Bounded Rational RRT-QX: Multi-Agent Motion Planning in Dynamic Human-Like Environments Using Cognitive Hierarchy and Q-Learning

Josh Netter [1], George P. Kontoudis [2], and Kyriakos G. Vamvoudakis [1]

***Abstract*— This paper presents a multi-agent motion planning algorithm for human-like navigation in dynamic environments. A cognitive hierarchy approach is used to model the motion of autonomous agents. We discuss potential levels of rationality and introduce a method to predict them in real-time. The rationality level prediction is achieved by observing the kinodynamic distance (KD) of other agents. An offline training phase is required to learn the maximum KD from multiple boundary value problems. Collision avoidance is ensured by introducing artificial obstacles in the environment based on the predicted levels of rationality. The motion planning is then carried out using RRT-Q^X. The effectiveness of the bounded rational motion planning algorithm is illustrated in simulations.**

I. Introduction

Recent developments in artificial intelligence have significantly advanced the capabilities of mobile robots. However, navigation remains a key problem for autonomous vehicles, while obstacle avoidance and safe path planning are considered necessary for applications of robotics [1]. Safe motion planning is difficult in dynamic environments, where the environment changes over time. This is particularly true in multi-agent environments, where an autonomous robot must navigate past other agents both with unknown destinations and path planning frameworks depending on their level of rationality. To avoid these other agents, it is crucial to quickly identify other agents in the environment and to classify their motion planning methodology. The robot must also be able to adjust these classifications online to account for changes in other agent's motions, and must accommodate online path replanning to utilize these observations to avoid collisions. Our focus in this work is on exploring how varying levels of rationality affect the path planning of agents navigating an environment, as well as identifying these levels of rationality in autonomous robots with only knowledge of its motion.

Related work: Rapidly-exploring Random Tree (RRT) [2], is a sampling-based path planning algorithm to navigate high-dimensional static environments. RRT is probabilistically complete. It is expanded upon with RRT*, which is shown to be asymptotically optimal in static environments [3]. The main drawback of RRT* is the computational complexity that does not allow for rapid replanning in dynamic environments. An extension of RRT* for dynamic environments, referred to as RRT^X, is introduced in [4], [5]. This algorithm provides asymptotically optimal motion planning and replanning in dynamic environments. However, this approach requires the dynamics of the system. Following this, RRT-Q^X is proposed in [6] as a sampling-based algorithm for navigating an unpredictable dynamic environment using a model-free Q-learning controller. This approach, however, does not account for multi-agent environments.

In a multi-agent environment, it is critical to model the motion of other agents in order to optimize path planning and ensure safety. This is primarily looked at through the lens of robots sharing environments with one or several humans, who are presumed to all have similar planning techniques [7]. In [8], human motion in an environment is estimated using an intent-driven model, yet does not consider a denser environment with additional agents necessitating further path replanning for some agents present, nor does it consider agents with differing levels of intelligence. Human motion in an environment was also modelled in [9] by considering each human as a player in a non-zero sum game, and learning from human motion examples. This was then used to guide an agent in [10]. Similarly, this work does not consider online motion past additional obstacles, or with agents with varying levels of intelligence. Robot navigation through an environment with numerous human agents is considered in [11], where human motion is modeled with interacting Gaussian processes. This work considers the influence of multiple other agents, but all still are assumed to operate with the same level of intelligence. A model for avoiding collisions accounting for varying levels of intelligence and cooperation is proposed in [12], but in this work agents are capable of communicating to jointly avoid a collision.

The kinodynamic motion problem is introduced in [13]. The kinodynamic distance (KD) of an agent is defined in [14], [15] as the distance between an agent and its planned path. Bounded rationality is presented in [16], referring to agents making decisions with imperfect information of their environment. Following from bounded rationality, cognitive hierarchy is a method of describing the relative intelligence of multiple players in a game [17] which is also extended for cyber-physical systems [18], but has not been applied to a multi-agent motion planning problem.

Contributions: The contributions of this is threefold. First, we consider and propose models of agent motion planning behavior in a multi-agent environment for several levels of agent rationality. Second, we implement these models on agents using a path-planning algorithm for online navigation through dynamic environments to measure effectiveness in

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

[2]G. P. Kontoudis is with the Bradley Department of Electrical and Computer Engineering, Virginia Tech, USA (email: gpkont@vt.edu).

