Safety Assured Online Guidance with Airborne Separation for Urban Air Mobility Operations in Uncertain Environments

Pengcheng Wu, Xuxi Yang, Peng Wei, *Member, IEEE,* and Jun Chen, *Member, IEEE*

***Abstract*—The concept of Urban Air Mobility (UAM) proposes to use revolutionary new electrical vertical takeoff and landing (eVTOL) aircraft to provide efficient and on-demand air transportation service between places previously underserved by the current aviation market. A key challenge for the success of UAM is how to manage large-scale autonomous flight operations with safety guarantee in high-density, dynamic and uncertain airspace environments. In this paper, a safety assured decentralized online guidance algorithm with airborne self-separation capability is proposed and analyzed for multi-aircraft autonomous flight operations under uncertainties. The problem is formulated as a multi-agent Markov Decision Process with continuous action space and is solved by a customized decentralized online algorithm based on Monte Carlo Tree Search (MCTS). To guarantee the safety of real-time autonomous flight operations in uncertain environments, the formulation of loss of chance constrained separation is introduced and integrated with the proposed MCTS algorithm. In addition, Gaussian process regression along with Bayesian optimization is employed to discretize the continuous action space, which helps shorten the flight time. A comprehensive numerical study shows that the proposed algorithm can provide safe onboard guidance with guaranteed low near mid-air collision probability in uncertain and high-density airspace environments.**

***Index Terms*—Autonomous System, Monte Carlo Tree Search, Chance Constraints**

I. Introduction

A. Motivation

In order to alleviate ground transportation congestion in urban areas, the concept of Urban Air Mobility (UAM) has received significant attention from various stakeholders, such as NASA, the Federal Aviation Administration (FAA), and airlines [1], [2]. The vision of UAM is to use revolutionary new electrical vertical takeoff and landing (eVTOL) aircraft to provide efficient and on-demand air transportation service between places previously underserved by the current aviation market. Companies such as Airbus, Boeing, Bell, Joby, Archer, Lilium, and Aurora Flight Sciences are competing to build and test their newly designed eVTOL aircraft [3]. The UAM operations are expected to bring significant changes to the city infrastructures and people's daily commutes. To well serve a significant proportion of urban transportation demand, UAM will introduce a large number of eVTOL aircraft in the limited urban airspace [4]. Therefore, a key challenge for the success of UAM is how to manage large-scale autonomous flight operations with safety guarantee in high-density, dynamic and uncertain airspace environments. In this paper, we investigate multi-aircraft online decision making to resolve conflicts and reach their destinations. The decisions are generated onboard and safety assured.

Currently, most researches on UAM are proposed based on structured airspace, where static corridors are specified and the eVTOL aircraft need to follow the fixed routes inside the corridors for conservative safety concerns [5]. The structured airspace concept borrowed from the current air traffic management (ATM) system is safe but not efficient for UAM because the traffic density of UAM will be much higher than ATM. To increase the airspace's capacity, the FAA and NASA are investigating the free-flight airspace framework with trajectory-based operations [6]. This paper will consider the free flight concept because previous studies show that free flight with self-separation ability can handle higher air traffic density even in uncertain environments [7]. Without the protection of structured airway, aircraft need to rely on accurate localization provided by navigation technologies such as Automatic Dependence Surveillance-Broadcast (ADSB) and Global Positioning System (GPS) [8]. Thus the location uncertainty or error of an aircraft is a critical issue to ensure safety. The location uncertainties usually arise from three different aspects: sensor inaccuracy, dynamical uncertainty due to vehicle performance, and environment uncertainties like winds. This paper will introduce a novel loss of chance constrained separation (LOCCS) formulation to ensure the probability of conflicts under required risk level. Moreover, when considering the potential conflicts between two aircraft, the location uncertainties of both aircraft are efficiently addressed as one common uncertainty through a new concept of relative uncertainty. The chance constraint is applied directly to the common uncertainty to avoid time-consuming sampling-based evaluation while ensuring safety.

This paper considers the decentralized decision making setting, in which each aircraft will act based on the sensorial or broadcast information. Model predictive control [9] is a commonly used method to solve the conflict avoidance problem, but its computational load is high thus not ideal for online

P. Wu is with the Department of Mechanical and Aerospace Engineering, University of California San Diego, La Jolla, CA 92093, and also with the Department of Aerospace Engineering, San Diego State University, San Diego, CA 92182 pcwupat@ucsd.edu, pwu@sdsu.edu

X. Yang was with the Department of Aerospace Engineering, Iowa State University, Ames, IA 50011 xuxiyang@iastate.edu

P. Wei is with the Department of Mechanical and Aerospace Engineering, George Washington University, Washington, DC 20052 pwei@gwu.edu

J. Chen (corresponding author) is with the Department of Aerospace Engineering, San Diego State University, San Diego, CA 92182 jun.chen@sdsu.edu

settings. Potential field method [10] is fast but it requires the discretization of the state space, which sacrifices the accuracy. Chai et al. developed a real-time strategy based on deep neural network to generate the path for a reentry vehicle [11]. Deep neural network can achieve a good performance, but it needs extremely long training time. In addition, the conflict avoidance problem is also modeled as a Markov Decision Process (MDP) and often solved offline [12]. Offline algorithms need a long time to compute the optimal policy with a discrete MDP formulation. To account for the dynamic changes in the uncertain environment, online algorithms are preferred. It only plans for the current states not the whole state space, thus discrete MDP formulations are not needed. The Monte Carlo Tree Search (MCTS) algorithm is suitable to solve this problem online without model training. However, only a limited set of discretized actions can be taken by the aircraft at each time step, which makes it difficult to generate a smooth trajectory. The MCTS with discrete actions costs extra energy since the heading angle of aircraft will keep changing while the aircraft approaches the desination [13]. This paper will overcome this limitation by developing the MCTS algorithm with continuous action space to achieve smooth and energy-efficient trajectories.

B. Related Work

The essentials of path planning for a multi-agent system in an environment are to find conflict-free paths for every agent in the system. The methods to achieve conflict detection and avoidance between aircraft can be categorized into centralized and decentralized frameworks.

For centralized methods, a central supervising controller is usually developed to provide all aircraft the collision-free trajectories. In such settings, the central controller usually knows all the key knowledge of the system like the states of all the aircraft in the system, the geometry of obstacles, the constraints to which trajectories are subject, and the terminal conditions. Incorporating all the knowledge obtained, the central controller computes the whole individual trajectory for every aircraft in the system before conducting flight operations. The central controller is often modeled as an optimal control problem. There are various methods which contribute to solving for such an optimal control problem, like semi-definite programming [14], mixed-integer linear programming (MILP) [15], [16], nonlinear programming [17], [18], mixed-integer quadratic programming [19], second-order cone programming [20], evolutionary techniques [21], and particle swarm optimization [22]. Chai et al. developed a unified multi-objective optimization scheme for aeroassisted vehicle trajectory planning and solved it using an evolutionary optimization algorithm [23]. However, the intensified computational load will lead to the intractability of those aforementioned computational geometry methods when the dimensions of the state space grow [24], [25]. Some sampling-based methods are proposed to address such issues, like Rapidly-exploring Random Tree (RRT) [26] and probabilistic roadmaps [27]. For these centralized methods, they are usually intended for the pursuit of globally optimal solutions. However, for a multi-agent system, when the number of aircraft in the system grows, the computational time also increases. In addition, it is often required to run these centralized methods again if the information of the environment changes, which makes it infeasible for many online implementations.

For decentralized methods, the conflicts between different aircraft in the system are handled individually by each aircraft in a distributed way. There are non-cooperative and cooperative ways for decentralized methods. The methods are considered non-cooperative when the communication between different aircraft in the system cannot be successfully established. Methods like model predictive control [9] and reinforcement learning [28] can contribute to the conflict avoidance, but their computational load is very high and thus the cost is too expensive. MCTS algorithms have been used to the resolution of conflict avoidance, but only a few discretized actions can be taken by aircraft at each time step, which may lead to unsmooth trajectories [13]. There is also a lot of existing literature working on the cooperative scenarios. Methods based on message-passing schemes in [29] are proposed to resolve conflicts without requiring the formulation of a joint optimization problem among all the agents in the system. The method of MILP in [30] is used to allot a time slot for every agent in the system to obtain conflict-free trajectories. In this paper, we will focus on the cooperative scenarios using the decentralized online algorithm of MCTS with continuous action space.

The multi-agent system may be subject to various forms of uncertainty during path planning. Ideally, we expect to identify a path for a multi-agent system, which can absolutely guarantee that no conflicts will happen if all the aircraft in the system follow the identified path. In other words, the probability of conflicts is 0%. However, due to the existence of uncertainty, it is impossible for us to achieve this. Instead, we seek to identify a path which can assure that the probability of conflicts is within a safety bound (not 0% though), following the identified path [31]. Thus, accounting for the knowledge of uncertainty in path planning is regarded as an essential step to achieve safety assurance. In [32], a probabilistic map is constructed using a likelihood function and a safe UAV path is then generated by solving a probability minimization problem. In [33], the sampling-based Monte Carlo method is accurate enough to estimate conflicts between aircraft stochastically, but it sacrifices computational efficiency since it takes much computational time to figure out the probability of conflict occurrence. To achieve the balance between planning conservatism and efficiency, the stochastic constraints can be reformulated as tightened deterministic constraints through chance constraints formulation. Blackmore et al. [34], [35] proposed chance-constrained programming model to consider various uncertainties for conflict avoidance problem, but such formulations based on MILP or constrained nonlinear programs are often computationally expensive, which may scale poorly as the dimensions of configuration spaces increase. Chai et al. planned trajectory for a single vehicle through converting probabilistic constraints to deterministic ones [36]. Luders et al. [37] present Chance Constrained Rapidly-exploring Random Tree (CCRRT) which uses sampling-based methods to identify paths for linear systems subject to uncertainty. Sampling-based algorithms like CCRRT scale well because they perform trajectory-wise constraint checking, but such paths do not satisfy optimality guarantees. In addition, when considering inter-agent conflict checking, both agents may have uncertainties but the aforementioned work treat both uncertainties separately, which in turn makes designing an analysis complicated and the method hard to be generalized. In this paper, we will build an LOCCS formulation to account for the knowledge of uncertainty and incorporate it in MCTS algorithm with continuous action space to identify safety assured trajectories for every aircraft in a multi-agent system in the presence of location uncertainty. Also, we will employ the transformation of relative uncertainty to convert the location uncertainties of both agents into a common one when performing conflict checking between different agents.

