Decentralized Multi-Agent Motion Planning Using Cognitive Hierarchy and Gaussian Process Classification

Josh Netter[1], *Student Member, IEEE*, George P. Kontoudis[2], *Member, IEEE*, Kyriakos G. Vamvoudakis[1], *Senior Member, IEEE*

***Abstract*—In this paper, we present a motion planning algorithm designed to guide agents, termed as player agents, optimally through multi-agent 3D urban air environments. The method integrates a sampling-based path planner, model-free optimal control, and a cognitive hierarchy model to predict the motion of other agents. Each player constructs a path through the environment, which is dynamically re-planned as the obstacle space of the environment evolves based on its online observations and the observations of cooperating players. The cognitive hierarchy model predicts the behavior of each agent in the environment, while a Gaussian process classification method estimates an unknown agent's level of rationality in real-time by observing each agent's kinodynamic distance. Once another agent's motion planning strategy is inferred, the player agents construct a predicted obstacle space based on each agent's expected motion to avoid potential collisions. Each player then traverses its planned path using a Q-learning controller. We validate the effectiveness of the proposed method in numerical experiments of a 3D urban air environment containing four and ten agents. We demonstrate that this approach is effective for reducing distance traveled by agents to reach their goals, mitigating the risk of collisions, and preventing deadlocks.**

***Index Terms*—Autonomous Vehicles, Control of dynamic systems, Machine Learning, Motion Planning**

I. Introduction

During the past two decades, significant advancements have been made in the flight of unmanned aerial vehicles (UAVs). These UAVs have a wide variety of uses, such as package delivery, search and rescue, aerial surveying and videography, and power line inspection among many others that cannot easily be accomplished on the ground or by larger, less maneuverable air vehicles. As a result, more consideration has been given on how to use a multitude of UAVs simultaneously in an urban air environment, a problem referred to as urban air mobility (UAM) [1]–[5]. UAM has emerged as a leading challenge for these drones in recent years, with many difficulties to overcome. For example, obstacle avoidance in real time and safe motion planning are still considered necessary developments for robotic motion planning and face numerous challenges [6]. In real-world environments, motion planning is more complex than accurately observing obstacles between a start and goal location, and then planning a path around them. Instead, robots may need to overcome various difficulties, such as dynamic environments where the obstacle space changes over time and a noisy perception system. To safely plan a path through dynamic environments, these vehicles must not only consider the presently observed obstacle space, but also consider future collisions by preemptively avoiding potential future and adversarial obstacles. This challenge is exacerbated by the simultaneous presence of multiple agents in the same environment, where other agents with unknown destinations and unknown paths may interfere with an autonomous robot's planning. Additionally, these paths may also need to account for *kinodynamic* constraints on the robot, which influences how the robot's motion based on physical constraints such as maximum velocity and acceleration [7]. In this work, we seek a scalable multi-agent motion planning method of decentralized communication networks in dynamic environments. We consider each previously outlined component of this problem and address them individually.

[1]J. Netter and K. G. Vamvoudakis are with the Daniel Guggenheim School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, Georgia, USA. Email: {jnetter6,kyriakos@gatech.edu}.

[2]G. P. Kontoudis is with the Department of Mechanical Engineering, Colorado School of Mines, Golden, Colorado, USA. Email: george.kontoudis@mines.edu.

This work was supported in part, by NSF under grant Nos. CPS-2038589, CPS-2227185, SATC-2231651, and SLES-2415479 and by NASA ULI under grant Nos. 80NSSC20M0161 and 80NSSC25M7104.

Related Work

We begin by considering previous work in motion planning in dynamic environments. RRT$^{\mathrm{X}}$ is introduced in [8] and provides rapid re-planning in dynamic environments for real-time implementation. RRT$^{\mathrm{X}}$ is also used as the path planning basis for RRT-Q$^{\mathrm{X}}$ [9], [10], which additionally includes a model-free method of finding the optimal control to traverse the planned path. This model-free control technique is originally introduced in [11]. The UAM problem is also considered an instance of a kinodynamic motion problem, detailed in [7]. Some works, such as [12], present a method to solve for optimal control in kinodynamic motion problems, but assume that the system dynamics of the agent navigating the environment are known. To navigate through a kinodynamic environment with unknown system dynamics, [13] presents an alternative solution making use of the *kinodynamic distance (KD)*, a concept defined as the distance between an agent and its planned path.

Multi-agent environments, such as an urban airspace, introduce additional challenges in avoiding the motion of other agents. Many approaches have been taken to navigate through these environments. For example, in [14], the authors introduce a UAM motion planning approach that accounts for multiple potential behaviors for drones in the obstacle space. However, this work plans paths offline and only makes minimal adjustments to avoid other agents. A similar multi-agent motion planning approach for nonlinear dynamics is shown in [15], but only with cooperative agents.

We also consider scenarios where an agent does not have perfect knowledge of the strategies of other agents in the environment, a concept referred to as *bounded rationality* [16]. In these cases, agents instead can find more efficient paths by modelling the behavior of other agents and using their predicted future states to plan their own path. This method is often based on specifically human-robot interaction, such as [17]. Another approach in [18] uses interacting Gaussian process regression to model human motion and considers the motion of multiple other agents operating in the same space. This work also uses Gaussian processes, but instead to classify motion strategies rather than form a model of motion with Gaussian process regression. An alternative approach to multi-agent motion planning using both reinforcement learning and force-based motion planning (FMP) to avoid collisions is shown in [19].

In this work, we must also determine the potential strategies of other autonomous agents rather than predicting human motion. One method of determining these strategies is by forming a *cognitive hierarchy* [20] to describe the levels of rationality of multiple players in a game. This approach is used to determine the behavior of cyber-physical systems such as drones in [21], [22]. In autonomous motion planning, cognitive hierarchy can be used to identify potential motion planning strategies used by individual agents, allowing other agents to plan their motion in response to these strategies to avoid collisions. For example, this is seen in a racing environment in [23]. Furthermore, a cognitive hierarchy specific to urban air mobility is introduced in [24], as well as a method to estimate the level of rationality expressed by individual agents.

Contribution: The contribution of this paper is three-fold:

- We formulate a decentralized motion planning method with a bounded rationality approach to navigate environments containing both communicating, cooperative agents as well as independent agents.
- We propose a method to mitigate sensor noise in obstacle detection to ensure more accurate observation of the obstacle space as well as observation of independent agent motion.
- We synthesize Distributed RRT-Q$^{\mathrm{X}}$, a robust method of optimal motion planning in a 3D urban air environment for multi-agent settings.

Previous work and results related to this paper are found in [24], [25]. This paper expands upon these results by first combining these previous works into an algorithm that considers both cooperative and non-cooperative agents, known as player agents and independent agents respectively, in the environment. We additionally use observations from player agents to propose a more sophisticated method of identifying the levels of rationality of independent agents using distributed Gaussian process classification. Next, we incorporate distance-varying noise in observations of the obstacle space to make our algorithm compatible to heterogeneous sensing capabilities.

Structure: The remainder of the present paper is structured as follows. In Section II we formulate the urban air mobility problem, including a brief overview of previous work. In Section III, we describe a decentralized motion planning method for communicating player agents to avoid collisions with one another. Section IV details the cognitive hierarchy of unknown agents in the environment, and Section V discusses how we predict an agent's level of rationality. In Section VI, we present how we use noisy observations of the environment to estimate the obstacle space. Section VII details the combined motion planning framework. Section VIII presents simulations using this framework and its results, and Section IX concludes the paper and discusses future work.

Assumptions: To derive our key results, we assume that the environment contains some number of cooperative agents capable of observing all obstacles in a neighborhood around themselves, and that these agents can communicate these observations to each other. We also assume that non-communicating agents in the environment are not adversarial (eg. seeking collisions with other agents) and not attempting to deceive our method of learning their motion planning strategies. Additionally, for the model-free control, we approximate vehicles as linear time-invariant systems.