This work was supported in part, by NSF under grant Nos. CAREER CPS-1851588, CPS-2038589, SATC-1801611, and by NASA ULI under grant number 80NSSC20M0161.

obstacle avoidance. Lastly, we employ an algorithm using each agent's KD to predict the level of rationality.

II. Problem Formulation

Consider an environment containing N agents with different capabilities. Each agent can be described as a linear-time-invariant system,

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

where $x_i(t) \in X \subseteq \mathbb{R}^n$ is the measurable state vector, $u_i(t) \in U \subseteq \mathbb{R}^m$ is the control input, and $A \in \mathbb{R}^{n\times n}$ and $B \in \mathbb{R}^{n\times m}$ are plant and input matrices, respectively, for each agent $i = 1, \ldots, N$. Let us define the difference of an agent's current state $x_i(t)$ and its goal state $x_{i,\mathrm{g}}$ as $x_i(t)$, and define,

$$\dot{x}_i(t) = Ax_i(t) + Bu_i(t), \quad t \geq 0. \tag{1}$$

We seek to guide an agent i in the environment from its initial state, $x_{i,0}$, to an individual goal state $x_{i,\mathrm{g}}$. We employ a finite horizon cost function,

$$J(x_{i,0}, u_i; t_0, T) = \phi(T) + \frac{1}{2}\int_{t_0}^{T} \left(x_i^\top M x_i + u_i^\top R u_i\right) d\tau, \tag{2}$$

where $\phi(T) := (1/2)x_i^\top(T)P(T)x_i(T)$ is the terminal cost with a symmetric, positive-definite final Riccati matrix $P(T) \in \mathbb{R}^{n\times n} \succ 0$, and $M \in \mathbb{R}^{n\times n} \succeq 0$, $R \in \mathbb{R}^{m\times m} \succ 0$ are user-defined matrices to penalize the state and control input, respectively. Our goal is to obtain the optimal control $u_i^\star(x, t)$ such that $J(x_{i,0}, u_i^\star; t_0, T) \leq J(x_{i,0}, u_i; t_0, T)$ is satisfied for all x_i. To this end, we formulate the minimization problem $J(x_{i,0}, u_i^\star; t_0, T) = \min_{u_i} J(x_{i,0}, u_i; t_0, T)$ subject to (1). Subsequently, the value function gets the form of,

$$V(x_i; t_0, T) = \min_{u_i \in U}\left[\phi(T) + \frac{1}{2}\int_{t_0}^{T} \left(x_i^\top M x_i + u_i^\top R u_i\right) d\tau\right]. \tag{3}$$

Assumption 1. The matrix pair (A, B) is controllable and the pair ($M^{1/2}$, A) is detectable. ■

In addition, consider an agent's closed dynamic obstacle space as $X_{\mathrm{obs},i} \subset X$. The free space of the environment $X_{\mathrm{free},i}$, conversely, is defined as the complement of the obstacle space $X_{\mathrm{free},i} = X \setminus X_{\mathrm{obs},i}$. In a dynamic environment, both the obstacle space and free space evolve in time. Define $\dot{X}_{\mathrm{obs},i} := f(X_{\mathrm{obs},i}, t)$ as the variation of the obstacle space in the environment, where $f(\cdot)$ is unknown.

To minimize its cost, the agent will efficiently search the environment by randomly constructing a space-filling tree, and use it to find its global path $\sigma_i(x_{i,0,k}, x_{i,\mathrm{g},k}, t) \in \mathbb{R}^{2(K\times n)}$ for $k = 1, \ldots, K$ where $K \in \mathbb{N}$ is the number of boundary value problems (BVPs) in the path. The path $\sigma_i(x_{i,0,k}, x_{i,\mathrm{g},k})$ includes the initial states $X_{i,0} = x_{i,0,k}$ for all i, where $X_{i,0} \in \mathbb{R}^{K\times n} \subset X_{\mathrm{free},i}$, as well as the goal states $X_{i,\mathrm{g}} = x_{i,\mathrm{g},k}$ for all i, where $X_{i,\mathrm{g}} \in \mathbb{R}^{K\times n} \subset X_{\mathrm{free},i}$. As the obstacle space $X_{\mathrm{obs},i}$ evolves in time, σ_i adapts using a goal-to-start replanning, and K changes accordingly. RRT$^{\mathrm{X}}$ also provides an initial graph $G_i = (V_i, E_i)$, where V_i is the initial set of nodes, and E_i is the initial set of edges. The global path σ_i in the graph is given by $T_i = (V_{T,i}, E_{T,i}) \subseteq G_i$.

