), the optimal action of one aircraft will be affected if other aircraft changed their actions.

Most of the literature studied multi-agent games in stochastic environments with a focus on equilibrium or long-term stable behaviors. In this paper, we use a sequential decision making technique to solve the above two issues in a fast,

dynamic and uncertain environment where the real-time online decisions are needed.

III. Problem Formulation

A. Problem Statement

The goal of this paper is to control all the aircraft flying in the airspace through a series of actions so that all the aircraft can arrive at the destination while avoiding potential conflict among them during the flight. Guiding the aircraft through a series of actions is a sequential decision making problem which can be formulated as a MDP problem. In this process, the action is decided directly from the state, which incorporates all the information (the position and velocity of all the aircraft and their respective destinations or goals) 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 height and this problem is solved in 2 dimensions. This assumption is reasonable because UTM limits its focus to different altitude levels with one proposal restricting altitudes to between 200 and 500 feet [75].

The objectives for this specific MDP problem are two-fold: the first is to guide the aircraft to the goal state in a short time, and the second is to avoid any conflicts among all the aircraft. Therefore, the reward function should be able to capture both two objectives.

Based on the above description, this problem will be mathematically formulated as a MDP problem in the next subsection, where the MDP is formulated in the perspective of single aircraft. The multi-agent cooperation strategy will be discussed in Section IV.

B. MDP Formulation

The MDP formulation in the view of single aircraft is similar to previous work [55] and here we will briefly introduce it.

1. State Space

For one single aircraft, the state includes the necessary information for it to select the optimal action: the position, velocity, speed, heading angle, and goal position for this aircraft, as well as the position, velocity of all the other aircraft. More specifically, the state for one aircraft will be $(x, y, v_x, v_y, v, \psi, g_x, g_y)$, where $(x, y), (v_x, v_y)$ is the position and velocity, v is the speed, ψ is the heading angle, and (g_x, g_y) is the goal position for this aircraft. The state for other aircraft will be n by 4 matrix which only consists their position and velocity information. 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 includes using a grid-based discretization of the state space $\mathcal{S}$ and the action space $\mathcal{A}$ [53, 76] or using policy compression techniques [77]. The advantage of MCTS algorithm is that it does not need to discretize the state space. And for each state, the MCTS algorithm will generate action for all 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 correspond to left turn and negative to right turn. At each time step, each aircraft will choose one action $a_\psi \in \mathcal{A}$ based on the current state and the aircraft will maintain the action during this time step.

3. Dynamical Model

After the aircraft chooses an action, Dubin's kinematic model will be used to compute state transition for the aircraft:

$$\begin{aligned} \dot{x} &= v\cos\psi \\ \dot{y} &= v\sin\psi \\ \dot{\psi} &= a_\psi \end{aligned}$$

where v is the speed of the aircraft, ψ is the heading angle, and a_ψ is the changing rate of heading angle.

After an aircraft execute an advisory, a normally distributed noise with standard deviation of 2° will be added to the heading angle, and a normally distributed noise with standard deviation of $5m/s$ will be added to the speed. The noises here aim to account for the uncertainties in the environment and aircraft dynamics.

4. Terminal State

For the consideration of safety, the conflict is defined to be when the distance of two aircraft is less than a minimum separation distance $r^{min} = 0.3$ nautical miles [78]. This separation standard was chosen using the definition of well clear for Unmanned Aircraft Systems (UAS) according to Cook and Brooks [79]. 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 0.3nmi horizontally. These are tighter than UAS standards because it is assumed that enhanced equipage capabilities will be installed on board UAM aircraft [78].