C. Contributions and Structure

UAM has its own constraints and concerns like traffic density and dynamic complexity. In this paper, we put forward a safety assured decentralized online algorithm for a multi-agent system of UAM. We first formulate a conflict-free guidance system in the presence of location uncertainty as a multi-agent Markov Decision Process (MMDP) problem, and then solve for this MMDP problem using the MCTS algorithm with continuous action space incorporating LOCCS formulation. In this paper, we focus on UAM free flight setting up, where no obstacles and constraints in structured airspace are considered. Indeed, our method can be extended to more general multi-agent systems.

The major contributions of this paper are summarized as follows:

1) The LOCCS formulation is incorporated into MCTS algorithm to deal with location uncertainty. It transforms the original stochastic uncertainty into deterministic ones, greatly saving computational time while assuring conflict avoidance with high probability. Also, the method of relative uncertainty transformation is introduced and implemented within the UAM free flight airspace, and therefore different uncertainties can be transformed into common one and then be handled efficiently in online fashion, even when the number of agents is very large, as indicated in numerical study;
2) The MCTS algorithm with continuous action space is developed to provide a safe, efficient and smooth trajectory. This algorithm is intended to explore continuous options of actions through Bayesian optimization with Gaussian process regression, and to exploit the current candidate actions to evaluate them accurately. Numerical study justifies that this shortens the flight time of the aircraft in UAM.

The structure of this paper is organized as follows: In section II, the description of the problem and the MMDP formulation are stated. In section III, the LOCCS formulation is introduced to convert the stochastic location uncertainty into deterministic constraints. In section IV, the MCTS algorithm with continuous action space incorporating LOCCS formulation is developed to solve the presented MMDP problem. In section V, a comprehensive numerical study is conducted to demonstrate the feasibility and efficiency of the proposed algorithm. Finally, we conclude in section VI.

II. Problem Formulation

A. Problem Statement

This paper focuses on the problem setting where multiple eVTOL aircraft are navigated from their departure points to respective destinations through a series of control actions with the intent to avoid conflicts among them. As a problem of sequential decision-making, we can formulate it as a Markov Decision Process (MDP) problem. In addition, instead of the MDP formulation which considers a single aircraft only, we formulate Multi-agent Markov Decision Process (MMDP) as an extension of MDP to fit the setting of the multi-agent system presented in this paper. For every decision-making procedure in an MMDP problem, the aircraft's action is determined by its state which contains all the information to decide the optimal action for the current state.

To achieve the goal, an algorithm which acts on every aircraft in the multi-agent system is required. This algorithm is developed based on the method of Monte Carlo Tree Search (MCTS). In this article, we take into account the scenarios of high-density free flight airspace: we will run the proposed online algorithm on every individual aircraft in the multi-agent system with the purpose of avoiding inter-agent conflicts. All the aircraft in the system are assumed to fly at the same altitude (flight level) and thus only horizontal actions need to be considered. In addition, the location uncertainty of the aircraft is explored and the performance of the proposed algorithm is tested under different levels of uncertainty.

The objectives of this specific MMDP problem are presented as follows:

1) To minimize the total number of potential conflicts among all the aircraft;
2) To navigate all the aircraft in a multi-agent system to reach their respective destinations as soon as possible.

We will introduce a reward function later to capture both objectives above. Building upon the reward function, this problem will be formally reformulated as an MMDP problem in the next section.

B. MMDP Formulation

Since the middle of the 20th century, MDPs have been well studied and widely applied in many areas like robotics, control, economics and so on [38], [39]. At every time step in the MDP formulation, an agent in the system selects a feasible action a according to the knowledge of current state s. Then by following the underlying state transition probability, the agent proceeds to next state s^0 and receives a reward.

To be specific, an MDP formulation is composed of the following four components:

1) The state space S containing all the possible states of an agent;
2) The action space A containing all the possible actions an agent can select;

3) Transition function $T(s_{t+1}|s_t,a_t)$ portraying the probability of reaching state s_{t+1}, given current state s_t and action a_t;
4) The reward function $Reward(s_t,a_t,s_{t+1})$ which decides the reward received after transition from state s to state s' according to action a.

In an MDP problem, a policy π is a mapping from a state to a specific action:

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

The purpose of our MDP formulation is to seek an optimal policy which maximizes the expected cumulative rewards over all the steps in the future, started from any initial state:

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

There are two important notions in MDP: Q-function and value function. The optimal Q-function $Q^*(s,a)$ represents the expected cumulative reward received by an agent which starts from state s, selects action a. Thus, $Q^*(s,a)$ indicates to what extent it is good for an agent to select an action a in a state s. The optimal value function $V^*(s)$ suggests the maximum expected total reward for the agents which start from state s, and can be illustrated as the maximum of $Q^*(s,a)$ over all possible actions:

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

In this paper, an MMDP problem is formulated as an extension of the aforementioned MDP to fit the setting of a multi-agent system. The MMDP formulation presented in this paper is made up of the following five components.

1) Continuous State Space: The state contains the necessary knowledge for the algorithm to perform actions on every single aircraft in the system: the position (x,y), speed v, heading angle ψ, and goal position (g_x,g_y) for every aircraft in the system. Capturing the information of every aircraft in the system, the state for the MMDP formulation turns out to be an $n \times 6$ matrix, where n indicates the quantity of aircraft in the system and each row of the matrix exhibits the knowledge for a single aircraft in the system.

It is worth noting that in this paper the state space being discussed is continuous. For instance, all the entries of a state can be taken continuously. However, there is no way that we can enumerate all the possible mappings from states to actions, and therefore it is not clear how to obtain the best representation of the policy. For MDP-based algorithms previously discussed, to seek the solutions to conflict avoidance problems, some methods have been presented to represent the policy through the discretization of the state space S and the action space A based on grids, or employing policy compression techniques [40]. Alternatively, the advantage of the proposed MCTS algorithm in this paper is that it doesn't require the discretization of the state space. For each state, the actions can be generated by MCTS algorithm for the aircraft to take in real time.

2) Continuous Action Space: At every time step, the aircraft can take actions to turn its heading angle at a certain rate. To be specific, the advisory of heading angle for every individual aircraft constitutes the set of continuous actions $A = \{a \in \mathbb{R} | -5\,\mathrm{deg/s} \leq a \leq 5\,\mathrm{deg/s}\}$, where the plus sign indicates right turn and the minus sign indicates left turn. The change rate of the heading angle can be decided afterwards. At each time step, the proposed algorithm can run in real time to select an action from the action set for the aircraft according to the knowledge of the current state. After running, the aircraft will maintain the selected action during the current time step.

3) Dynamic Model with Location Uncertainty: Built on the information from current state and action, the following kinematic model is used to find the state transition for every aircraft in the system:

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

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

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

where v denotes the cruise speed, ψ indicates the heading angle, and a indicates the selected action which determines the change rate of heading angle for an aircraft.

After an aircraft executes an action, the aircraft speed is held constant during one time step. In addition, we also consider some disturbance that influences the aircraft speed and its changing rate of the heading angle. This ultimately accounts for the location uncertainty of the aircraft, which can be modeled as a Gaussian distribution centered at the expected position of the aircraft after one time step with a certain covariance. The disturbance here aims to capture the uncertainty from environment and aircraft dynamics respectively. When it comes to the conflict checking between two aircraft in the system, both location uncertainties of each aircraft should be considered. Instead of dealing with both uncertainties separately, we can introduce a relative uncertainty transformation to get them settled together, which will be fully discussed in next section.

4) Terminal State: When two aircraft cannot maintain a minimum separation requirement, a Loss of Separation (LOS) event occurs, which is defined formally as follows:

Definition 1 **(LOS Event)** An LOS event occurs when

$$\|x_{jt} - x_{it}\| \leq r_j + r_i \tag{7}$$

where x_{jt} and x_{it} are the locations of the aircraft j and the aircraft i at time t respectively. r_j and r_i are the minimum safety ranges for both aircraft to stay away from each other to ensure safety, which depends on the speed of the aircraft [41]. In this paper, for simplicity, r_j and r_i are assumed to be constant at any time t.

However, given unknown location uncertainty of the aircraft, both x_{it} and x_{jt} should be treated as random variables. Thus it's impossible to figure out the real distance between two aircraft with the information of their states only. Instead we turn to ensure that the probability of LOS event occurrence is less than a given risk level. To handle this probabilistic measurement, we propose a new concept named as Loss of Chance Constrained Separation (**LOCCS**) event, which will be fully defined and detailedly explained in Section III. Based on LOCCS, we can further prove that whether the probability of LOS event occurrence is smaller than a given risk level or not can be judged through evaluating the distance between a point and an ellipse and then checking whether that distance is larger than a threshold value, which depends on the given risk level.

Building upon the aforementioned aircraft separation requirements, the terminal state in the MMDP formulation is made up of two different categories of states indicated as follows:

1) An LOCCS event occurs, i.e., the probability that an LOS event occurs is greater than a given risk level (denoted as an LOCCS state);
2) The agent reaches its goal position, i.e., the distance between the agent and its corresponding destination is less than a threshold (denoted as a goal state).