We shall next connect the game-theoretic formulation to the motion planning problem. For each BVP provided by RRT$^{\mathrm{X}}$ for an agent i, we seek to drive the system to the goal state. For the k-th BVP, define the *initial distance* as the distance from the initial state $x_{i,0,k}$ and goal state $x_{i,\mathrm{g},k}$,

$$D_0(x_{i,0,k}) := \|x_{i,0,k} - x_{i,\mathrm{g},k}\| = \|x_{i,0,k}\|, \forall x_{i,0} \in \mathbb{R}^n, \tag{4}$$

and the *relative distance* as,

$$D(x_i) := \|x_i - x_{i,\mathrm{g},k}\| = \|x_i\|, \forall x_{i,0} \in \mathbb{R}^n.$$

Since the game-theoretic problem utilizes a *free-final state*, $x_i(T)$ will approximate the desired state $x_{i,\mathrm{g}}$ to reduce the total navigation time [19]. We assume that $x_{i,\mathrm{g}}$ is reached when the agent reaches the close neighborhood around $x_{i,\mathrm{g}}$. In other words, the agent is considered to have reached its final state when $D(x_i) \leq \epsilon D_0(x_{i,0,k})$ where $\epsilon \in \mathbb{R}$ is a user-defined window to determine the neighborhood. Upon reaching this goal state neighborhood, the agent continues to its $(k+1)$-th BVP.

When RRT$^{\mathrm{X}}$ calculates a collision-free path σ_i, it selects only straight lines as edges in the set E_i. However, the robot's true trajectory is subject to both kinodynamic constraints (1) and the optimal performance constraints (2). The actual trajectory thus deviates from the chosen path σ_i, which may result in collisions when the agent i passes closely by obstacles. We reduce this risk by adopting an obstacle augmentation strategy. Let us define the KD as,

$$D_{\mathrm{rob},i}(x_i) := \frac{|x_{i,0,k} \times x_i|}{D_{i,0,k}}, \tag{5}$$

to find the difference between the agent's current location and the straight path determined by the pair $(x_{i,0,k}, x_{i,\mathrm{g},k})$. We then use this distance to form an augmented obstacle space $X^{\mathrm{aug}}_{\mathrm{obs},i}$ based on the maximal KD $D^{\mathrm{kin}}_{\mathrm{rob},i}$,

$$X^{\mathrm{aug}}_{\mathrm{obs},i} := X_{\mathrm{obs},i} \cup X_{\mathrm{kin},i} \tag{6}$$

where $X_{\mathrm{kin},i}$ is the space of a compact set bounded by a circle centered on the origin with a radius of $R^{\mathrm{kin}}_{\mathrm{rob},i}$. As $R^{\mathrm{kin}}_{\mathrm{rob},i}$ is updated, the augmented obstacle space $X^{\mathrm{aug}}_{\mathrm{obs},i}$ is recalculated, and RRT$^{\mathrm{X}}$ will accordingly plan a collision-free path further from each obstacle with all newly-invalid nodes and their descendants pruned.

Each agent is operating without knowledge of the other agents' planned motions, and instead limited to its own observations of other agents' states. Therefore, each agent's motion planning is a problem with bounded rationality. Each agent i's knowledge of the environment is limited to the location of obstacles and other agents within a perception radius r around $x_i(t)$ and constructs an individual obstacle space $X_{\mathrm{obs},i}(t)$ using the locations of perceived obstacles as well as the other perceived agents. To ensure optimal motion planning and to mitigate the risk of colliding with the other agents making up its obstacle space, each agent also forms a predicted obstacle space $\hat{X}_{\mathrm{obs},i}$ which is added to the obstacle space, considered by the motion planning algorithm, to form $X^{\text{tot}}_{\text{obs};i} = X^{\text{aug}}_{\text{obs};i} \cup \hat{X}_{\text{obs};i}$. This is the obstacle space ultimately used to construct a path using RRT^{X}. To form this predicted obstacle space, each agent adopts a level-k rationality cognitive hierarchy approach to anticipate the motion of the other agents.