Based on the above separation requirements, the terminal state of this MDP includes three different types of states:

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

5. Reward Function

The goal in this paper is to make an aircraft reach its destination and avoid potential conflict. These two objectives can be captured in the reward function defined as follows:

$$R(s) = \begin{cases} 1, & \text{if } s \text{ is goal state,} \\ 0, & \text{otherwise.} \end{cases}$$

With this reward setting, reaching a conflict state before the goal state will terminate the whole process with a reward of 0. Reaching a goal state will terminate this process with a reward of 1. So when maximizing the reward, the agent will try to reach the goal state and avoid conflict states. Therefore we do not need to introduce a penalty conflict 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 this problem. The details of UCT algorithm implementation can be found at [55]. In MCTS, we build a search tree for each aircraft and treat other aircraft as intruder aircraft with the same dynamical model. In this way we can avoid the exponential growth of state space and action space. For the cooperation strategy between aircraft, we will describe in the following subsection how this proposed algorithm can coordinate multiple aircraft.

Cooperative Sequential Decision Making

The cooperation among multiple agents is known to be a challenging problem. As shown in Fig. 3, if one aircraft doesn't know the action taken by the other aircraft, then each action could possibly lead to a conflict between them. On the other hand, if one aircraft decides to turn left and communicate this information to the other aircraft, then the next aircraft, through simulation and building the search tree, will be able to find that going straight or turning left can avoid this potential conflict.

The cooperation strategy used in this paper will be based on this idea, which is to let the centralized controller make decisions for all the aircraft one after another. When building a search tree for one aircraft, the centralized controller will use the action decision it made for previous aircraft.

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

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

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

Second, starting from aircraft 1, the centralized controller will simulate and build a search tree, assuming all the other aircraft will take action according to the joint action $\mathbf{a}$ and follow the dynamical model described in Section III. B.

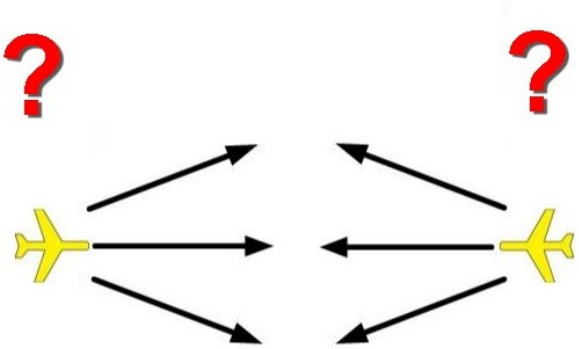

Fig. 3 The action selected by one aircraft also depend on the action selected by the other aircraft.

3. The searching process is similar to the process described in [55] and the only difference is all aircraft can turning according to the joint action.

Next, assume the result of search for the first aircraft is a_1^*, then we update the joint action as follows:

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

After updating the joint action, the search for the second aircraft begins. This process will iterate over all the aircraft until all the $a_1, a_2, \cdots, a_n$ are updated. Then all the aircraft will take action according to the updated joint action for 5 time steps, after which the algorithm will run again to generate new joint action for all the aircraft.

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. The airspace has 48km length and 48km width. 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.

To see the performance of this algorithm in real world applications, we will simplify the UAM network by following the generic city model presented in [80, 81]. In this generic city model, seven vertiports are distributed in a "six around one" hexagonal pattern. As shown in Fig. 4, Vertiport 1 is located in the center of the hexagon and is 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 5pm [82].

Given the above vertiports network, the demand model will generate 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 3 minutes. Each aircraft 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 conflicts and NMACs (short for near mid-air collision), the total number of aircraft generated, the number of aircraft which reached goals, and the average computation time to make each decision will be recorded and compared. The near mid-air collision (NMAC) standard is defined to be 500 feet by the Aeronautical Information Manual (7-6-3) [83].

B. Results

It should be noted here there are two parameters that can impact the performance of MCTS algorithm: the number of simulations and the search depth. In previous work [55], it is found that setting number of simulations to 100 and search depth to 3 in this problem can have decent performance. Thus in this paper we adopt the same parameter setting.