In addition to the terminal states listed above, all the other scenarios encountered can be stacked together and defined as non-terminal states.

5) Reward Function: The primary objective of our model is to minimize the total travel time of all aircraft while maintaining safety. This bi-objective can be handled with the following reward function:

$$reward(s) = \begin{cases} \frac{\max d(\boldsymbol{x}_t;g) - d(\boldsymbol{x}_t;g)}{\max d(\boldsymbol{x}_t;g)}; & s \text{ is a non-terminal state} \\ 0; & s \text{ is an LOCCS state} \\ 1; & s \text{ is a goal state} \end{cases} \tag{8}$$

where $d(x_t;g)$ is the distance between the aircraft and its goal position; max $d(x_t;g)$ is the maximum distance, for example, the diagonal of a square map or the diameter of a convex map. Therefore, based on the current distance from the aircraft to its destination, the aircraft will be rewarded a positive value between 0 and 1, if it is not at an LOCCS state.

Based on the above reward function for an individual aircraft, the actual reward function for the MMDP is the sum of all individual rewards, which is shown as follows:

$$Reward(s) = \sum_{i=1}^{N} reward_i(s) \tag{9}$$

where N is the number of aircraft in the system.

In summary, this reward function will encourage the aircraft to determine actions that drive it closer to its destination for a positive reward and avoid any LOCCS event, which is rewarded zero.

III. LOCCS Formulation

A. Risk Domain Definition

Uncertainty is a major concern when it comes to the path planning of UAM. For an eVTOL in a multi-agent system, its location x_t at a particular time t may be stochastic, due to uncertainty arising from the inaccurate sensor of self-position or environmental disturbance like wind. Especially, under the assumption of Gaussian distribution, the location x_t obeys

$$x_t \sim N(\mu_t; \Sigma) \tag{10}$$

where $N(\mu_t; \Sigma)$ represents a Gaussian distribution which has a time-varying mean μ_t and a time-invariant covariance Σ. That is, the expected location for each aircraft is the mean of the Gaussian distribution, while the real location of each aircraft obeys the Gaussian distribution.

To operate safely in a dynamic environment, each eVTOL should seek to avoid conflicts with other eVTOLs in the system. This can be achieved by introducing the following constraints

$$x_t \notin X_t \quad \forall t \\ \text{where } X_t := \bigcup_{i=1}^{N-1} X_{ti} \tag{11}$$

In the above constraints, we use X_{ti} to denote the area in which aircraft i may be located at time t due to aircraft location uncertainty, where $i \in \{1;2;:::;N-1\}$ and N is the total number of all the aircraft in the system. X_t is formed by all the other aircraft that the current aircraft seeks to avoid. We call such area as the possible region for all the other aircraft i at time t. Note that, from the perspective of the current aircraft, all the other aircraft in the system actually can be viewed as obstacles to avoid. The time dependence of X_t allows the inclusion of either static or dynamic obstacles. In this paper, for the current aircraft in the system, all the other aircraft need to be treated as dynamic ones.

From the perspective of the current aircraft, the possible region of another aircraft i at time step t can be modeled by the following equation, by assuming Gaussian uncertainty,

$$X_{ti} = \{x \in \mathrm{R}^d \mid \|x - \boldsymbol{c}_{ti}\| \leq r_i\} \quad \forall t; i: \\ \boldsymbol{c}_{ti} \sim N(\mu_{ti}; \Sigma_i) \tag{12}$$

where r_i is the minimum safety range for the current aircraft to stay away from the ith aircraft to ensure safety. In addition, $\boldsymbol{c}_{ti}$ represents the position of the ith aircraft at time t, which is an independent Gaussian random variable, i.e., $\boldsymbol{c}_{ti} \sim N(\mu_{ti}; \Sigma_i)$ with a time-varying mean μ_{ti} and a time-invariant covariance Σ_i.

Given the concept of LOS event defined in last section, this paper aims to find probabilistically guaranteed conflict-free paths for a set of eVTOLs to reach their goal positions, such that the probability of the LOS event occurrence between any two aircraft in the system at any time is less than a certain threshold, i.e.,

$$\Pr(\text{LOS event}) \leq \delta \tag{13}$$

where δ is a prescribed threshold, called risk level.

We aim to quantify the probability of an aircraft to avoid conflicts with other aircraft considering location uncertainty as presented in Eq. (13). To realize this goal, we first define the risk domain of a d-dimensional Gaussian random variable.

Definition 2 **(Risk Domain)** A set $D \subseteq \mathrm{R}^d$ that satisfies

$$\Pr(X \in D) \geq 1 - \delta \tag{14}$$

is called a risk domain at risk level α of a random variable X.

We can convert the possible region X_{ti} introduced in Eq. (12) into a risk domain D at a given risk level α (or a confidence level $1-\alpha$). Especially when the aircraft location c_{ti} is assumed to obey Gaussian distribution and the safety range is not considered (i.e. $r_i = 0$), the boundary of the corresponding risk domain D for c_{ti} only can be proved to be a circle or an ellipse, according to Lemma 1 in our previous work [42], [43] as follows. For the uncertainty obeying Gaussian distribution, an ellipse would be a good choice since it is just the contour of the probability density function of Gaussian distribution.

Lemma 1 Let $X \in \mathbb{R}^d$ be a d-dimensional random variable that obeys a d-dimensional Gaussian distribution $N_d(\mu; \Sigma)$, then the following set

$$D := \{X \mid (X-\mu)^{\top}\Sigma^{-1}(X-\mu) \leq F_d^{-1}(1-\alpha)\} \quad (15)$$

specifies the risk domain of X at risk level α, where F_d^{-1} is the inverse mapping of the cumulative distribution function (CDF) of χ^2 distribution with d degrees of freedom. ■

We first consider simple scenarios where only aircraft i has location uncertainty when checking conflicts between the current aircraft and the other aircraft i. It is straightforward to see that if the distance between the current aircraft and the risk domain D of aircraft i is less than the sum of safety ranges $r_i + r$ of both aircraft, then the probability of LOS event occurrence between both aircraft must be less than the given risk level α. In this way, the introduction of risk domain allows stochastic constraints to be represented as equivalent deterministic constraints. However, the current aircraft and aircraft i may both have location uncertainty. To cope with such scenarios, we introduce the transformation of relative uncertainty between the current aircraft and aircraft i in next subsection, so as to reduce the location uncertainty of both aircraft into a common one, which greatly simplifies the process of feasibility checking for the probabilistic bounds.

B. Transformation of Relative Uncertainty

When considering conflict checking between two aircraft, we can introduce the transformation of relative uncertainty, which transforms the uncertainty of both aircraft into one common uncertainty. For the simplicity of notation, let random variable X denote the location c_{ti} of an aircraft i and random variable Y denote the location x_t of the current aircraft at time step t. Consider that both $\mathrm{Var}(X)$ and $\mathrm{Var}(Y)$ are expressed in a common coordinate system. It follows that the new random variable $Z = (X - Y)$ represents the relative position between the aircraft i and the current aircraft given uncertainty, which also follows a Gaussian distribution [44].

With this, the risk domain established above for the situations where only one aircraft has location uncertainty can be extended to the situations where there are both uncertainties. we can also build the corresponding risk domain for the random variable Z which represents the relative position between the aircraft i and the current aircraft at the time step t,

$$(Z-\mu'')^{T}\Sigma''^{-1}(Z-\mu'') = F_2^{-1}(1-\alpha) \quad (16)$$

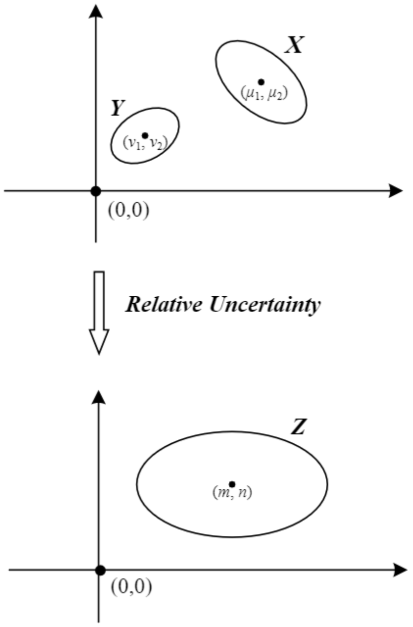


Fig. 1. Illustration of the transformation of relative uncertainty

where α is the given risk level, F_2^{-1} represents the inverse mapping of the cumulative distribution function (CDF) of χ^2 distribution with 2 degrees of freedom, $\mu'' = (m; n)^T$ is the mean vector after transformation, and Σ'' is the covariance after transformation.

It's easy to find that the boundary of Eq. (16) is also a circle or an ellipse. Further details can be found in [43]. To check whether the current aircraft would conflict with the other aircraft i, it's only required to check the minimum distance $d_{\min}$ between the origin $(0; 0)$ and the risk domain of Z in Eq. (16) whose center is $\mu'' = (m; n)^T$, as shown in Fig. 1. If $d_{\min}$ is greater than the safety range, the probability of LOS event occurrence will be bounded by the prescribed risk level α. Building upon the exploration of the transformation of relative uncertainty above, we successfully transform both uncertainties of two aircraft into a common one, in lieu of addressing them separately.

C. LOCCS Definition

In this paper, the safety ranges of both aircraft are non-negligible. The radius of the safety range for the current aircraft and the other aircraft i are r and r_i respectively. Given that both aircraft have location uncertainty, we apply the transformation of relative uncertainty presented in last subsection to convert both uncertainties into a common one. After that, the possible region X_{ti} of the aircraft i at time t as defined in Eq. (12) can be represented by the union of a series of circles with radius r_i, where the circle's center c_{ti}, i.e. the location of the aircraft i, follows a Gaussian distribution.