II. Problem Formulation

Let us consider an environment $\mathcal{X}$ containing S dynamic agents. The location of each agent i in the environment at a time t is defined as their current state $x_i(t)$, where $i = 1, \ldots, S$. Additionally, each agent is assigned an initial state $x_{i,0}$ and a goal state $x_{i,\mathrm{g}}$. We consider two kinds of agents which may be present in the environment: *player agents*, which are agents we control and are capable of communicating with one another, and *independent agents*, which we do not control and cannot communicate with. For each player agent i we also define an obstacle space $\mathcal{X}_{\mathrm{obs},i}(t) \subset \mathcal{X}$. We seek to generate a high-level path to guide each player agent i through the environment $\mathcal{X}$ from $x_{i,0}$ to $x_{i,\mathrm{g}}$ with no collision, where a collision is defined as $x_i(t) \in \mathcal{X}_{\mathrm{obs},i}(t)$ for any time $t \geqslant 0$.

To avoid collision with obstacles and reach their goal state, each player agent implements RRT$^{\mathrm{X}}$ [8] to construct a graph $\mathcal{G}_i = (V_i, E_i)$, where V_i is the set of nodes and E_i is the set of edges. The edges of this graph are used in turn to construct a collision-free path $\pi_i(x_{i,0}, x_{i,\mathrm{g}}; t) \in \mathbb{R}^{2(K_i \times n)}$ for $k = 1, ..., K_i$ of two-point boundary value problems (BVPs), where $K_i \in \mathbb{N}$ is the number of BVPs in its path. These series of BVPs connect the player agent's initial state to its final goal state, and each BVP k is represented by a tuple of points $< x_{i,0,k}, x_{i,\mathrm{g},k} >$, where $x_{i,0,k}$ and $x_{i,\mathrm{g},k}$ are the initial and goal states of the BVP, respectively. We choose this algorithm as the obstacle space of our environment may change over time and RRT$^{\mathrm{X}}$ can quickly refine its graph $\mathcal{G}_i$ in real-time and generate new waypoints of collision-free paths. Additionally, it is shown to be asymptotically globally optimal.

To traverse the path, we define a distance function between two states x_1, x_2 as

$$D(x_1, x_2) := ||x_1 - x_2||. \quad (1)$$

The agent seeks to decrease $D(x_i(t), x_{i,\mathrm{g},k})$ until $D(x_i(t), x_{i,\mathrm{g},k}) < \rho D(x_{i,0,k}, x_{i,\mathrm{g},k})$ where $\rho < 1$ is a user-defined constant. Upon reaching this neighborhood, the agent defines its location as $x_{i,0,k+1}$ and begins to travel to the next goal state $x_{i,\mathrm{g},k+1}$ in accordance with the planned path.

Next, we consider the optimal control problem for each player agent to reach the goal state with completely unknown system dynamics. Each agent i can be described as a linear time-invariant system,

$$\dot{x}_i(t) = Ax_i(t) + Bu_i(t), \quad x_i(0) = x_{i,0},\, t \geqslant 0 \tag{2}$$

where $x_i(t) \in \mathcal{X} \subseteq \mathbb{R}^n$ and $u_i(t) \in \mathcal{U} \subseteq \mathbb{R}^m$ are the state and control input vectors, respectively, for agent $i = 1, \ldots, S$, and $A \in \mathbb{R}^{n \times n}$ and $B \in \mathbb{R}^{n \times m}$ as the respective plant and input matrices. We alternatively define a "goal difference" state $\bar{x}_i(t) = D(x_i(t), x_{i,\mathrm{g}})$, where $x_{i,\mathrm{g}}$ is agent i's current goal state, and additionally the system dynamics,

$$\dot{\bar{x}}_i(t) = A\bar{x}_i(t) + Bu_i(t), \quad t \geqslant 0. \tag{3}$$

As the origin of the environment is arbitrary, this transformation still describes an identical system to Equation (2). We employ a finite horizon linear-quadratic regulator to drive the player agent to its goal state at a finite horizon time T. Let us consider for each agent i in the environment a minimization of the finite-horizon cost function,

$$J_i(\bar{x}_{i,0}, u_i; t_0, T) = \phi_i(T) + \frac{1}{2}\int_{t_0}^{T} \bar{x}_i^{\intercal} M \bar{x}_i + u_i^{\intercal} R u_i \,\mathrm{d}\tau, \tag{4}$$

where $\phi_i(T) := (1/2)\bar{x}_i^{\intercal}(T)P_i(T)\bar{x}_i(T)$ is the terminal cost, $P_i(T) \in \mathbb{R}^{n \times n} \succ 0$ is the final Riccati matrix, and $M \in \mathbb{R}^{n \times n} \succeq 0$, $R \in \mathbb{R}^{m \times m} \succ 0$ are penalty matrices for the goal difference state and control input, respectively.

Assumption 1. The matrix pair (A, B) is controllable and the matrix pair ($M^{1/2}$, A) is detectable for all values i. □

To minimize the cost function, each player agent seeks an optimal control $u_i^{\star}(\bar{x}, t)$ such that $J_i(\bar{x}_{i,0}; u_i^{\star}; t_0, T) \leqslant J_i(\bar{x}_{i,0}; u_i; t_0, T), \forall \bar{x}_i$. We alternatively express this problem as the value function,

$$V_i(\bar{x}_i) = \min_{u_i \in \mathcal{U}_i} \left\{ \phi_i(T) + \frac{1}{2}\int_{t_0}^{T} \bar{x}_i^{\intercal} M \bar{x}_i + u_i^{\intercal} R u_i \,\mathrm{d}\tau \right\}. \tag{5}$$

We find this optimal control using a model-free actor-critic structure, as seen in previous work [26].

To mitigate the risk of collisions due to kinodynamic constraints and optimal performance constraints, we construct a "buffer" in between the obstacle space and the free space. More formally, let us define the kinodynamic distance (KD) of each agent as,

$$D_{\mathrm{rob},i}(\bar{x}_i) := \frac{|\bar{x}_{i,0,k} \times \bar{x}_i|}{D(x_{i,0,k}, x_{i,\mathrm{g},k})}, \tag{6}$$

where $\bar{x}_{i,0,k} \times \bar{x}_i$ is the cross-product of the vectors $\bar{x}_{i,0,k}, \bar{x}_i$, and $D_{\mathrm{rob},i}$ is a measure of the distance between player agent i's location and the line segment represented by the current BVP $< x_{i,0,k}, x_{i,\mathrm{g},k} >$. As agent i travels, we record and update its maximum KD $D^{\max}_{\mathrm{rob},i}$. Then, each agent i uses $D^{\max}_{\mathrm{rob},i}$ to form an augmented obstacle space, defined as the Minkowski sum

$$\mathcal{X}^{\mathrm{aug}}_{\mathrm{obs},i} := \mathcal{X}_{\mathrm{obs},i} \oplus \mathcal{X}_{\mathrm{kin},i} \tag{7}$$

where $\mathcal{X}_{\mathrm{kin},i}$ is a compact set bounded by a sphere with a radius of $D^{\max}_{\mathrm{rob},i}$. This effectively adds a "buffer" to the obstacle space and defines $\mathcal{X}_{\mathrm{obs}}(t)$. We now prove a theorem first introduced in [25].