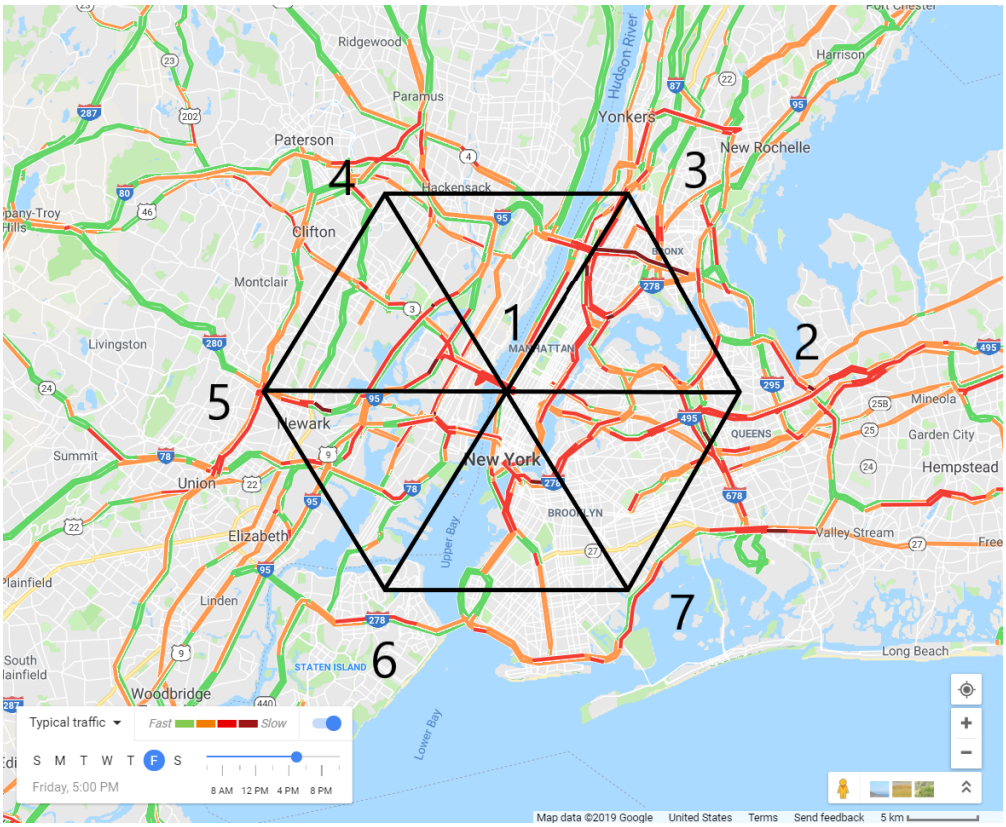

Fig. 4 Network of seven vertiports overlaid on New York city with segment length 16km.

Table 1 Performance of MCTS algorithm.

	mean	variance	min	max
Number of Conflicts	18.6	9.04	15	22
Number of NMACs	0.4	0.24	0	1
Number of Aircraft Reached Goal	9999.2	0.96	9998	10000

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 1500m, we set the number of simulations to 10 and search depth to 2.

The above experiment is conducted over 5 random seeds and in each experiment, there are 10,000 aircraft generated in total. The code implementation of this algorithm is available on GitHub * and a short video demo can be found on YouTube †. The result of the numerical experiment is shown below.

Table 1 shows the number of conflicts/NMACs and the number of aircraft reached the goal, from which we can see for all the 10,000 aircraft generated, the conflict probability is around 0.2% and the NMAC probability is 0.004%. This means the MCTS algorithm is very efficient in this guidance and collision avoidance problem.

Fig. 5 shows the computation time needed to decide a joint action for all the aircraft, with the shaded area being the standard deviation. This shows the computation is growing with the increase of the number of aircraft and if the number of aircraft is less than 20, the search can finish in a reasonable time (600ms).