We can first obtain the risk domain D_{ti} for the location c_{ti} of aircraft i at time t according to Lemma 1, assuming that $r_i = 0$. Next, when considering the safety range r_i of aircraft i, the corresponding risk domain for aircraft i at time step t is in fact the enveloping area of a series of circles whose centers are inside or on the boundary of D_{ti}. We denote this actual risk domain as D_{ti}^{range}, which is a larger area than D_{ti} as shown in Fig. 2.

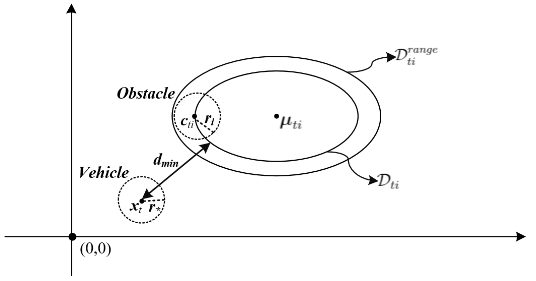


Fig. 2. Illustration of LOCCS formulation

Since there is no explicit expression of the actual risk domain D_{ti}^{range}, we will check the feasibility of the chance constraints by measuring the minimum distance $d_{\min}$ between the current aircraft's location x_t and the risk domain D_{ti} for the location c_{ti} of aircraft i at time t. In fact, $d_{\min}$ can be treated as the distance between a point and an ellipse, which can be evaluated explicitly according to [45]. Then we check the feasibility of the deterministic constraints using the inequality

$$d_{\min}(x_t, D_{ti}) > r_i + r \tag{17}$$

if it holds, then

$$\begin{aligned} &\Pr(\text{LOS event} \mid \text{Eq. (17) holds}) \\ &= \Pr(\|x_t - c_{ti}\| \le r_i + r \mid x_t \notin D_{ti}^{range}) \\ &\le \Pr(c_{ti} \notin D_{ti}) \\ &= \epsilon \end{aligned} \tag{18}$$

Based on the above discussion, we can now quantitatively define the concept of an LOCCS event, which has already been mentioned in last section.

Definition 3 **(LOCCS Event)** An LOCCS event occurs when

$$d_{\min}(x_t, D_{ti}) \le r_i + r \tag{19}$$

In summary, given a risk level ϵ which denotes the probabilistic bound on the chance that the current aircraft conflicts with any other aircraft i at any time step t, we can convert the stochastic constraint Eq. (18) into a deterministic constraint Eq. (17) equivalently, in consideration of the safety ranges and location uncertainty of both aircraft.

IV. Solution Method

We are seeking to develop an online algorithm to provide navigation commands for the aircraft, which guides every individual aircraft in the system to reach their respective destinations without conflicting with each other as much as possible. When it comes to the procedure of decision-making, the control action can be determined by the proposed algorithm according to the knowledge of current state. Following the algorithm's control action, the aircraft can reach their destinations while avoiding LOCCS events. In this paper, the original problem is formulated as an MMDP problem and solved by algorithms based on MCTS. For original MCTS, readers can refer to [40] and in this paper we will focus on extending original MCTS algorithm to deal with the scenarios of continuous action space and incorporate LOCCS formulation we built in last section.

A. Bayesian Optimization with Gaussian Process Regression

To globally optimize the black box functions, Bayesian optimization can be introduced as a sequential policy without resort to derivatives [46]. The principle of Bayesian optimization is: given the assumption that the unknown function is sampled from a prior Gaussian process, a posterior distribution is kept for this function as we make sequential observations. Next we will give a concise introduction to Gaussian process and Bayesian optimization.

For Gaussian process (GP), it offers a strong and convenient way to explore non-parametric statistical models over the Hilbert space consisting of functions. Specifically, a GP is a stochastic process such that any finite subcollection of random variables has a multivariate Gaussian distribution [47]. In particular, a collection of random variables $\{f(x) : x \in X\}$ is said to be drawn from a GP with mean function $m(\cdot)$ and covariance function $k(\cdot,\cdot)$ if for any finite set of elements $x_1, \cdots, x_m \in X$, the associated finite set of random variables $f(x_1), \cdots, f(x_m)$ has distribution

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(x_1) \\ \vdots \\ m(x_m) \end{bmatrix}, \begin{bmatrix} k(x_1, x_1) & \cdots & k(x_1, x_m) \\ \vdots & \ddots & \vdots \\ k(x_m, x_1) & \cdots & k(x_m, x_m) \end{bmatrix} \right) \tag{20}$$

and for simplicity, this equation can be abbreviated as:

$$f(\cdot) \sim GP(m(\cdot), k(\cdot,\cdot)) \tag{21}$$

Next we are going to employ GP to make predictions for the function value evaluated at a sampling point x_*. Given a training set $X = \{(x_i, f(x_i)) : i = 1, \ldots, n\}$, then the function value f_* evaluated at the sampling point x_* will obey the following distribution:

$$\begin{aligned} &f_* \mid x_*, X \\ &\sim \mathcal{N}(K(x_*, X)K(X, X)^{-1}f, \\ &\quad K(x_*, x_*) - K(x_*, X)K(X, X)^{-1}K(X, x_*)) \end{aligned} \tag{22}$$

where f is a vector comprised of all the function values, and k is a matrix comprised of entries indicating the covariance function $k(\cdot,\cdot)$. It is straightforward to show that GP not only offers the knowledge of the possible value of the function f, but more importantly, the knowledge of the uncertainty around that function value.

Provided that the intended function is captured from a GP prior and then based on the data of observation $X = \{(x_i, f(x_i)) : i = 1, \ldots, n\}$, it follows that a posterior over the space of functions can be obtained. An acquisition function can be introduced to decide which point to be evaluated next according to a proxy optimization $x_{\text{next}} = \arg\max_x a(x)$, and this is temporarily the best guess to approximate the global optima. The criterion of Upper Confidence Bound (UCB) [48] is a good choice to serve as an acquisition function. It is intended to minimize the regret (the expected loss due to the selection of sub-optimal action) during the process of optimization. The UCB acquisition function reads:

$$a_{\mathrm{UCB}}(x|X) = \ (x|X) + \ \ (x|X) \tag{23}$$

where is a tuning parameter aimed at achieving a trade-off between mean and variance.

Therefore, Bayesian optimization is comprised of two main components:

1) A Bayesian statistical model intended to capture the objective;
2) An acquisition function intended to find next sampling point.

Following the evaluation of the objective on the basis of an initial space-filling set-up, which is often made up of points sampled uniformly at random, these components are iteratively utilized to allocate the budget of evaluations for the function.

B. MCTS with Continuous Action Space

Monte Carlo Tree Search (MCTS) is an approach which is intended to obtain optimal decision-making through sampling randomly over the decision space and generate a search tree based on the consequence of decision-making [49]. It has already been ubiquitously applied in planning problems and games like the artificial intelligent computer program AlphaGo which was used in Go games [50]. MCTS generates a tree in an incremental and asymmetric manner, as shown in Fig. 3. The technical details of MCTS will not be covered in this paper (For further contents readers can refer to [49]). Instead, we are interested in extending the original MCTS to fit the scenarios of continuous action space.

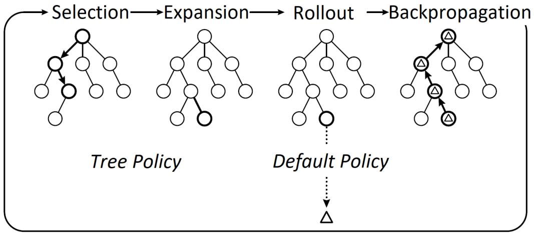


Fig. 3. Illustration for one iteration in MCTS [49]

Extending the tree search algorithm to fit the scenarios of continuous action space poses a great challenge of achieving a balance between searching more extensively to explore more candidate actions through broader expanded nodes and searching more intensively to exploit the current candidate action through deeper search. Recent research work has been dedicated to adapting the tree search algorithm to continuous action space. Truncated Monte Carlo presented by Tesauro et al. [51] trims both candidate actions which are impossible to become the best action, and the candidate actions whose values are close to the current best estimate. Likewise, a trained policy network explored in AlphaGo [52] can be used to restrict the search to actions of high value. The conventional way [53] to perform progressive widening can address continuous action space in consideration of a slow-growing discrete set for sampling actions [54]. In this article, we will take the progressive widening strategy because of its computational efficiency and Bayesian optimization with GP regression will be employed to help us to select the best action.

In previous algorithms applying progressive widening strategy, new actions are added to the tree according to some predefined distribution, which ignores the insight that similar actions may have close Q values if we assume the Q function $Q(\ ;a)$ given in Section II is continuous in a. In this paper, we use GP regression to generalize action value estimates over the entire parameter space, which can guide the agent to add the most promising action to the search tree from the action space. This approach has the following benefits. First, it allows information sharing between all actions under consideration to approximate the values for untried actions. Second, it can identify actions outside of the initial candidates for further exploration by combining the mean and variance of GP regression using UCB acquisition function introduced in Eq. (23) [55]. Last, it can ultimately select actions outside of the candidate set allowing it to improve on less-than-perfect domain knowledge.

Now we turn to illustrate the details of MCTS algorithm with continuous action space. The overall steps of the algorithm are similar to [40], and in this paper we only describe the difference in the selection step due to the incorporation of continuous action space. MCTS algorithm with continuous action space, similar to MCTS algorithm with discrete action space, selects actions by searching ahead. In the step of selection, the current node will select a node (corresponding to an action) from all of its child nodes as next state, which maximizes the function of Upper Confidence Bound Applied to Trees (UCT) introduced in Eq. (24). The UCT function consists of two terms: a mean action value $\overline{X}_j$, and an uncertainty addition [49]:

$$UCT = \overline{X}_j + 2C\sqrt{\frac{2\ln n}{n_j}} \tag{24}$$

The first term is for exploitation, and the second for exploration.