Theorem 1. *If an agent i expands all obstacles by the set $\mathcal{X}_{\mathrm{kin},i}$ and plans its path around these augmented obstacles, then the agent is guaranteed to avoid collisions.*

Proof. We prove this theorem by contradiction. Consider a collision between the agent i and its obstacle space $\mathcal{X}_{\mathrm{obs},i}$. This collision requires that at time t, $D(x_i(t), x_{\mathrm{o}}(t)) < d_{\mathrm{col}}$, where d_{col} is a "collision radius" indicating how close the agent is allowed to pass by obstacles. However, RRT$^{\mathrm{X}}$ seeks to avoid the augmented obstacle space $\mathcal{X}^{\mathrm{aug}}_{\mathrm{obs},i}$ which expands obstacles in all directions by a distance $D_{\mathrm{rob},i}$. This subsequently implies the agent's planned path will not travel within a distance $(D^{\mathrm{kin}}_{\mathrm{rob},i} + d_{\mathrm{col}})$ of obstacles. The agent can deviate from its path by a maximum distance $D^{\max}_{\mathrm{rob},i}$, but this still will not bring the distance between an agent and the obstacle space below d_{col}. Therefore, there are no possible states $x_i(t)$ where,

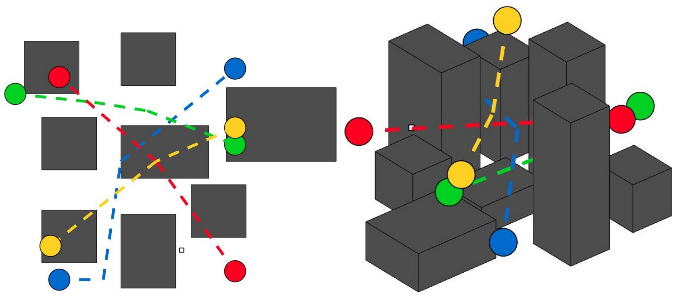
$$\min_{x_{\mathrm{o}} \in \mathcal{X}_{\mathrm{obs},i}(t)} \{D(x_i(t), x_{\mathrm{o}}(t))\} < d_{\mathrm{col}}, \tag{8}$$

and the theorem is true by contradiction. ■

In this work, we also presume that an agent's perception system for observing the obstacle space is affected by noise. Let us consider an obstacle point observed by the agent to follow $\tilde{x}_{\mathrm{obs}} = x_{\mathrm{obs}} + d_{\mathrm{var}}$, where $x_{\mathrm{obs}} \in \mathcal{X}_{\mathrm{obs},i}(t)$ is a point inside the true obstacle space and $d_{\mathrm{var}} = \mathcal{N}(0, \sigma^2(x)I_n) \in \mathbb{R}^n$ represents a random noise in the detected location of the obstacle, dependent on the distance between the agent and the obstacle. To ensure safety, we must account for this noise similarly to how we account for imperfect motion planning caused by kinodynamic distance constraints $D_{\mathrm{rob},i}$.

We seek to find the mean location of the obstacle and minimize the effects of the variance due to noise by using repeated measurements from the agent. In addition, we supplement each player agent's observations with observations from neighboring player agents in the environment to further mitigate the effects of noise in obstacle sensing and promote safety. To do so, we allow each player agent in the environment to distinguish between obstacles, and allow two player agents to recognize when they observe the same obstacle as one another.

In the urban air mobility problem, the environment will also be populated by other player agents and independent agents traversing paths towards their own destinations. In the case of cooperative player agents, we introduce a framework for these player agents to safely form paths around one another. For player agents to effectively plan a path around the motion of independent agents, player agents require the ability to predict their motion. We assume that independent agents are similarly driven towards an individual goal state while minimizing



(a) Top-down view (b) Isometric view

Fig. 1: A 3D example of an urban airspace populated with numerous drones.

a finite-horizon cost function (4). We structure the multi-agent planning problem as a game where each agent i in the environment is a player seeking an individual optimal control $u_i^\star$. As the agents in the environment do not directly impact one another's optimal control, we consider the game as non-cooperative and non-zero-sum. To predict how independent agents plan their motion in a non-cooperative game, we use a level-k cognitive hierarchy framework to formulate potential strategies that independent agents use to plan motion around other agents, and use the framework to predict the motion of these independent agents. These motion predictions allow the player agents to avoid future collisions with independent agents.

Definition 1. *Level-k rationality* is a model of game strategies using cognitive hierarchy, where a player with a level-k strategy (or level of rationality) assumes all other players employ a level-$(k-1)$ strategy and chooses its actions accordingly to minimize its own cost. □

For example, an independent agent with a level of rationality of 1 will assume that all other agents have a level of rationality of 0, a level-2 agent will assume all others are level-1, etc. The level-k strategies, and the likely resulting motion, are defined *a priori*. After predicting the strategies used by independent agents, player agents construct a predicted obstacle space $\hat{\mathcal{X}}_{\text{obs},i}$ to pre-emptively avoid the future states of independent agents. The total obstacle space then used by RRT$^{\text{X}}$ for the agent i is thus defined as the union of the augmented obstacle space (7) as well as $\hat{\mathcal{X}}_{\text{obs},i}$,

$$\mathcal{X}_{\text{obs,i}}^{\text{tot}} = \mathcal{X}_{\text{obs},i}^{\text{aug}} \cup \hat{\mathcal{X}}_{\text{obs},i}. \tag{9}$$

We define the strategy used by our player agents to adapt to the motion of independent agents as level-∞.

We now state the problem of motion planning in a 3D urban air environment.

Problem 1. *Urban air mobility problem*: Consider multiple communicating agents, known as the "player agents," in a 3D environment representing an urban area, possibly populated with static obstacles and independent agents. Given an initial state and goal state for each player agent, we seek a method to guide the player agents to their goal state both safely and optimally with respect to energy. Visual descriptions of the problem are shown in Figure 1. □

In previous work, we described the approach used to construct a path from an initial state to the goal state. We then modeled the dynamics of agents in the environment as a continuous-time linear system and define a finite-horizon cost function that drives the system optimally to its goal state. We also presented the kinodynamic constraints in the environment and mitigate the risk of collision. In this work, we consider the motion of other agents in the environment and construct strategies for collision avoidance with both cooperative and independent agents. We follow this by also considering uncertainty in the obstacle space based on inaccurate observations.

III. Decentralized Motion Planning

In this section, we propose a framework for individual player agents aware of each other's motion to avoid collisions with each other and prove Theorems originally presented in [25]. Consider two different player agents i, j with radii r_i, r_j respectively. Each agent is aware of the other's position x_i, x_j, and the goal state of their current BVP $x_{i,\text{g}}, x_{j,\text{g}}$. The obstacle space of i is continually updated to include an obstacle located at x_j with a radius r_j to avoid immediate collision. In addition, an obstacle $\mathcal{X}_{\text{obs},i,j}$ is defined with a radius r_j, and is added to the obstacle space of agent i along the path between x_j and $x_{j,\text{g}}$ to represent the current path of agent j. The obstacle representing the agent's predicted path is continuously updated as agent j approaches its current goal, and is updated to the next BVP along the path π_j when agent j begins along its next planned BVP. By defining an obstacle along the path of agent j, we can ensure that agent i does not plan a path that will cross in front of agent j and risk a collision. Agent j undergoes a similar process with the current location and planned path of the agent i. To ensure that the planned paths of both i and j are safe in a kinodynamic environment, the generated obstacles representing their paths are expanded as,

$$\mathcal{X}_{\text{obs},i,j}^{\text{aug}} := \mathcal{X}_{\text{obs},i,j} \oplus \mathcal{X}_{\text{kin},i,j}, \tag{10}$$

where $\mathcal{X}_{\text{kin},i,j}$ is the space of a compact set bounded by a sphere with a radius of $D_{\text{rob},i} + D_{\text{rob},j}$.

Theorem 2. *Consider agents i, j that are familiar with maximum KDs $D_{\text{rob},i}, D_{\text{rob},j}$ respectively. By expanding the obstacles representing one another by the space $\mathcal{X}_{\text{kin},i,j}$, the agents can guarantee avoiding a collision.*

Proof. Consider a collision between the agent i and agent j. The collision requires that at time t, $D(x_i(t), x_j(t)) < d_{\text{col}}$. However, each agent's path-planning seeks to avoid the expanded obstacle $\hat{\mathcal{X}}_{\text{obs},i,j}$ which adds a radius $D_{\text{rob},i}$. This means both agents will never plan to travel within a distance $D_{\text{rob},i} + D_{\text{rob},j}$ of each other. Agent j can deviate from its path by a maximum distance $D_{\text{rob},j}$, i.e., the minimum distance between the path π_i and state x_j is $D_{\text{rob},i}$. Vice versa is true for agent j, such that the minimum distance between the path π_j and state x_i is $D_{\text{rob},j}$. By Theorem 1, both of these agents still successfully avoid a collision with each other. Thus, the theorem is true by contradiction. ■

By Theorem 2, we ensure that the two player agents are not at risk of collision even when their motion is influenced by kinodynamic constraints.