Definition 1. *Level-k rationality* is a cognitive hierarchy model of strategies where an agent using a level-k strategy assumes all other agents employ a level-($k-1$). $\square$

Our goal is to describe the potential levels of rationality present in each agent in the environment, and then to develop an algorithm to reliably estimate the level of each agent in the environment by observing its motion.

III. Model-Free Formulation

The Hamiltonian associated with (1) and (3) is,

$$H\left(x_i, u_i, \frac{\partial V^\star}{\partial x}, \frac{\partial V^\star}{\partial t}\right) := \frac{1}{2}\left(x_i^\top M x_i + u_i^\top R u_i\right) + \frac{\partial V^{\star\top}}{\partial x}\left(Ax_i + Bu_i\right) + \frac{\partial V^\star}{\partial t}, \quad \forall x_i, u_i.$$

Since the system (1) is linear, the optimal value function can be written in the form of,

$$V^\star(x_i; t) = (1/2)x_i^\top P(t) x_i, \tag{7}$$

where $P(t) \in \mathbb{R}^{n\times n} > 0$ is the symmetric positive-definite Riccati matrix calculated by,

$$\dot{P}(t) = -P(t)A - A^\top P(t) - M + P(t)BR^{-1}B^\top P(t). \tag{8}$$

Therefore, the optimal control is computed as,

$$u_i^\star(x_i; t) = -R^{-1}B^\top P(t) x_i, \quad \forall x_i, t.$$

Let us now define the Q-function for an agent i as

$$\begin{aligned} Q_i(x_i, u_i; t) := {} & V^\star(x_i; t) + \frac{1}{2}\left(x_i^\top M x_i + u_i^\top R u_i\right) \\ & + x_i^\top P(t)(Ax_i + Bu_i) + \frac{1}{2}x_i^\top \dot{P}(t) x_i, \end{aligned} \tag{9}$$

where $Q_i(x_i, u_i; t) \in \mathbb{R}$ is an action-dependent value. We next define the augmented state $U_i := [x_i^\top \ u_i^\top]^\top \in \mathbb{R}^{(n+m)}$ to express (9) in compact form as,

$$Q_i(x_i, u_i; t) = \frac{1}{2}U_i^\top \begin{bmatrix} Q_{\text{xx}}(t) & Q_{\text{xu}}(t) \\ Q_{\text{ux}}(t) & Q_{\text{uu}} \end{bmatrix} U_i := \frac{1}{2}U_i^\top Q_i(t) U_i, \tag{10}$$

where $Q_{\text{xx}}(t) = \dot{P}(t) + P(t) + M + P(t)A + A^\top P(t)$, $Q_{\text{xu}}(t) = Q_{\text{ux}}(t) = P(t)B$, and $Q_{\text{uu}} = R$, with $Q_i : \mathbb{R}^{n+m} \times \mathbb{R}^{(n+m)\times(n+m)} \to \mathbb{R}$. By using the stationarity condition $\partial Q_i(x_i, u_i; t)/\partial u_i = 0$, we find a model-free expression of the optimal control $u_i^\star$ as, $u_i^\star(x_i; t) = \arg\min_{u_i} Q_i(x_i, u_i; t) = -Q_{\text{uu}}^{-1} Q_{\text{ux}}(t) x_i$.

Lemma 1. *The minimization problem $Q_i^\star(x_i, u_i^\star; t) := \min_{u_i} Q_i(x_i, u_i; t)$ results the same solution to* (3) *$Q_i^\star(x_i, u_i^\star; t) = V^\star(x_i; t)$ from* (7)*, where $P(t) > 0$* (8).

Proof. The proof follows from [15]. ■

Each agent shall use an actor/critic structure in order to approximate its optimal control for each BVP. The structure used is described in detail in Section IV of [15].