The first term $\overline{X}_j$ can be understood as an exploitation term derived from the formula:

$$\overline{X}_j = Q_j/n_j \tag{25}$$

where n_j is the number of the visits to the child node $s^{k\ \ 1}$ and Q_j is the total reward of all rollouts that passed through this node; The second term can be understood as an exploration term where n is the number of visits to the current node and C is a weighting parameter which seeks to balance exploitation and exploration of tree search. Note that the UCT function in Eq. (24) is different from the UCB acquisition function in Eq. (23), since the latter one is intended for the trade-off between mean and covariance of GP. Also, the UCT function acts when selecting the best child node while the UCB acquisition function acts when expanding child nodes.

For MCTS algorithm with discrete action space, we can readily evaluate the UCT scores for all the candidate child nodes. However, for continuous action space, it is unlikely any more to figure out all the UCT scores for all the candidate actions. Under this circumstance, the strategy of progressive widening can address the continuous action space through limiting the number of actions in the search tree according to the times the node has been visited before and the slow-growing discrete set for sampling actions. Once the attribute of the best available is well estimated, we can take into account the additional actions. To be specific, for action selection, MCTS algorithm runs through either the improvement of value estimation for current child actions by selecting an action of highest UCT score, or the exploration of untried actions by attaching a new action to the current node. The decision can be made according to maintaining the quantity of child actions for a node which is bounded by a sub-linear function $m(s)$ of the number of visits to the current node $n(s)$:

$$m(s) = B \quad n(s) \tag{26}$$

where B and $\quad 2\ (0; 1)$ are two parameters that achieve a trade-off between covering more actions and improving the estimate of fewer actions. At each step of action selection, if the quantity of child actions under the current node s is smaller than $m(s)$, a new action will be attached to the current node s as a child action. Otherwise, an action which owns the highest UCT score will be selected from the existing child actions. On the one hand, UCT score assures that the depth of the search tree grows more rapidly in its promising parts; on the other hand, the progressive widening strategy tells that the search tree also grows broader to explore more actions in some parts of the search tree.

During each step of action selection, the simulation starts from the initial current state s^0, selecting actions repetitively before arriving at an unexpanded node (which means it has no child actions). For each time step k, a decision is made according to the comparison between the quantity of child actions $jA_{s^k \ 1}j$ for the node $s^{k \ 1}$ and the value $m(s^{k \ 1})$ in Eq. (26). If $jA_{s^k \ 1}j \quad m(s^{k \ 1})$, an action a^k will be determined by maximizing the score of UCT presented in Eq. (24); otherwise, the agent will select a new action from the action space, attach it to the search tree, and expand this new edge.

In this paper, assuming the Q value is continuous in a, we use GP regression to fit the Q value from the tree search result, which can inform us to select the most promising action. More specifically, we assume that the current node s has child actions $a_1; \quad ; a_p$ with Q value $Q(s; a_1); \quad ; Q(s_p)$. We use GP to fit the observation data $X = f(a_i; Q(s; a_i)) : i = 1; : : : ; pg$. Then we sample m actions $a_1; \quad ; a_m$ according the Gaussian policy distribution. After evaluating these m actions using the acquisition function in Eq. (23), the proposed algorithm will attach the action with the maximum value to the search tree.

C. MCTS Incorporating LOCCS Formulation

As discussed in Section II, in order to evaluate the reward of a selected action, it is required to conduct conflict checking between two aircraft. Through introducing LOCCS formulation in Section III, we can propose an equivalent deterministic constraint instead of evaluating the probability of LOS event occurrence between two aircraft. Once in compliance with the proposed deterministic constraint, we can guarantee that the probability of LOS event occurrence between two aircraft should be less than a prescribed risk level.

When conducting conflict checking between two aircraft considering location uncertainty at each time step, the location of one aircraft can be represented as an ellipse while the other aircraft can be also represented as another ellipse. Note that the direction of the long axis of the ellipse aligns with the heading orientation of the aircraft. Next employ the relative uncertainty transformation introduced in Section III and we can obtain the LOCCS constraint ($d_{\min} \quad r_i + r \quad$) in Eq. (19). If such a constraint is satisfied, it means the probability of LOS event occurrence indeed exceeds a prescribed risk level. Thus, the aircraft is at the state of LOCCS and the reward $reward(s)$ of the current state should be set to be $reward(s) = 0$, as presented in Eq. (8).

For the conflict detection and avoidance of a multi-agent system, the formulation of conflict between one pair of aircraft in the system is first considered. Then that formulation will be applied to every pair of aircraft in the system so that we can perform inter-agent conflict checking between any two aircraft in the system.

D. Summary

The aforementioned procedure is summarized in **Algorithm 1** as follows. In the pseudo codes, v is used to represent the node and s to represent the state of the node v. $s(v)$ means the state of a node and $v(s)$ means the node created from state s. $Q(v)$ is the total reward of all rollouts that passed through the node v. $N(v)$ is the number of visits to the node v. $d(v)$ indicates the search depth under the node v. In Line 28, Gaussian process regression is introduced to guide the sampling of actions from the continuous action space. In Line 33, the LOCCS formulation is incorporated into MCTS algorithm to help judge whether an aircraft is at an LOCCS state.

This algorithm runs in a distributed way since each aircraft can compute its own action onboard and individually, and thus it can guarantee scalability. The algorithm will check the LOCCS feasibility on every step, so the feasibility under required risk level can also be guaranteed. When it comes to the decision making of an aircraft in the system, it only needs the local information of other aircraft in its neighbourhood. Then according to the local information acquired, it runs the proposed MCTS algorithm, makes decisions and broadcasts its decision to the aircraft nearby. Then the next aircraft takes the above actions repetitively until every aircraft in the system makes decisions and moves forward to their next states respectively.

Algorithm 1 MCTS Algorithm with Continuous Action Space Incorporating LOCCS Formulation

```
function SEARCH($s_0$)
    initial node $v_0$ generated with state $s_0$
    while within the budget of computation do
        $v_l$     TREEPOLICY ($v_0$)
        $reward$     ACTIONPOLICY ($s(v_l)$)
        BACKPROPAGATION ($v_l; reward$)
    return $a$(BESTCHILD($v_0; 0$))

function TREEPOLICY($v$)
    while  NONTERMINAL($v$) and $d(v)$     $d$ do
        if $v$ not fully expanded then
            return EXPAND($v$)
        else
            $v$     BESTCHILD ($v; C$)
    return $v$

function EXPAND($v$)
    select $a$ 2 untried actions $A(s(v))$
    $s(v^0)$ =  PROCEED ($s(v); a$)
    attach the new child $v^0$ to $v$
    return $v^0$

function BESTCHILD($v; C$)
    return  $\underset{v^0 2 \text{ children of } v}{\text{argmax}} \frac{Q(v^0)}{N(v^0)} + C$ q $\frac{2 \ln N(v)}{N(v^0)}$

function ACTIONPOLICY($s$)
    while NONTERMINAL($s$) and $d(v(s))$     $d$ do
        choose $a$ 2 $A(s)$ using GP regression
        $s$     PROCEED ($s; a$)
    return $reward(s)$

function NONTERMINAL($s$)
    if  $s$ 2 LOCCS State or $s$ 2 Goal State  then
        return False
    else
        return True

function BACKPROPAGATION($v; reward$)
    while $v$ is not zero do
        $N(v)$     $N(v) + 1$
        $Q(v)$     $Q(v) + reward$
        $v$     parent of $v$

function PROCEED($s; a$)
    $s^0$     next state from current $s; a$
    $d(v(s^0))$     $d(v(s)) + 1$
    return $s^0$
```

V. Numerical Study

In this section, a series of numerical cases are conducted to demonstrate the feasibility and efficiency of our proposed MCTS algorithm with continuous action space which incorporates LOCCS formulation. To be specific, first the solution quality of our proposed algorithm is compared with another three algorithms in terms of three different aspects. Then several parameter sensitivity tests are performed to quantitatively evaluate the impact that different parameters have on the performance of the proposed algorithm. At last, the average flight time using the proposed algorithm with continuous action space is compared with that of another algorithm with discrete actions under the scenarios of different routes and different traffic densities.

A. Test Settings

To test the solution quality of the proposed algorithm, a simulator is built in Python where multiple aircraft can fly freely in the two dimensional en route airspace above a city, which is intended for future free flight operations in Urban Air Mobility (UAM). To demonstrate the quality of this algorithm in practice, the UAM network can be reduced by following the generic city model presented in [56]. In this generic model, seven vertiports are configured in a hexagonal pattern of "six around one". As shown in Fig. 4, one vertiport is at the center of the hexagon and located equidistant from the other six vertiports at a distance of $16\,\text{km}$. The cruise speed of every aircraft in the system is $190\,\text{km=h}$ [40]. The near mid-air conflict (NMAC) is defined to be $500\,\text{ft}$ [13]. The discretized time step for an aircraft to move forward is $2\,\text{s}$. The covariance of the location uncertainty of an aircraft is $= [0{:}04; 0;\ 0; 0{:}01]$. In Eq. (23), is the mean value returned from the Gaussian process, and is the exploration coefficient which decides whether we should put on more weights on the actions whose values we already know (exploitation), or the actions we haven't explored (exploration). In this paper, is set to 1. In Eq. (26), we actually didn't use B and (they are simply set to 1), and we list them in this paper just to make the formulation extendable. In this section, all the tests were implemented in Python 3.8 and were run on an Intel Xeon Silver 4210 CPU 2.20GHz desktop with 32GB RAM.