*https://github.com/xuxiyang1993/MCTS-for-Multi-Agent-Computational-Guidance

†https://www.youtube.com/watch?v=Q6AkCCt-9nI&t=

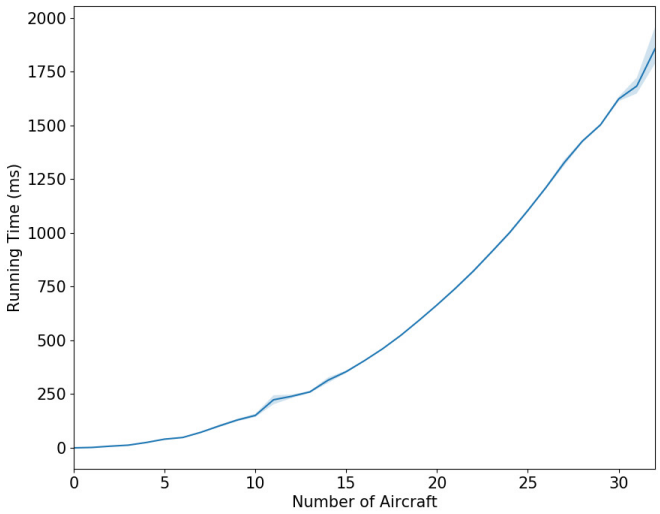

Fig. 5 The computation time with different number of aircraft.

VI. Conclusion

A computational guidance algorithm with collision avoidance capability for autonomous on-demand free flight operations with multiple cooperative aircraft in urban air mobility is proposed in this paper. The problem is formulated as a Markov Decision Process (MDP) and then solved by an online algorithm Monte Carlo Tree Search (MCTS). A coordination mechanism was designed to manage multiple cooperative aircraft. Numerical experiments show that this proposed algorithm has promising performance to help an aircraft reach its destination and avoid potential conflicts with other aircraft. This proposed algorithm provides a potential solution framework to enable autonomous on-demand free flight operations in urban air mobility.

The contribution of this research is integrating the power of onboard aircraft intelligence (vehicle autonomy technology) and the advantage of the free flight concept for airspace operations to enable safe and efficient flight operations in on-demand urban air transportation for multi cooperative aircraft.

Acknowledgments

This material is based upon work supported by the National Science Foundation under award #1565979 and Airbus A^3. This work has benefited from discussions with Karthik Balakrishnan and Richard Golding at Airbus A^3. The authors thank the A^3 Altiscope team for their guidance and support throughout this work.

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: 2018-01-19.

[2] Gipson, L., "Taking Air Travel to the Streets, or Just Above Them," `https://www.nasa.gov/aero/taking-air-travel-to-the-streets-or-just-above-them`, 2018. Accessed: 2018-08-13.

[3] "Uber Elevate | The Future of Urban Air Transport," `https://www.uber.com/info/elevate/`, 2017. Accessed: 2018-08-13.

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

[5] "Urban Air Mobility," `http://publicaffairs.airbus.com/default/public-affairs/int/en/our-topics/Urban-Air-Mobility.html`, 2018. Accessed: 2018-08-13.

[6] "Future of Urban Mobility," `http://www.airbus.com/newsroom/news/en/2016/12/My-Kind-Of-Flyover.html`, 2017. Accessed: 2018-08-13.

[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.

[8] Antcliff, K. R., Moore, M. D., and Goodrich, K. H., "Silicon Valley as an Early Adopter for On-Demand Civil VTOL Operations," *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3466.

[9] Moore, M. D., and Goodrich, K. H., "High speed mobility through on-demand aviation," *2013 Aviation Technology, Integration, and Operations Conference*, 2013, p. 4373.

[10] Vascik, P. D., "Systems-level analysis of On Demand Mobility for aviation," Ph.D. thesis, Massachusetts Institute of Technology, 2017.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] Kahne, S., and Frolow, I., "Air traffic management: Evolution with technology," *IEEE Control Systems*, Vol. 16, No. 4, 1996, pp. 12–21.

[16] Harman, W. H., "TCAS- A system for preventing midair collisions," *The Lincoln Laboratory Journal*, Vol. 2, No. 3, 1989, pp. 437–457.