In cluttered environments containing numerous player agents, this process can create situations where an individual player agent cannot find a viable path towards its goal state without intersecting another agent's path, and therefore cannot move. This deadlock is referred to as the "freezing robot problem" [27] (FRP). To address the FRP, each agent measures the distance it has traversed over time. If the player agent i does not exit a neighborhood with a user-chosen radius r_f over a certain time-frame, then that agent may be currently "frozen" in place by obstacles. If this occurs, the agent can communicate with other nearby player agents to force them to re-plan their own paths. It does so by temporarily changing the broadcasted KD to other agents to be a value $d > r_f$. The increased KD in turn motivates other agents to re-plan paths to avoid the area of the frozen player agent.

Theorem 3. *Consider a frozen agent i "trapped" in a neighborhood of radius r_f and blocked at least partially by another agent j. By increasing its broadcasted maximum KD $D_{\text{rob},i}$ by a distance l greater than r_f, agent i can guarantee that it can plan a path out of this neighborhood.*

Proof. Consider an agent i still trapped after increasing the $D_{\text{rob},i}$. This implies that agent j is close enough for the obstacle $\hat{\mathcal{X}}^{\text{aug}}_{\text{obs},i,j}$ to intersect with the neighborhood agent i is trapped in. This in turn implies that the agent j is within a distance $D_{\text{rob},i} + D_{\text{rob},j} + r_f$, or the sum of KDs as well as the radius of the neighborhood, of the center of the neighborhood. Therefore, the corresponding obstacle created by i that j seeks to avoid, $\hat{\mathcal{X}}^{\text{aug}}_{\text{obs},j,i}$, has a maximum augmentation distance of $D_{\text{rob},i} + D_{\text{rob},j} + r_f$. However, as the broadcasted KD was increased by a value l larger than r_f, the obstacle that j seeks to avoid is augmented by a value,

$$D_{\text{rob},i} + D_{\text{rob},j} + l > D_{\text{rob},i} + D_{\text{rob},j} + r_f, \tag{11}$$

and the agent j is not allowed within this distance by its path-planning algorithm, considering Theorem 2. Thus, the theorem is true by contradiction. ■

Following Theorem 3, agent i can successfully create a path out of the neighborhood f_i by increasing its KD.

IV. Cognitive Hierarchy

In this section, we briefly detail the cognitive hierarchy used by independent agents in the urban air environment, and then employ the cognitive hierarchy to determine the behavior of our player agents. The hierarchy we use is originally derived and detailed in [24], and includes two major levels of rationality, as higher levels of rationality result in a loop in strategies: level-0 rationality, and level-1 rationality. We then use the defined cognitive hierarchy to determine a strategy for our player agents to respond appropriately to each possible level of rationality, resulting in both safe and optimal planning in a diverse environment containing multiple strategies. We refer to the final player agent strategy as the *level-∞ strategy*.

In the cognitive hierarchy for urban air mobility, a level-0 agent i ignores all other agents in the environment, and construct the obstacle space $\mathcal{X}_{\text{obs},i}$ using solely perceived non-agent obstacles, i.e., the predicted obstacle space $\hat{\mathcal{X}}_{\text{obs},i} = \{\varnothing\}$.

Next, we consider a level-1 agent. Similarly to a level-0 agent, a level-1 agent j drives to its goal state by constructing its obstacle space $\mathcal{X}_{\text{obs},j}$ and seeks an optimal path $\pi^\star(x_{i,0}, x_{i,\text{g}})$. However, level-1 agents additionally attempt to predict the motion of other agents in the environment, and use this to construct a predicted obstacle space $\hat{\mathcal{X}}_{\text{obs},j}(t)$ to form a total obstacle space $\mathcal{X}^{\text{tot}}_{\text{obs},j}$. In this case, the level-1 agent j anticipates level-0 behavior from all other agents. The agent thus avoids collision by avoiding the *reachable set* over a set time-frame t_s, which is proven to be safe in Theorem 4.

Definition 2. The *reachable set* of an agent i at a time t and over a time frame t_s is a set of all states $x_{i,f}$ where there exists a control u_i such that $x(t + t_\text{s}) = x_{i,f}$. □

Theorem 4. *Consider a level-1 agent j that is familiar with the kinodynamic constraints of a different agent i and can observe that agent's current state and velocity vector. Then, agent j can plan a motion that is guaranteed to avoid a collision with agent i over a time frame t_s.*

Proof. This theorem is proven in [24]. ■

The level-1 motion planning strategy is also effective for avoiding the reachable set of a dynamic obstacle with no predictable rationality. As a result, any dynamic obstacles in the environment can also be interpreted as level-0 agents for higher level motion planning strategies.

After defining the cognitive hierarchy, we now consider the level-∞ strategy. While formulating the cognitive hierarchy, we find a loop in strategies. This means that the optimal strategy is to imitate a level-1 strategy when navigating around level-0 agents, and to imitate a level-0 strategy while navigating around level-1 agents. This ensures safety by avoiding collisions with level-0 agents, while also ensuring optimality by not excessively replanning to avoid level-1 agents. This approach is shown in Figure 2. Now that we have considered the player's response to each level of rationality, we attempt to determine the motion planning of the player agent by predicting the levels of rationality of each other agent in the environment and how to respond to each agent with the appropriate strategy.

V. Level of Rationality Prediction

In this section, we formulate a method to predict the levels of rationality of other agents using distributed Gaussian process classification (GPC). We first detail how GPC is used to predict the level of rationality of an agent by observing their motion over time. We then formulate a distributed GPC framework across multiple player agents to increase the accuracy of the level of rationality predictions. Next, we describe how the strategy predictions of other agents are used to further augment each player agent's obstacle space.

A Gaussian process classifier, as described in [28], [29] is a binary classification method where a Gaussian process model is used to predict whether an input w belongs to a given class.

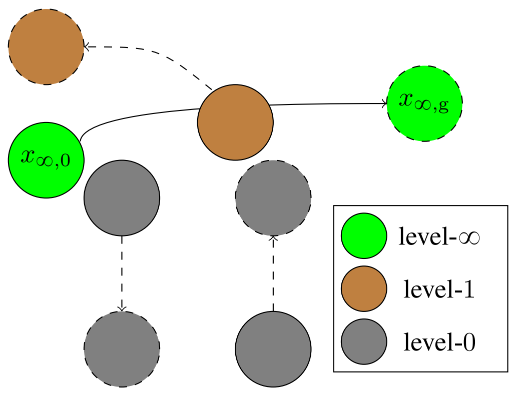


Fig. 2: The optimal path $\pi_q^\star(x_{\infty,0}, x_{\infty,g})$ of a level-∞ agent (shown in green).