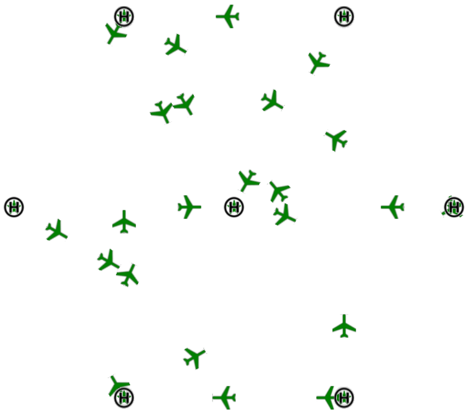


Fig. 4. Seven vertiports overlaid with segment length of $16\,\text{km}$

B. Solution Quality of MCTS GP Compared with Another Three Algorithms: MCTS Uniform, MCTS Discrete and ORCA

Based on the above test settings, we conduct the following several tests to demonstrate the performance of our proposed algorithm. The proposed algorithm we develop in this paper is based on continuous action space and uses Gaussian process regression to discretize the continuous action space (named as **MCTS GP**). In addition, LOCCS formulation is employed to perform LOS event checking. To show the solution quality of our proposed algorithm, we compare it with another three algorithms, which are:

1) MCTS algorithm based on discrete action space (named as **MCTS Discrete**). It has three discrete options of control actions: left, right, and straight. It uses Monte Carlo Simulation to perform LOS event checking instead of LOCCS formulation we established in this paper. Further details can be found in [40];
2) MCTS algorithm based on continuous action space using uniform sampling (named as **MCTS Uniform**). For MCTS Uniform, it uses uniform sampling to discretize the continuous action space. That is, the heading angle is chosen uniformly from the continuous action space;
3) Optimal Reciprocal Collision Avoidance Algorithm (**ORCA**) [13], [57].

For MCTS GP, its parameters are set to: confidence level 90%, search depth 2, and the number of expanded nodes 5. For MCTS Uniform, its parameter settings are the same as MCTS GP. It is worth noting that since the number of the expanded nodes for MCTS GP is the same as MCTS Uniform, the fineness of the discretization of MCTS GP and MCTS Uniform is considered the same. For MCTS Discrete, it doesn't take confidence level into account since it uses Monte Carlo simulation to perform LOS event checking in lieu of LOCCS formulation proposed in this paper; also, it only considers three options of heading angles when taking control actions (left, right, straight), which means the number of expanded nodes are set to 3; its remaining parameter setting of search depth is the same as MCTS GP. ORCA Algorithm serves as a baseline. For all four algorithms, the number of aircraft varies from 10 to 80 with an incremental step of 10.

The solution quality of all four different algorithms is compared. Fig. 5 shows the solution quality of aforementioned algorithms. All these algorithms perform similarly. As the number of aircraft increases, the probability of NMAC and running time both increase while probability of reaching the goal state decreases. For all the algorithms, the aircraft can reach the goal state with the probability beyond 90%, and the NMAC probability is under 10%, which means that all the algorithms are effective when it comes to potential conflicts avoidance. Fig. 5a shows that MCTS GP performs better than MCTS Uniform, MCTS Discrete, and ORCA in terms of goal probability. Thus, MCTS GP has the lowest NMAC probability, as shown in Fig. 5b. In addition, Fig. 5c shows that MCTS GP spends limited time while achieving high goal probability and high NMAC probability owing to the efficiency of LOCCS formulation when it comes to LOS event checking.

MCTS GP applies to a multi-agent system of UAM which has its own constraints and concerns like traffic density and dynamic complexity. Even when the number of agents is very large, the computational time is still acceptable for UAM applications. We can also adjust the parameters to obtain shorter computational time. This ensures that MCTS GP can run in online fashion. Although the computational time of MCTS GP is slightly longer than MCTS Discrete or ORCA since Gaussian process regression requires a bit more computational cost, the difference in terms of computational time is not significant. In contrast, the computational time of MCTS Discrete, which uses Monte Carlo Simulation instead to perform LOS event checking, is much longer than using LOCCS formulation.

C. Parameter Sensitivity Analysis of MCTS Algorithm with Continuous Action Space (MCTS GP)

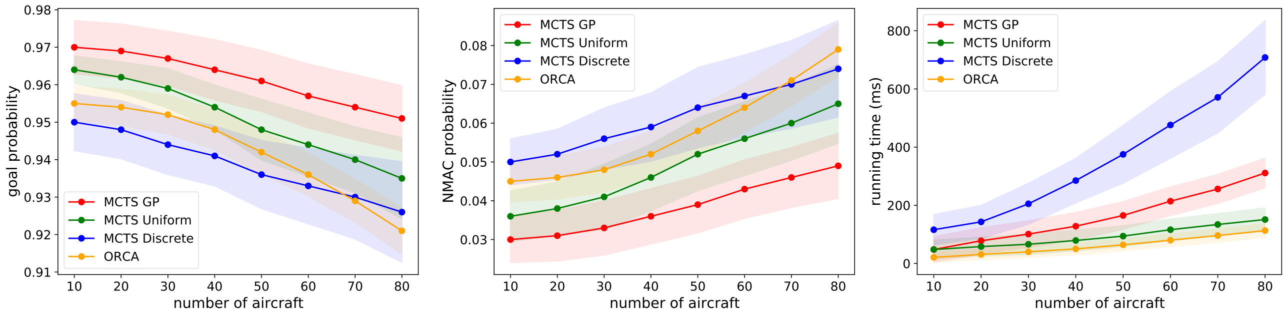
There are mainly three parameters impacting the performance of the proposed algorithm MCTS GP: the confidence level, the search depth, and the number of expanded nodes. The confidence level reflects the level of location uncertainty imposed on the aircraft. A higher confidence level indicates more uncertainty. Search depth indicates the number of steps to look ahead. Deeper search depth means more intensive exploitation of the tree search. For more detailed illustration, please refer to [13]. Typically for MCTS GP, a deeper search tree can give rise to better quality of the algorithm but need more expensive computational cost. The number of expanded nodes decides how many child nodes a parent node in the search tree should have when it is fully expanded. A larger number of expanded nodes reflects more extensive exploration of the tree search. In this subsection, the baseline is set to: confidence level 90%, search depth 2, and the number of expanded nodes 5. The number of aircraft varies from 10 to 80 with an incremental step of 10.

To explore the impact of these three different parameters on the performance of MCTS GP, we conduct sensitivity analysis for these three different parameters respectively.

1) Results of Varying Confidence Levels: In this simulation, the parameter of confidence level varies while the remaining two parameters are the same as the baseline. The parameter of confidence level is chosen from 90%, 95% through 97%.

As shown in Fig. 6, the performance trends of MCTS GP with respect to varying aircraft numbers are consistent for three different confidence levels above. Fig. 6a shows that the higher confidence level, the better goal probability. Fig. 6b shows that the highest confidence level owns the lowest NMAC probability. Fig. 6c shows that the performance of goal probability and NMAC probability improves at the cost of longer running time. This is because when the confidence level increases, the risk domain becomes larger and the algorithm needs to perform more iterations to find feasible path without violating the risk domain. When the confidence level is too high, we may not find a feasible solution to the trajectory planning problem in real time.

2) Results of Varying Search Depth: In this simulation, we discuss the performance of MCTS GP with respect to three

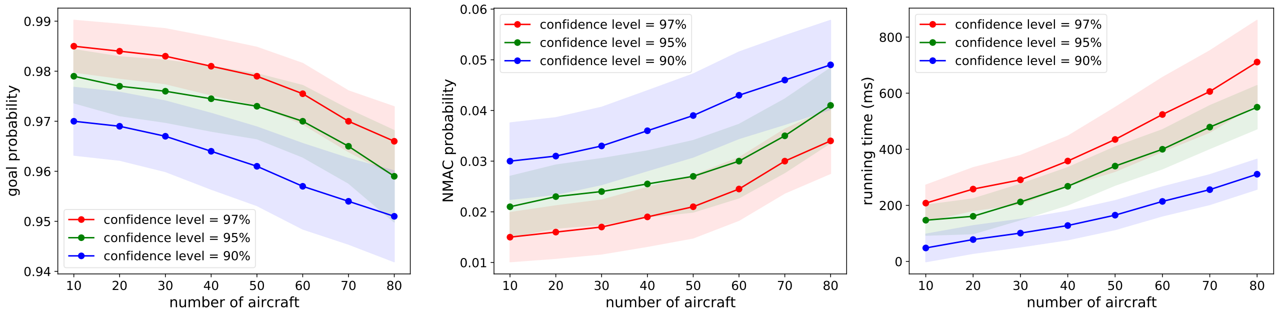


(a) goal probability

(b) NMAC probability

(c) running time

Fig. 5. Solution quality of different algorithms: MCTS GP, MCTS Discrete, MCTS Uniform, ORCA



(a) goal probability

(b) NMAC probability

(c) running time

Fig. 6. Solution quality of MCTS GP under different confidence levels

different search depth 2, 3 and 4. The remaining parameter settings are the same as the baseline.

From Fig. 7a and Fig. 7b we can observe that MCTS GP with search depth 4 owns the highest goal probability, and therefore owns the lowest NMAC probability. Fig. 7c indicates that as the number of aircraft increases, the running time grows and the improved performance of MCTS GP with search depth 4 comes at the cost of longer computational time.

3) Results of Varying Expanded Nodes: In this simulation, we investigate the performance of MCTS GP with respect to the number of expanded nodes 5, 10 and 20. The other two parameters are set to be the same as the baseline.