[17] Krozel, J., and Peters, M., "Conflict detection and resolution for free flight," *Air Traffic Control Quarterly*, Vol. 5, No. 3, 1997, pp. 181–212.

[18] Pradeep, P., and Wei, P., "Energy efficient arrival with RTA constraint for urban eVTOL operations," *2018 AIAA Aerospace Sciences Meeting*, 2018, p. 2008.

[19] Lovering, Z., "Vahana Configuration Trade Study - Part I – Vahana," , Dec 2016. URL `https://vahana.aero/vahana-configuration-trade-study-part-i-47729eed1cdf`.

[20] 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.

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

[22] 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.

[23] 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.

[24] Enright, P. J., and Conway, B. A., "Discrete approximations to optimal trajectories using direct transcription and nonlinear programming," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992, pp. 994–1002.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] Acikmese, B., and Ploen, S. R., "Convex programming approach to powered descent guidance for mars landing," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 5, 2007, pp. 1353–1366.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

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

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

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

[40] 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.

[41] 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.

[42] 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.

[43] 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.

[44] 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.

[45] 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.

[46] 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.

[47] 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.

[48] 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.

[49] Sigurd, K., and How, J., "UAV trajectory design using total field collision avoidance," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003, p. 5728.

[50] Langelaan, J., and Rock, S., "Towards autonomous UAV flight in forests," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005, p. 5870.

[51] 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.

[52] Zhang, T., Kahn, G., Levine, S., and Abbeel, P., "Learning deep control policies for autonomous aerial vehicles with mpc-guided policy search," *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, IEEE, 2016, pp. 528–535.

[53] 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.

[54] 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.

[55] Yang, X., and Wei, P., "Autonomous On-Demand Free Flight Operations in Urban Air Mobility using Monte Carlo Tree Search," 2018.

[56] 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.

[57] 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.

[58] 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.

[59] Van Den Berg, J., Guy, S. J., Lin, M., and Manocha, D., "Reciprocal n-body collision avoidance," *Robotics research*, Springer, 2011, pp. 3–19.

[60] 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.

[61] Yang, X., and Wei, P., "Autonomous On-Demand Free Flight Operations in Urban Air Mobility using Monte Carlo Tree Search," , 2018. URL `https://www.youtube.com/watch?v=UARP_-RnubA&feature=youtu.be`.

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

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

[64] 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.

[65] 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:10.1038/nature16961.

[66] 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.

[67] 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.

[68] 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:10.1038/nature24270.

[69] Alejo, D., Conde, R., Cobano, J., and Ollero, A., "Multi-UAV collision avoidance with separation assurance under uncertainties," *Mechatronics, 2009. ICM 2009. IEEE International Conference on*, IEEE, 2009, pp. 1–6.

[70] Weiss, G., *Multiagent systems: a modern approach to distributed artificial intelligence*, MIT press, 1999.

[71] Shoham, Y., and Leyton-Brown, K., *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*, Cambridge University Press, 2008.

[72] Kothari, M., Postlethwaite, I., and Gu, D.-W., "Multi-UAV path planning in obstacle rich environments using rapidly-exploring random trees," *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. 3069–3074.

[73] 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.

[74] Vlassis, N., "A concise introduction to multiagent systems and distributed artificial intelligence," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, Vol. 1, No. 1, 2007, pp. 1–71.

[75] "Unmanned Aerial System Traffic Management Fact Sheet," `https://www.nasa.gov/ames/utm2015`, 2015. Accessed: 2018-01-19.

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

[77] 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.

[78] 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:10.2514/6.2018-3365.

[79] 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:10.2514/atcq.23.2-3.137.

[80] 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.

[81] 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.

[82] Google, "Google Maps," , 2019. URL `https://www.google.com/maps`, accessed: 2019-03-12.

[83] Administration, F. A., "Near Midair Collision Reporting," , 2017. URL `http://www.faraim.org/aim/aim-4-03-14-530.html`, accessed: 2018-11-14.