Algorithm 1 rBCM

Input: $\mathcal{F}$ - set of learned posterior modes; Y - outputs; k - covariance function; K - covariance matrix; $p(\mathcal{Y}|\mathbf{f})$ - likelihood function; $w^\star$ - test input
Output: $p(f^\star|\mathcal{W}, \mathcal{Y}, w^\star)$ - Level-1 probability

```
 1: for f ∈ $\mathcal{F}$ do
 2:     $W_l \leftarrow -\nabla\nabla \log p(\mathcal{Y}|\mathbf{f})$                ▷ Hessian
 3:     $L_l \leftarrow \texttt{Chol}(I + W_l^{1/2} K W_l^{1/2})$  ▷ Cholesky Decomposition
 4:     $\bar{f}^\star \leftarrow k(w, w^\star)^\intercal \nabla \log p(\mathcal{Y}|\mathbf{f})$
 5:     $\mathbf{v} \leftarrow L_l \backslash (W_l^{1/2} k(w, w^\star))$
 6:     $\mathbb{V}_l[f^\star] \leftarrow k(w^\star, w^\star) - \mathbf{v}^\intercal \mathbf{v}$
 7:     $p_l(w^\star) \leftarrow \int \sigma(z) \mathcal{N}(z|\bar{f}^\star, \mathbb{V}_l[f^\star]) \mathrm{d}z$
 8:     $\beta_l = \frac{1}{2}(\log \sigma^2 - \log \mathbb{V}_l[f^\star])$
 9: end for
10: $p(f^\star|\mathcal{W}, \mathcal{Y}, w^\star) = \frac{\prod_{l=1}^{L} p_l^{\beta_l}(f^\star|\mathcal{W}_l, \mathcal{Y}_l, x^\star)}{p^{-1+\sum_l \beta_l}(f^\star|w^\star)}$
11: return $p(f^\star|\mathcal{W}, \mathcal{Y}, w^\star)$
```

In this work, we choose to use a Gaussian process classifier as it returns not only a binary classification, but a probability that the classification is accurate. This is useful as it allows player agents to gauge the certainty of their predictions, which is important for distributing the classification task as seen later in this section. As we consider two levels of rationality in this work, we treat predicting the levels of rationality of each agent as a binary classification problem that we solve with GPC by observing the motion of the agent over time. To do so, we must consider how each strategy of the cognitive hierarchy affects the agent motion. In the case of level-1 rationality, we expect the agent to be conservative in its motion, steering away from both dynamic obstacles and their current trajectories. In the context of the planned path π of level-1 agents, this results in numerous BVPs being cut short, as well as increased KDs as agents rapidly replan and correct their course. Conversely, level-0 agents do not share this obstacle avoidance behavior. This results in longer BVPs in general that are not cut short, and lower KDs as the agent's trajectory will rarely change.

To learn our Gaussian process classifier, we run offline simulations of both level-0 and level-1 agents moving through an urban air environment and keep a record of the calculated paths and when they change over the course of the simulation. For each BVP along an agent's path, we take note of both the BVP length and the maximum KD along it to use as inputs for the classifier, along with user-supplied tags of $y = 1$ for "re-planned" and $y = -1$ for "non-re-planned" BVPs. Numerous simulations are executed with agents of different rationality levels to create the training set $\mathcal{D}$ used to train the Gaussian process model. Next, we use the trained model to estimate the probability that a given input w was planned by a level-1 agent, and we define this probability as $p(w)$.

To improve the accuracy of classifications of each agent's level of rationality, and to allow player agents to classify independent agents that they cannot directly observe, we utilize a distributed form of classification called Distributed GPC. Distributed GPC [30] is a method of using numerous trained "expert" models, each trained individually with a subset of the total training set, to reduce the computations necessary for classification. We now consider an independent agent i of an unknown level of rationality. Consider an environment containing some number $L > 1$ of level-∞ player agents capable of observing independent agent i, and each player agent has its own model $\mathbf{f}_l$, $l = 1, \ldots, L$ for predicting the likelihood that a BVP was re-planned. Each model $\mathbf{f}_l$ is trained with a unique subset of the total training data $(\mathcal{W}_l, \mathcal{Y}_l)$. As agent i traverses its path π_i and begins along a new BVP, each player uses its model to estimate the likelihood that the previous BVP was re-planned. Each of models' prediction, in turn, will be incorporated into the final distributed prediction.

We employ a distributed classification method named the robust Bayesian Committee Machine (rBCM), which assigns weights to each expert's prediction based on its output variance. The assigned weights help to alleviate concerns of sparse data in each model by placing more weight on models with training data that more closely resembles the current input. Let us assign each player agent as an "expert" for rBCM, where each expert has previously learned a model $\mathbf{f}_l$ and a likelihood function $p(\mathcal{Y}|\mathbf{f}_l)$. For a given test input $w^\star$, we then calculate the predictive distribution of rBCM $p(f^\star|\mathcal{W}, \mathcal{Y}, w^\star)$ using each expert's predictions as shown in Algorithm 1.

Remark 1. Note that rBCM is a method of distributed classification that relies on a central server. A decentralized variant of rBCM is discussed in [31]. Although rBCM is distributed, the proposed motion planner remains decentralized and does not require a central server.

As an independent agent i navigates the environment, the player agents monitor its motion over time, and observe when it begins a new BVP. Then, each BVP along its path is provided as a test case $w_i^\star = (\ell_{\mathrm{BVP}}, KD_{\mathrm{BVP}})$ for each player agent, which is subsequently used by rBCM to create an estimate of the level of rationality of the given agent $\mathrm{LoR}_i = p(f^\star|\mathcal{W}, \mathcal{Y}, w_i^\star) \in [0, 1]$. A higher value of LoR_i indicates stronger likelihood that the agent is following a level-1 motion planning strategy.

We now consider how to alter the obstacle space of player agents in response to estimated levels of rationality and ensure optimal motion planning. In the previous section, we discussed how a level-1 agent computes the *reachable set* during a time frame t_{s} of the other agents in the environment, and how conversely a level-0 agent does not respond to other agents. In addition, player agents desire to mimic either one of these levels of rationality, depending on the strategies of other agents in the environment. This presents us with a relatively simple way to alter the obstacle space depending not only on the perceived level of rationality of another agent, but also on the likelihood that this prediction is accurate.

For each independent agent in the environment i, we add their estimated reachable set $\mathcal{X}_{\text{rs},i}$ over a time frame $t_{\text{s},i}$ to the player agents' obstacle space, where $t_{\text{s},i} \in [0, t_{\text{s,max}}]$ is a value chosen depending on the agent's estimated level of rationality LoR_i between 0 and a maximum value $t_{\text{s,max}}$. Specifically, we choose a value $t_{\text{s},i}$ inversely proportional to LoR_i,

$$t_{\text{s},i} = t_{\text{s,max}}(1 - \text{LoR}_i). \tag{12}$$

In practice, this means that the player agents aim to avoid a reachable set of agent i. The size of $t_{\text{s},i}$ is related to the reachable set and is proportional to the likelihood that agent i is level-0. The impact on the obstacle space of the agent is conversely decreased if it is more likely to be a level-1 agent, down to a minimum time frame of $t_{\text{s},i} = 0$. In this case, the reachable set only consists of the current location of the agent, although in practice this value of $t_{\text{s,max}}$ will not be reached as agent levels will never be known for certain. This results in player agents being more cautious of level-0 agents to ensure safe planning, while also being less cautious of level-1 agents to ensure no unnecessary loss in optimality while planning a path.

VI. Obstacle Uncertainty

In this section, we formulate how a method of using repeated observations by multiple player agents to construct a shared obstacle space which will ensure that other player agents are familiar with obstacles which they cannot currently detect. We again consider some number L cooperating player agents in the environment capable of repeatedly observing an obstacle located at x_{obs}. Each observation of the obstacle made by a given agent $l \in L$ is written as

$$\tilde{x}_{\text{obs},l,o} = x_{\text{obs}} + d_{\text{obs},l,o},\ o \in \mathcal{O}_l, \tag{13}$$

where $d_{\text{obs},l,o} \in \mathbb{R}^n$ is a noise vector composed of normal random variables centered on 0. Using their own observations, each agent compiles the average perceived location of the obstacle,

$$\tilde{x}^{\text{avg}}_{\text{obs},l} = \frac{\sum_{o=1}^{\mathcal{O}_l} \tilde{x}_{\text{obs},l,o}}{\mathcal{O}_l}, \tag{14}$$

where each value $\tilde{x}_{\text{obs},l,i}$ correlates with a unique observation made by agent l. We then use the average location of each player agent to create a final estimate of the obstacle location x_{obs}, as discussed in Theorem 5.