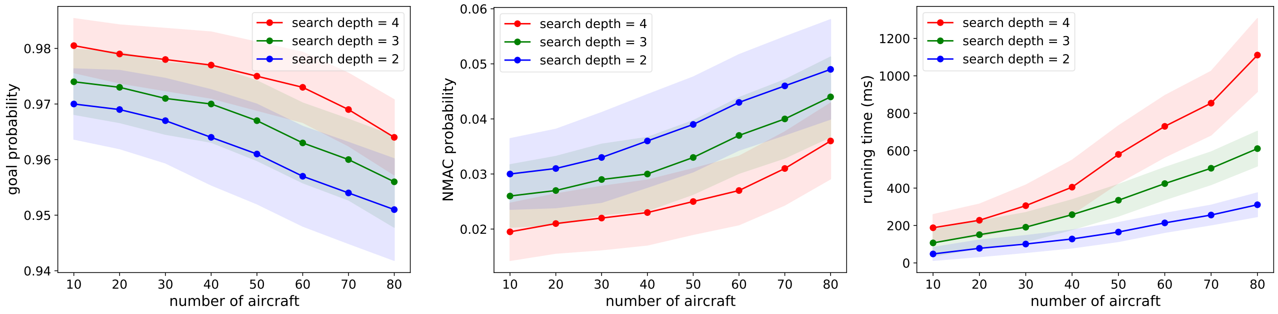
From Fig. 8a and Fig. 8b we find that the goal probability increases (NMAC probability decreases) as the number of expanded nodes increases. Fig. 8c shows that compared with expanded nodes of 5 and 10, the running time of MCTS GP with expanded nodes 20 grows largely while there is little improvement in terms of goal probability and NMAC probability.

Above all, this proposed algorithm will only get a sub-optimal solution because MCTS is not fully completed in every iteration due to the computational time budget. It sacrifices some optimality to achieve online operation in reality. The simulation results show that our approach still works well even it is not true optimal. In real applications, MCTS GP algorithm, serving as a higher level design, will work together with a lower level design of airborne collision avoidance system (ACAS) [58] to guarantee high aviation safety. Also, it can help us decide the density of aircraft. The setting of different parameters in this part proposes to demonstrate that there is a trade-off between safety performance (goal probability) and computational cost (average running time). We leave it to the users, like Air Traffic Controller (ATC), to make their own choices of parameter setting. If ATC is more concerned with safety, we need to sacrifice the running time, and vice versa.

D. Average Flight Time Comparisons: MCTS GP, MCTS Uniform and MCTS Discrete

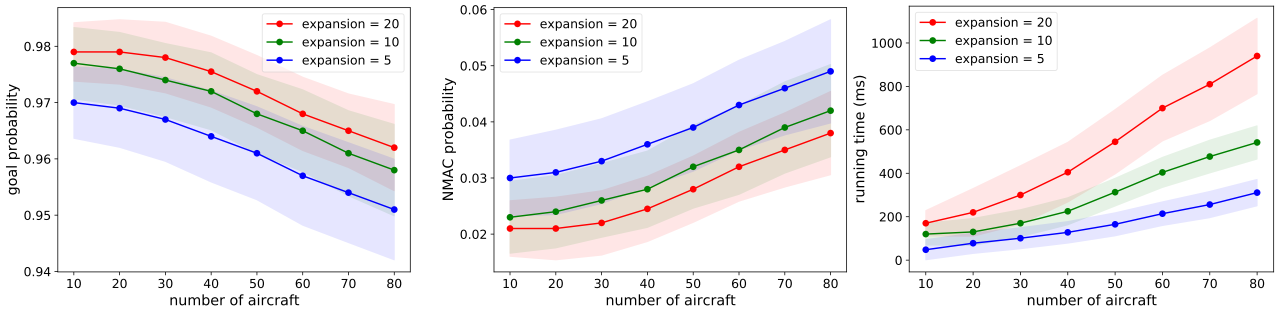
In Fig. 9 we draw the identified trajectories by following the actions of MCTS GP and MCTS Discrete when there is no other aircraft blocking the way to the destination. We can find that compared with MCTS Discrete, aircraft utilizing MCTS GP can fly to the destination more smoothly.

In view of the fact that without intruder aircraft MCTS GP can help the aircraft fly more smoothly, we expect that the performance of flight time can be also improved when it comes to the scenarios where intruder aircraft exist. We study the total flight time for an aircraft from the start vertiport to the target one, with respect to different routes and different air traffic densities. Here we classify the route options into three types according to the distance between the start vertiport and the target one. In Fig. 10 we tag the vertiports from Fig. 4 with different ID numbers. With this vertiport setting, Route 1 (from vertiport 1 to 2) is the route of the shortest length, Route 2 (from vertiport 3 to 7) is of the medium length, and Route 3 (from vertiport 3 to 6) is of the longest length. The three different routes are displayed in Fig. 10. We compare the flight time for different route options with respect to different air



(a) goal probability (b) NMAC probability (c) running time

Fig. 7. Solution quality of MCTS GP under different search depths



(a) goal probability (b) NMAC probability (c) running time

Fig. 8. Solution quality of MCTS GP under expanded nodes

traffic densities using MCTS GP, MCTS Uniform and MCTS Discrete separately, which is recorded in Table I.

From this table, we can find that the flight time ratio of MCTS GP to MCTS Discrete ranges from 85.4% to 90.9% and the ratio of MCTS GP to MCTS Uniform ranges from 91.6% to 95.8%, which indicates that given any route and any air traffic density, the flight time of aircraft using MCTS GP is always shorter than that using MCTS Discrete or MCTS Uniform. This is because compared with MCTS Discrete or MCTS Uniform, MCTS GP employs Gaussian process regression to guide the sampling of actions from the continuous action space, which helps the aircraft move more smoothly to its destination.

We can also observe that for Route 1, the flight time ratio of MCTS GP to MCTS Discrete has decreased from 90.9% to 88.7% and the flight time ratio of MCTS GP to MCTS Uniform is decreased from 95.8% to 94.4%, as traffic density increases. Similar trends of flight time ratio also apply to Route 2 and Route 3. This means that as the traffic density increases, the flight time difference between MCTS GP and MCTS Discrete or MCTS Uniform becomes more significant. Simultaneously, when the traffic density is fixed, for instance, to be low level, the flight time ratio of MCTS GP to MCTS Discrete varies from 90.9% to 88.9% and the flight time ratio of MCTS GP to MCTS Uniform varies from 95.8% to 94.5% for different routes. Similar manners also apply to middle level traffic density and high level traffic density, which suggests that as the length of route increases, the flight time of aircraft using MCTS GP will be increasingly shorter than that using MCTS Discrete or MCTS Uniform. This demonstrates that MCTS GP outperforms MCTS Discrete or MCTS Uniform in terms of flight time, since Gaussian process regression is a guided search.

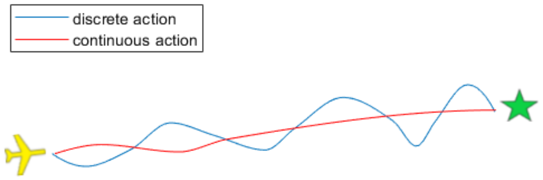


Fig. 9. Trajectories generated by MCTS GP and MCTS Discrete

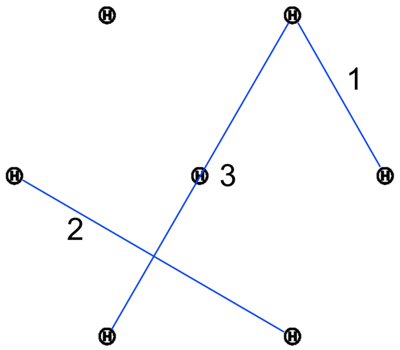


Fig. 10. Vertiport setting with three different routes

VI. Conclusions

In this paper, we come up with a decentralized online computational guidance algorithm with the capability of separation assurance for a multi-agent system. The MCTS algorithm proposed in this paper employs LOCCS formulation to perform LOS event checking. Numerical study illustrates the introduction of LOCCS formulation helps the MCTS algorithm run much faster than employing Monte Carlo Simulation. This ensures the proposed MCTS algorithm in this paper can run in real time while assuring flying safety, which is a great advantage especially when it comes to the applications of UAM. Also, the MCTS algorithm proposed in this paper uses Gaussian process regression to discretize the continuous action space. Numerical study indicates the flight time of MCTS using Gaussian process regression is significantly shorter than using uniform sampling, at the cost of a bit more computational time, which justifies the benefit of the use of Gaussian process regression.

In the future, we will take into account the non-cooperative intruder aircraft which cannot be controlled by our computational guidance algorithm. We will also consider the scenarios where the communication network fails. The fairness between the flights is another important issue which remains to be explored. In addition, we are working on the location uncertainty obeying non-Gaussian distribution through Kernel Density Estimation and Mixed Integer Linear Programming.

Acknowledgment

This work was supported by the National Science Foundation under Grants CMMI-2138612.

References

[1] L. Gipson, “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] J. Holden and N. Goel, “Fast-forwarding to a future of on-demand urban air transportation,” *San Francisco, CA*, 2016.

[3] N. Polaczyk, E. Trombino, P. Wei, and M. Mitici, “A review of current technology and research in urban on-demand air mobility applications,” in *8th Biennial Autonomous VTOL Technical Meeting & 6th Annual Electric VTOL Symposium*, January 2019.

[4] E. R. Mueller, P. H. Kopardekar, and K. H. Goodrich, “Enabling airspace integration for high-density on-demand mobility operations,” in *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, p. 3086.

[5] G. Zhu and P. Wei, “Pre-departure planning for urban air mobility flights with dynamic airspace reservation,” in *AIAA Aviation 2019 Forum*, 2019, p. 3519.

[6] J. M. Hoekstra, R. N. van Gent, and R. C. Ruigrok, “Designing for safety: the ‘free flight’ air traffic management concept,” *Reliability Engineering & System Safety*, vol. 75, no. 2, pp. 215–232, 2002.

[7] H. A. Blom and G. Bakker, “Safety evaluation of advanced self-separation under very high en route traffic demand,” *Journal of Aerospace Information Systems*, vol. 12, no. 6, pp. 413–427, 2015.