IV. Cognitive Hierarchy And Motion Planning Framework

A. Levels of rationality

Let us consider the scenario where the agents navigate in a bounded space X with no perfect rationality. To this end, we consider the cognitive hierarchy theory of "level-k" reasoning (Definition 1). Under this framework, each agent is assigned an individual time-invariant rationality level of k. An agent operating with level-k reasoning assumes that every other agent in the environment operates at level-($k-1$). By predicting the strategies resulting from different levels of rationality and observing the motions of other agents, each agent i forms a predicted obstacle space at time t, $\hat{X}_{\text{obs};i}(t)$. This predicted obstacle space is incorporated into the agent's total perceived obstacle space used for motion planning. To determine the levels of rationality, we presume each agent in the environment seeks to minimize its cost-to-go to its individual goal state while avoiding collisions.

B. Level-0 Policy

To describe higher levels of rationality, we find the level-0, or "anchor," policy. The anchor can be defined as either a random approach or a naive approach where the agent is unable to detect any other agents. As random navigation is often ineffective, we consider the naive approach. A level-0 agent i will ignore the other agents in the environment, and construct the obstacle space $X_{\text{obs};i}$ using solely perceived non-agent obstacles. In other words, $\hat{X}_{\text{obs};i} = \{\emptyset\}$. Then, the agent plans its motion to its goal state using RRT^{X} which seeks the optimal path according to,

$$\pi_i^\star(x_{i;0}, x_{i;\text{g}}) = \min_{(x_{i;0}, x_{i;\text{g}}) \in X_{\text{free};i}} d\left(\pi(x_{i;0}, x_{i;\text{g}})\right), \tag{11}$$

constrained by the dynamics (1), where $d(\pi)$ is the length of the path between $x_{i;0}$ and $x_{i;\text{g}}$. An example of the predicted behavior of a level-0 agent i in a multi-agent environment with no other obstacles is shown in Fig. 1-(a).

C. Level-1 Policy

Similarly to a level-0 agent, a level-1 agent j traversing the environment seeks to drive to its goal state by constructing its obstacle space $X_{\text{obs};j}$, and conducting online motion planning to find the optimal path $\pi^\star(x_{i;0}, x_{i;\text{g}})$ using RRT^{X}. Level-1 agents additionally seek to predict the motion of other agents in the environment, and use this to construct the predicted obstacle space $\hat{X}_{\text{obs};j}(t)$ to form their total obstacle space $X^{\text{tot}}_{\text{obs};j}$. In this case, the agent i anticipates level-0 behavior from all other agents. A level-1 agent is not aware of the path that a level-0 agent may be following, as the bounded rationality of the problem dictates that no agent knows another's current or future goal states. Thus, the level-1 agent will instead need to avoid collision by avoiding all possible collisions with level-0 agents.

Consider the distance between a level-0 agent i and level-1 agent j as $D_{ij}(x_i, x_j) = ||x_i - x_j||$. Let us define a collision between i and j as occurring at time t if $D_{ij}(x_i(t), x_j(t)) \le d_{\text{col}}$, where $d_{\text{col}} \in \mathbb{R}^+$ is a user-defined distance based

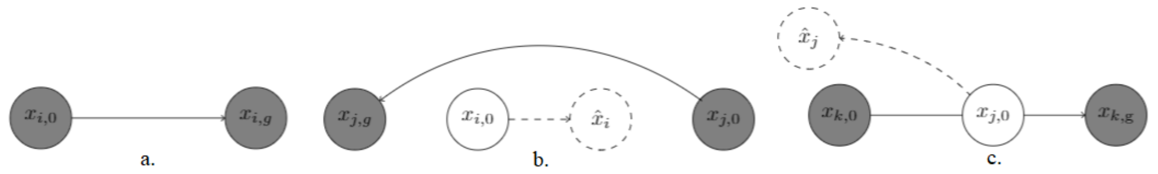

Fig. 1. Increasing levels of rationality of agents. (a) A level-0 agent i, travelling optimally (directly) to its goal. (b) A level-1 agent j travelling to its goal while avoiding an agent i. (c) A level-2 agent k travelling to its goal while presuming that another agent j will avoid a possible collision.