Assumption 2. The standard deviation of the noise in an agent's observations of a given obstacle is proportional to distance from the obstacle and can be modeled as $\sigma_{\text{obs},l} = cD(x_l, \tilde{x}^{\text{avg}}_{\text{obs},l})$ where c is an identical constant for each player agent. □

Theorem 5. *Given Assumption 2, the effects of noise on the final estimate of an obstacle's location can be minimized by taking the final estimate to be*

$$x_{\text{obs}} = \frac{\sum_{l=1}^{\mathcal{L}} D(x_l, \tilde{x}^{\text{avg}}_{\text{obs},l})^{-2} \tilde{x}^{\text{avg}}_{\text{obs},l}}{\sum_{l=1}^{\mathcal{L}} D(x_l, \tilde{x}^{\text{avg}}_{\text{obs},l})^{-2}}. \tag{15}$$

Proof. It can be shown that for any number of independent observations y_i with standard deviations σ_i, the variance of the weighted sum of observations Y is minimized by taking the inverse-variance weighted average

$$Y = \frac{\sum_{i=1} \sigma_i^{-2} y_i}{\sum_{i=1} \sigma_i^{-2}}. \tag{16}$$

By using this property while also substituting the standard deviation with the equation in Assumption 2, we can then minimize the variance of the final estimate of an obstacle's location with the equation

$$x_{\text{obs}} = \frac{\sum_{l=1}^{\mathcal{L}} c^{-2} D(x_l, \tilde{x}^{\text{avg}}_{\text{obs},l})^{-2} \tilde{x}^{\text{avg}}_{\text{obs},l}}{\sum_{l=1}^{\mathcal{L}} c^{-2} D(x_l, \tilde{x}^{\text{avg}}_{\text{obs},l})^{-2}}, \tag{17}$$

where the constant c^{-2} can be removed from both the numerator and denominator to result in Equation (15). ■

This approach requires a limited number of samples to accurately estimate the location of an obstacle, and forming the estimated obstacle location is not computationally intensive. As a result, this approach requires minimal increase in the computational capabilities of player agents.

VII. Motion Planning Framework

The motion planning structure of a given player agent i consists of five stages that are repeated throughout a player agent's motion: (i) dynamic planning with RRT^{X} ; (ii) Q-learning control; (iii) terminal state evaluation; (iv) obstacle space computation and augmentation, and (v) level of rationality estimation with distributed GPC. We term our method as Distributed RRT-Q^{X}. The pseudo-algorithm is shown in Algorithm 2. Each player agent plans its own path according to this framework, based on observations and trained GPC model, as well as the observations and models of other agents as specified in the algorithm.

Extension from Previous Work: The proposed method builds our previous work [24], [25], but is superior in several key aspects. First, by combining predicting levels of rationality and distributed motion planning, we present an algorithm capable of planning safe and optimal motion through environments containing both cooperative and independent agents. We further improve predicting the level of rationality of independent agents by combining the observations of numerous player agents, weighing their predictions based on their certainty, and making changes to the predicted obstacle space. The predicted obstacle space is a function of the rationality likelihood (i.e., level-0 or level-1), rather than using a simple binary prediction of an agent's strategy. Lastly, this work is first to incorporate distance-varying noise into agent's observations of both other

Framework	Ge [14]	Semnani [19]	Chen [15]	Dist. RRT-Q$^{\mathrm{X}}$
Optimality	Energy	Time	No	Energy
Agents	Both	Indep.	Coop.	Both
Scalability	Offline	Online	Online	Online
Dynamics	Known	Learned	Known	Learned
Disturbances	No	No	No	Yes

TABLE I: Comparison of several multi-agent motion planning algorithms.

agents and of obstacles in the environment, and successfully mitigates this noise through agent cooperation.

Qualitative Comparison: We present a qualitative comparison between our proposed framework and other works in Table I. For this comparison we consider prior work most similar to our own, dealing with multi-agent motion planning in dynamic environments. We compare these frameworks on the following aspects: 1) Optimality of the agent's control; 2) Cooperative nature of other agents in the environment, i.e., cooperative, independent, or both; 3) Computational scalability of path planning algorithm, i.e., offline or online; 4) Knowledge of system dynamics, i.e., known, approximated a priori, or learned in real-time; 5) Disturbances in agent's observations from the environment. As shown, the frameworks proposed in [19] and [15] deal with independent and cooperative agents, respectively. Distributed RRT-Q$^{\mathrm{X}}$ is capable of navigating around both efficiently to minimize the replanning necessary and the risk of collision depending on the situation. The framework shown in [14] also acommodates both kinds of agents, but plans a path offline and only makes minimal adjustments to this path to avoid collision, which is unreliable if an environment becomes congested after initial planning. Additionally, Distributed RRT-Q$^{\mathrm{X}}$ does not require system dynamics a priori unlike most other proposed frameworks, and is the only one to consider and to mitigate noise in observations of the environment.

VIII. Simulations

To test our classification approach, we use simulations of a variety of three-dimensional urban environments, populated with a variety of static obstacle spaces $\mathcal{X}_{obs}$. Each simulation involves several agents given a random initial and goal state, and operating individually with either level-0 or level-1 rationality. Each agent in the simulation follows (3), with dynamics and user-defined cost matrices identical to those in [25].

In each simulation, the agents are given a random initial and goal state and calculate their optimal path to the goal using RRT-Q$^{\mathrm{X}}$ and the proposed path planning methodology for their respective level of rationality. After each case, the BVPs calculated by each agent are saved and flagged if they were the result of re-planning to avoid obstacles. The saved BVPs comprise the training data for several GP expert models, each of which are provided with a subset of the data.

Additional simulations are conducted containing player agents with these trained expert models, and each agent in the environment observes the location of all other agents, accompanied with distance-varying noise. The observations of each agent are used to create an estimate of each agent's location over time. We repeat the process of observing independent agent motion to estimate their BVPs and generate noisy testing data for our GPC models. The results of the noisy classification tests are shown in Table III, and demonstrate that GPC reliably identifies the replanned paths of level-1 agents even in noisy environments. The predictions are then factored into the simulations to lessen the size of obstacles if they are predicted to be more likely to be level-1 agents, allowing level-∞ player agents to plan more optimal paths while still being safe. A large simulation demonstrating the complete implementation is shown in Figure 3, and includes level-1, level-0, and level-∞ agents. In this simulation, both level-1 and level-∞ agents successfully planned safe motion to their goals, and the level-∞ agents were found to plan shorter paths than the level-1 agents by correctly identifying, and minimizing replanning around, level-1 agents. A separate