[8] S. Kahne and I. Frolow, “Air traffic management: Evolution with technology,” *IEEE Control Systems*, vol. 16, no. 4, pp. 12–21, 1996.

[9] D. H. Shim and S. Sastry, “An evasive maneuvering algorithm for uavs in see-and-avoid situations,” in *American Control Conference, 2007. ACC’07*. IEEE, 2007, pp. 3886–3891.

[10] Z. Huang, Y. Lu, H. Wen, and D. Jin, “Ground-based experiment of capturing space debris based on artificial potential field,” *Acta Astronautica*, vol. 152, pp. 235–241, 2018.

[11] R. Chai, A. Tsourdos, A. Savvaris, Y. Xia, and S. Chai, “Real-time reentry trajectory planning of hypersonic vehicles: A two-step strategy incorporating fuzzy multiobjective transcription and deep neural network,” *IEEE Transactions on Industrial Electronics*, vol. 67, no. 8, pp. 6904–6915, 2019.

[12] T. B. Wolf and M. J. Kochenderfer, “Aircraft collision avoidance using monte carlo real-time belief space search,” *Journal of Intelligent & Robotic Systems*, vol. 64, no. 2, pp. 277–298, 2011.

[13] X. Yang and P. Wei, “Autonomous free flight operations in urban air mobility with computational guidance and collision avoidance,” *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[14] E. Frazzoli, Z.-H. Mao, J.-H. Oh, and E. Feron, “Resolution of conflicts involving many aircraft via semidefinite programming,” *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 1, pp. 79–86, 2001.

[15] L. Pallottino, E. M. Feron, and A. Bicchi, “Conflict resolution problems for air traffic management systems solved with mixed integer programming,” *IEEE transactions on intelligent transportation systems*, vol. 3, no. 1, pp. 3–11, 2002.

[16] Z. Zhou, J. Chen, and Y. Liu, “Optimized landing of drones in the context of congested air traffic and limited vertiports,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 9, pp. 6007–6017, 2021.

[17] A. U. Raghunathan, V. Gopal, D. Subramanian, L. T. Biegler, and T. Samad, “Dynamic optimization strategies for three-dimensional conflict resolution of multiple aircraft,” *Journal of guidance, control, and dynamics*, vol. 27, no. 4, pp. 586–594, 2004.

[18] P. Wu, H. Wen, T. Chen, and D. Jin, “Model predictive control of rigid spacecraft with two variable speed control moment gyroscopes,” *Applied Mathematics and Mechanics*, vol. 38, no. 11, pp. 1551–1564, 2017.

[19] D. Mellinger, A. Kushleyev, and V. Kumar, “Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 477–483.

[20] B. Acikmese and S. R. Ploen, “Convex programming approach to powered descent guidance for mars landing,” *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 5, pp. 1353–1366, 2007.

[21] R. Chai, A. Savvaris, A. Tsourdos, Y. Xia, and S. Chai, “Solving multi-objective constrained trajectory optimization problem by an extended evolutionary algorithm,” *IEEE transactions on cybernetics*, vol. 50, no. 4, pp. 1630–1643, 2018.

[22] M. Pontani and B. A. Conway, “Particle swarm optimization applied to space trajectories,” *Journal of Guidance, Control, and Dynamics*, vol. 33, no. 5, pp. 1429–1441, 2010.

[23] R. Chai, A. Savvaris, A. Tsourdos, S. Chai, and Y. Xia, “Unified multiobjective optimization scheme for aeroassisted vehicle trajectory planning,” *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 7, pp. 1521–1530, 2018.

[24] D. González, J. Pérez, V. Milanés, and F. Nashashibi, “A review of motion planning techniques for automated vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1135–1145, 2015.

[25] Y. Lin and S. Saripalli, “Sampling-based path planning for UAV collision avoidance,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 11, pp. 3179–3192, 2017.

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

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

[28] S. Li, M. Egorov, and M. Kochenderfer, “Optimizing collision avoidance in dense airspace using deep reinforcement learning,” *arXiv preprint arXiv:1912.10146*, 2019.

[29] V. R. Desaraju and J. P. How, “Decentralized path planning for multi-agent teams in complex environments using rapidly-exploring random trees,” in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 4956–4961.

[30] T. Schouwenaars, J. How, and E. Feron, “Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, p. 5141.

[31] L. Cheng, H. Wen, and D. Jin, “Uncertain parameters analysis of powered-descent guidance based on chebyshev interval method,” *Acta Astronautica*, vol. 162, pp. 581–588, 2019.

[32] J. P. Hespanha, H. H. Kizilocak, and Y. S. Ateskan, “Probabilistic map building for aircraft-tracking radars,” in *Proceedings of the 2001 American Control Conference.(Cat. No. 01CH37148)*, vol. 6. IEEE, 2001, pp. 4381–4386.

TABLE I
AVG. FLIGHT TIME COMPARISONS IN SECONDS

Traffic Density	Algorithm	Route 1	Route 2	Route 3
Low (10 aircraft)	MCTS Discrete	342.6 ± 7.5	610.1 ± 9.8	698.9 ± 10.9
	MCTS Uniform	323.6 ± 7.1	572.7 ± 7.9	662.5 ± 8.5
	MCTS GP	309.9 ± 6.9	545.8 ± 7.3	621.3 ± 7.7
	MCTS GP/MCTS Uniform	95.8%	95.3%	94.5%
	MCTS GP/MCTS Discrete	90.9%	89.6%	88.9%
Middle (20 aircraft)	MCTS Discrete	482.4 ± 8.4	862.0 ± 11.2	988.3 ± 12.7
	MCTS Uniform	458.2 ± 8.0	812.3 ± 10.8	934.7 ± 12.1
	MCTS GP	436.5 ± 7.9	769.2 ± 10.6	875.4 ± 11.5
	MCTS GP/MCTS Uniform	95.2%	94.7%	93.6%
	MCTS GP/MCTS Discrete	90.5%	89.2%	88.6%
High (80 aircraft)	MCTS Discrete	1344.2 ± 13.1	2454.6 ± 16.7	2836.7 ± 18.5
	MCTS Uniform	1263.2 ± 12.6	2273.4 ± 15.8	2643.9 ± 18.0
	MCTS GP	1192.5 ± 12.3	2132.5 ± 15.4	2421.8 ± 17.6
	MCTS GP/MCTS Uniform	94.4%	93.8%	91.6%
	MCTS GP/MCTS Discrete	88.7%	86.9%	85.4%

[33] X. Yang, L. Deng, J. Liu, P. Wei, and H. Li, "Multi-agent autonomous operations in urban air mobility with communication constraints," in *AIAA Scitech 2020 Forum*, 2020, p. 1839.

[34] L. Blackmore, M. Ono, and B. C. Williams, "Chance-constrained optimal path planning with obstacles," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1080–1094, 2011.

[35] B. Du, J. Chen, D. Sun, S. G. Manyam, and D. W. Casbeer, "Uav trajectory planning with probabilistic geo-fence via iterative chance-constrained optimization," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[36] R. Chai, A. Tsourdos, A. Savvaris, S. Wang, Y. Xia, and S. Chai, "Fast generation of chance-constrained flight trajectory for unmanned vehicles," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 2, pp. 1028–1045, 2020.

[37] B. Luders, M. Kothari, and J. How, "Chance constrained rrt for probabilistic robustness to environmental uncertainty," in *AIAA guidance, navigation, and control conference*, 2010, p. 8160.

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

[39] E. A. Feinberg and A. Shwartz, *Handbook of Markov decision processes: methods and applications*. Springer Science & Business Media, 2012, vol. 40.

[40] X. Yang and P. Wei, "Scalable multi-agent computational guidance with separation assurance for autonomous urban air mobility," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 8, pp. 1473–1486, 2020.

[41] Y. Liu, "A progressive motion-planning algorithm and traffic flow analysis for high-density 2d traffic," *Transportation Science*, vol. 53, no. 6, pp. 1501–1525, 2019.

[42] P. Wu, L. Li, J. Xie, and J. Chen, "Probabilistically guaranteed path planning for safe urban air mobility using chance constrained rrt*," in *AIAA AVIATION 2020 FORUM*, 2020, p. 2914.

[43] P. Wu, J. Xie, and J. Chen, "Safe path planning for unmanned aerial vehicle under location uncertainty," in *2020 IEEE 16th International Conference on Control & Automation (ICCA)*. IEEE, 2020, pp. 342–347.

[44] T. A. Snijders, *Multilevel analysis*. Springer, 2011.

[45] V. A. Zorich, *Mathematical analysis II*. Springer, 2016.

[46] J. Mockus, *Bayesian approach to global optimization: theory and applications*. Springer Science & Business Media, 2012, vol. 37.

[47] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*. Springer, 2003, pp. 63–71.

[48] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," *arXiv preprint arXiv:0912.3995*, 2009.

[49] C. B. Browne, E. Powley, D. Whitehouse, S. M. Lucas, P. I. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A survey of monte carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in games*, vol. 4, no. 1, pp. 1–43, 2012.

[50] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

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

[52] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.

[53] T. Yee, V. Lisỳ, M. H. Bowling, and S. Kambhampati, "Monte carlo tree search in continuous action spaces with execution uncertainty." in *IJCAI*, 2016, pp. 690–697.

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

[55] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine learning*, vol. 47, no. 2-3, pp. 235–256, 2002.

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

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

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

Pengcheng Wu received both Master and Bachelor degree from Department of Aerospace Engineering, Nanjing University of Aeronautics and Astronautics. He is now a joint Ph.D. student with Department of Mechanical and Aerospace Engineering, University of California San Diego, and with Department of Aerospace Engineering, San Diego State University. He has extensive research interests in dynamics, guidance and control of unmanned vehicles. Currently, he is working on the path planning and control of multi-agent systems under uncertainty.