Algorithm 2 Distributed RRT-Q$^{\mathrm{X}}$

Input: T - finite horizon; Δt - resolution; M, R - cost weight matrices; $P(T)$ -fixed Riccati matrix; ρ - admissible window; x_{goal} - goal state; x_{start} - start state; $\mathcal{L}$ - Set of player agents; n - Set of independent agents; w_i - BVP of agent i; $\mathcal{X}_{\text{obs},l}$ - obstacle space of agent $l \in \mathcal{L}$; $\mathcal{X}_{\text{a}}$ - states of other agents; $\mathcal{X}$ - state space; L - level of rationality; t_{s} - agent range time-frame; t_f - freezing check time-frame; r_f - freezing radius
Output: $\hat{u}$ - control

```
1: $\alpha_{\text{a}}, \alpha_{\text{c}} \leftarrow \texttt{Stability}(M, R)$;    ▷ Initialization
2: $\mathcal{X}_{\text{obs}} \leftarrow \texttt{ObstacleEstimation}(\mathcal{X}_{\text{obs},l}),\ l \in \mathcal{L}$;
3: $\mathcal{X}_{\text{obs}}^{\text{aug}} \leftarrow \mathcal{X}_{\text{obs}}$;
4: $D_{\text{rob}}, D_{\text{rob}}^{\text{kin}} \leftarrow 0;\ k \leftarrow 1$;
5: for $i \in n$ do
6:     $\text{LoR}_i = 0$;
7: end for
8: while $x_{\text{goal}} \neq x$ do
9:     $\mathcal{G}, \pi \leftarrow \texttt{RRT}^{\text{X}}(\mathcal{X}, \mathcal{X}_{\text{obs}}^{\text{aug}}, x_{\text{start}}, x_{\text{goal}})$;    ▷ Dynamic planning
10:    $\hat{W}_{\text{c}} \leftarrow \texttt{Critic}(M, R, \Delta t, \alpha_{\text{c}}, \bar{x}, \hat{u})$;    ▷ Q-learning
11:    $\hat{\mathcal{Q}}_i \leftarrow \texttt{EstimateQ}(\hat{W}_{\text{c}}, \bar{x}, \hat{u})$;
12:    $\hat{W}_{\text{a}} \leftarrow \texttt{Actor}(\hat{\mathcal{Q}}_i, \alpha_{\text{a}}, \bar{x})$;
13:    $\hat{u} \leftarrow \texttt{Control}(\hat{W}_{\text{a}}, \bar{x})$;
14:    return $\hat{u}$;
15:    $D_0 \leftarrow \texttt{Distance}(x_{0,k}, x_{\text{g},k})$ (1);
16:    $D \leftarrow \texttt{Distance}(x_k, x_{\text{g},k})$;
17:    if $D \leqslant \rho D_0$ then    ▷ Terminal state evaluation
18:        $x_{0,k} \leftarrow x(t)$;
19:        $k \leftarrow k + 1$;
20:        break;
21:    end if
22:    $\hat{\mathcal{X}}_{\text{obs}} \leftarrow \texttt{PotentialStates}(\mathcal{X}_a, L, t_{\text{s}}, \text{LoR}_i)$;
23:    $\mathcal{X}_{\text{obs}}^{\text{tot}} \leftarrow \mathcal{X}_{\text{obs}} + \hat{\mathcal{X}}_{\text{obs}}$    ▷ Predictive obstacle avoidance
24:    if $t < T$ then
25:        $D_{\text{rob}} \leftarrow \texttt{KinodynamicDist}(x_{0,k}, \bar{x}, D_0)$ (6);
26:        if $\texttt{Norm}(x(t) - x(t - t_f)) < r_f$ then
27:            $D_{\text{rob}} \leftarrow \texttt{Expand}(D_{\text{rob}})$    ▷ Freezing check
28:        end if
29:        if $D_{\text{rob}} > D_{\text{rob}}^{\text{kin}}$ then    ▷ Obstacle augmentation
30:            $D_{\text{rob}}^{\text{kin}} \leftarrow D_{\text{rob}}$;
31:        end if
32:        $\mathcal{X}_{\text{obs}}^{\text{aug}} \leftarrow \texttt{Augment}(\mathcal{X}_{\text{obs}}^{\text{tot}}, D_{\text{rob}}^{\text{kin}})$ (7);
33:    end if
34:    for $i \in n$ do
35:        $\text{LoR}_i = \texttt{rBCM}(p(w_i)_l), l \in \mathcal{L}$    ▷ LoR Estimation
36:    end for
37: end while
```

Level of Rationality	Level-1	Level-∞
Avg. Instances of Replanning	14.3	10.7
Avg. Potential Kinodynamic Collisions	3.6	1.7
Deadlock Rate	40%	10%

TABLE II: Comparison of the performance of 10 level-1 agents and 10 level-∞ agents across several simulations of a densely populated environment.

simulation demonstrating the methodology for avoiding the freezing robot problem is shown in Figure 4. In the deadlock simulation, a single agent that is unable to find a path to its goal state successfully creates a path by using its broadcasted kinodynamic distance to drive other agents away from it. This result was verified in repeated simulations with a minimal risk of deadlocks, where no path could be found for an agent. In Table II, we compare the performance of these level-∞ agents with an identical simulation where all agents instead exhibit level-1 rationality. This alternative simulation was found to result in significantly more replanning, a much higher risk of collisions due to kinodynamic constraints, and frequent deadlocks where no valid path is found for at least one agent. Videos of these simulations can be found at:

- https://youtu.be/5OZ-4UsUKGE
- https://youtu.be/rhEZwF9WjoE

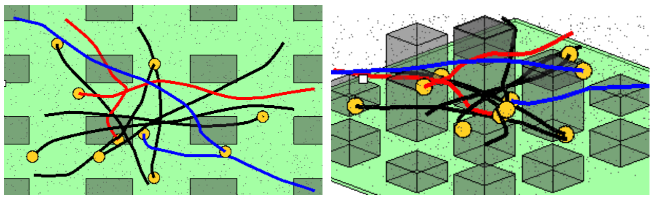

(a) Top View (b) Side View

Fig. 3: A test simulation containing 10 agents. Level-0 agents (shown in black) travel to their goal state while ignoring other agents. Also present are level-1 agents (shown in blue) and level-∞ agents (shown in red). In this simulation, the level-∞ agents to take a shorter path with less replanning.

IX. Conclusion

This paper proposes a motion planning framework for congested urban air environments using distributed classification, termed as Distributed RRT-Q$^{\mathrm{X}}$. The proposed method classifies independent agent's levels of rationality in accordance with a pre-constructed cognitive hierarchy and implements safe kinodynamic motion planning to avoid independent agents and their predicted motion. Distributed RRT-Q$^{\mathrm{X}}$ also allows for cooperating player agents to plan safe paths while also preventing deadlocks. Repeated simulations show that this approach significantly reduces deadlocks and the frequency of online replanning compared to a naive strategy that avoids all potential future motion of other agents. As a result, the proposed planner reduces the risk of collisions between agents and obstacles. For the kinodynamic motion planning, we integrate a dynamic path planning algorithm to navigate through

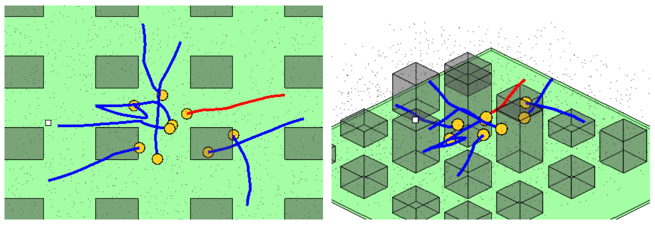

(a) Top View (b) Side View

Fig. 4: A test simulation containing 10 level-∞ agents. One agent (shown in red) cannot find a path forward through the other agents, and becomes "frozen." As a result, it increases its broadcasted kinodynamic distance to drive other agents (shown in blue) away from its current location, allowing it to find a path to its goal state by driving away other agents.

Type of BVP	Original Path	Re-planned Path
Model 1 $p(w)$	0.36	0.71
Model 2 $p(w)$	0.22	0.68
Model 3 $p(w)$	0.38	0.71
Model 4 $p(w)$	0.18	0.64

TABLE III: The average likelihood $p(w)$ of a replanned path predicted by four different GPC models for various BVPs, categorized as either BVPs along the agent's original path or re-planned BVPs.

these environments with a Q-learning controller to learn the optimal control policies and execute waypoint navigation. This structure yields scalable computations suitable for real-time deployment. In addition, we introduce a method to estimate obstacle locations that accounts for potential sensor noise. The results indicate safe and efficient motion planning for all agents, even in realistic environments with sensor noise.

While this work presents a motion planning methodology in the context of UAVs operating in an urban air environment, the framework is applicable to a wide range of multi-agent environments. The generalized nature of the model-free control algorithm, the RRT-Q$^{\mathrm{X}}$ path planning algorithm, the cognitive hierarchy model, and the level of rationality learning strategy enable Distributed RRT-Q$^{\mathrm{X}}$ to operate effectively in environments with controllable cooperative player agents, any number of observable independent agents, and an observable dynamic obstacle space. Future work will focus on executing optimal timed transitions between waypoints while predicting agent's motions. This will ensure temporal task specifications in an experimental platform. In addition, we will explore the optimization of actor-critic learning parameters to accelerate convergence to optimal solutions.

References

[1] A. Bauranov and J. Rakas, "Designing airspace for urban air mobility: A review of concepts and approaches," *Progress in Aerospace Sciences*, vol. 125, p. 100726, 2021.

[2] D. P. Thipphavong, R. Apaza, B. Barmore, V. Battiste, B. Burian, Q. Dao, M. Feary, S. Go, K. H. Goodrich, J. Homola, I. Husni, P. Kopardekar, J. Lachter, N. Neogi, H. K. Ng, R. M. Oseguera-Lohr, M. D. Patterson, and S. A. Verma, "Urban air mobility airspace integration concepts and considerations," in *Aviation Technology, Integration, and Operations Conference*, 2018, p. 3676.

[3] A. Straubinger, R. Rothfeld, M. Shamiyeh, K.-D. Büchter, J. Kaiser, and K. O. Plötner, "An overview of current research and developments in urban air mobility–setting the scene for UAM introduction," *Journal of Air Transport Management*, vol. 87, p. 101852, 2020.

[4] R. Rothfeld, M. Balac, K. O. Ploetner, and C. Antoniou, "Agent-based simulation of urban air mobility," in *Modeling and Simulation Technologies Conference*, 2018, p. 3891.

[5] ——, "Initial analysis of urban air mobility's transport performance in Sioux Falls," in *Aviation Technology, Integration, and Operations Conference*, 2018, p. 2886.

[6] G.-Z. Yang, J. Bellingham, P. E. Dupont, P. Fischer, L. Floridi, R. Full, N. Jacobstein, V. Kumar, M. McNutt, R. Merrifield, B. Nelson, B. Scassellati, M. Taddeo, R. Taylor, M. Veloso, Z. L. Wang, and R. Wood, "The grand challenges of science robotics," *Science Robotics*, vol. 3, no. 14, p. eaar7650, 2018.

[7] B. Donald, P. Xavier, J. Canny, and J. Reif, "Kinodynamic motion planning," *Journal of the ACM*, vol. 40, no. 5, pp. 1048–1066, 1993.

[8] M. Otte and E. Frazzoli, "RRT$^{\mathrm{X}}$: Asymptotically optimal single-query sampling-based motion planning with quick replanning," *The International Journal of Robotics Research*, vol. 35, no. 7, pp. 797–822, 2016.

[9] G. P. Kontoudis, K. G. Vamvoudakis, and Z. Xu, "RRT-QX real-time kinodynamic motion planning in dynamic environments with continuous-time reinforcement learning," in *Brain and Cognitive Intelligence Control in Robotics*. CRC Press, 2022, pp. 1–19.

[10] Z. Xu, G. P. Kontoudis, and K. G. Vamvoudakis, "Online and robust intermittent motion planning in dynamic and changing environments," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 12, pp. 17 425–17 439, 2024.

[11] K. G. Vamvoudakis, "Q-learning for continuous-time linear systems: A model-free infinite horizon optimal control approach," *Systems & Control Letters*, vol. 100, pp. 14–20, 2017.

[12] R. E. Allen and M. Pavone, "A real-time framework for kinodynamic planning in dynamic environments with application to quadrotor obstacle avoidance," *Robotics and Auton. Systems*, vol. 115, pp. 174–193, 2019.

[13] G. P. Kontoudis and K. G. Vamvoudakis, "Kinodynamic motion planning with continuous-time Q-learning: An online, model-free, and safe navigation framework," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 12, pp. 3803–3817, 2019.

[14] D. Ge and U. Topcu, "Hierarchical path planning for urban on-demand air mobility," in *IEEE Conference on Control Technology and Applications*, 2019, pp. 894–899.

[15] J. Chen, J. Li, C. Fan, and B. C. Williams, "Scalable and safe multi-agent motion planning with nonlinear dynamics and bounded disturbances," in *AAAI Conference on Artificial Intelligence*, vol. 35, no. 13, 2021, pp. 11 237–11 245.

[16] H. A. Simon, *Models of bounded rationality: Empirically grounded economic reason*. MIT Press, 1997, vol. 3.

[17] C. Mavrogiannis, F. Baldini, A. Wang, D. Zhao, P. Trautman, A. Steinfeld, and J. Oh, "Core challenges of social robot navigation: A survey," *ACM Trans. on Human-Robot Interaction*, vol. 12, no. 3, pp. 1–39, 2023.

[18] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of human–robot cooperation," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 335–356, 2015.

[19] S. H. Semnani, H. Liu, M. Everett, A. De Ruiter, and J. P. How, "Multi-agent motion planning for dense and dynamic environments via deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3221–3226, 2020.

[20] C. F. Camerer, T.-H. Ho, and J.-K. Chong, "A cognitive hierarchy model of games," *The Quarterly Journal of Economics*, vol. 119, no. 3, pp. 861–898, 2004.

[21] A. Kanellopoulos and K. G. Vamvoudakis, "Non-equilibrium dynamic games and cyber–physical security: A cognitive hierarchy approach," *Systems & Control Letters*, vol. 125, pp. 59–66, 2019.

[22] N.-M. T. Kokolakis, A. Kanellopoulos, and K. G. Vamvoudakis, "Bounded rational unmanned aerial vehicle coordination for adversarial target tracking," in *American Control Conference*, 2020, pp. 2508–2513.

[23] X. Zhang, X. Zeng, and Z. Peng, "Enhancing autonomous racing strategies: A cognitive hierarchy-based safe motion planning approach," in *Chinese Control Conference*, 2024, pp. 6463–6468.

[24] J. Netter, G. P. Kontoudis, and K. G. Vamvoudakis, "Bounded rational RRT-Q$^{\mathrm{X}}$: Multi-agent motion planning in dynamic human-like environments using cognitive hierarchy and q-learning," in *IEEE Conference on Decision and Control*, 2021, pp. 3597–3602.

[25] J. Netter and K. G. Vamvoudakis, "Decentralized multi-agent motion planning in dynamic environments," in *American Control Conference*, 2023.

[26] G. P. Kontoudis and K. G. Vamvoudakis, "Robust kinodynamic motion planning using model-free game-theoretic learning," in *American Control Conference*, 2019, pp. 273–278.

[27] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 797–803.

[28] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*. MIT press Cambridge, MA, 2006, vol. 2, no. 3.

[29] H. Nickisch and C. E. Rasmussen, "Approximations for binary gaussian process classification," *Journal of Machine Learning Research*, vol. 9, no. 10, pp. 2035–2078, 2008.

[30] M. Deisenroth and J. W. Ng, "Distributed Gaussian processes," in *International Conference on Machine Learning*. PMLR, 2015, pp. 1481–1490.

[31] G. P. Kontoudis and D. J. Stilwell, "Multi-agent federated learning using covariance-based nearest neighbor Gaussian processes," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 4, pp. 115–138, 2025.

Josh Netter is from Athens, Georgia, USA. In 2018, he received Bachelors of Science in both Aerospace Engineering and Computer Science at the Georgia Institute of Technology in Atlanta, Georgia. He then continued on at Georgia Tech to receive a Master of Science degree in Aerospace Engineering in 2020. He now works as a graduate research assistant and Ph.D. candidate under Dr. Kyriakos G. Vamvoudakis in the Guggenheim School of Aerospace Engineering at Georgia Tech. His expertise is primarily in control theory, safe autonomy, game theory, bounded rationality, urban air mobility, and reinforcement learning.

George P. Kontoudis (M'22) received the M.S. and PhD degrees in Mechanical Engineering and Electrical Engineering at Virginia Tech, in 2018 and 2021 respectively. He was a Postdoctoral Research Associate in the Aerospace Engineering Department at the University of Maryland, College Park for two years. Since 2024, he is an Assistant Professor with the Mechanical Engineering Department, Colorado School of Mines. His research interests include multi-agent systems, Gaussian processes, motion planning, and optimal control.

Kyriakos G. Vamvoudakis (Senior Member of IEEE) was born in Athens, Greece. He received the Diploma (a 5-year degree, equivalent to a Master of Science) in Electronic and Computer Engineering from the Technical University of Crete, Greece in 2006 with highest honors. After moving to the United States of America, he studied at The University of Texas at Arlington with Frank L. Lewis as his advisor, and he received his M.S. and Ph.D. in Electrical Engineering in 2008 and 2011 respectively. He currently serves as the Dutton-Ducoffe Endowed Professor at The Daniel Guggenheim School of Aerospace Engineering at Georgia Tech. He holds a secondary appointment in the School of Electrical and Computer Engineering. His expertise is in reinforcement learning, control theory, game theory, cyber-physical security, bounded rationality, and safe/assured autonomy.