on the radii of the agents to represent when the agents have physically collided. We assume that agent j is familiar with the kinodynamic constraints of agent i, and with the kinodynamic constraints present on agent i. Hence, agent j constructs the potential future state space $\hat{X}_i$, which consists of all states agent i can reach within a given time-frame t_s. Agent i then generates obstacles over this space, augments them to account for the maximal KD $D^{\text{kin}}_{\text{rob};j}$, incorporates them into the predicted obstacle space $\hat{X}_{\text{obs};j}$, and finds the optimal policy,

$$\pi^\star_j(x_{j;0}, x_{j;\text{g}}) = \min_{(x_{j;0},x_{j;\text{g}})\in(X_{\text{free};j}\setminus\hat{X}_{\text{obs};j})} d(x_{j;0}, x_{j;\text{g}}),$$

constrained by (1). This space is rapidly updated by j as i traverses the environment to ensure its accuracy.

Theorem 1. *Consider a level-1 agent j that is familiar with the kinodynamic constraints of a different agent i. In addition, the level-1 agent j can observe that agent's i state, velocity, and trajectory. Then, agent j can plan a motion that is guaranteed to avoid a collision with i.*

Proof. Consider a collision between the agents i and j. A collision necessitates that at some time t, $D(x_i(t), x_j(t)) \le d_\text{col}$. However, because of the predicted obstacle space of agent j $\hat{X}_{\text{obs};j}$, the free space of agent j $X_{\text{free};j}$ contains no potential states of j $\hat{x}_j$ such that $D(\hat{X}_i, \hat{x}_j) \le d_\text{col}$, and therefore j's motion planning will not enter these points. Thus, the theorem is true by contradiction. ■

Following Theorem 1, the level-1 agent plans an asymptotically optimal path minimizing its path length through the environment while ensuring that it safely avoids all other agents present. An example of this is shown in Fig. 1-(b).

D. Higher Level Policies

A level-2 agent k assumes level-1 behavior from all other agents, and must choose its predicted obstacle space $\hat{X}_{\text{obs};k}$ such that it avoids all collisions with these agents. It then seeks to find the optimal path $\pi^\star(x_{k;0}, x_{k;\text{g}})$ similarly to agents of a lower level. However, Theorem 1 states that if all other agents in the environment are level-1 agents, then their motion planning will avoid collisions regardless of the actions of agent k. Thus, to find the optimal possible path, the agent k chooses $\hat{X}_{\text{obs};k} = \{\emptyset\}$ in order to maximize $X_{\text{free};k}$. It then finds the optimal policy,

$$\pi^\star_k(x_{k;0}, x_{k;\text{g}}) = \min_{(x_{k;0},x_{k;\text{g}})\in X_{\text{free};k}} d(x_{k;0}, x_{k;\text{g}}).$$

Theorem 2. *Consider a level-0 agent i and level-2 agent k placed in identical environments with a shared initial state and goal state x_0 and x_g, respectively. Then, both agents have the same optimal path.*

Proof. Consider the optimal path of agents i and k, where $\pi^\star_i \neq \pi^\star_k$. This implies that $\min_{(x_0,x_\text{g})\in X_{\text{free};i}} d(x_0, x_\text{g}) \neq \min_{(x_0,x_\text{g})\in X_{\text{free};k}} d(x_0, x_\text{g})$. However, as $X^{\text{aug}}_{\text{obs};i} = X^{\text{aug}}_{\text{obs};k}$, then $X_{\text{free};i} = X_{\text{free};k}$. Thus, there exists no optimal path $\pi^\star$ such that $\pi^\star \in X_{\text{free};i}, \notin X_{\text{free};k}$ or vice versa. Therefore, the theorem is true by contradiction. ■

An example of level-2 path planning is shown in the third image of Fig. 1. The level-2 policy being identical to the level-0 policy also indicates that the level-3 policy is identical to the level-1 policy. This alternating pattern repeats for all higher levels of rationality. Therefore, to model all levels of rationality of each agent, we only need to consider two levels: level-0 and level-1.

E. Motion Planning Framework

The motion planning structure consists of five stages: i) dynamic planning with RRT$^\text{X}$; ii) Q-learning; iii) terminal state evaluation; iv) obstacle augmentation; and v) predictive obstacle avoidance. The four stages (i)–(iv) are similar to those used in RRT-Q$^\text{X}$. The key difference in the implementation is the fifth stage, where the agent, is operating with an appropriate level of rationality, incorporates the potential motion of other agents into its obstacle space. This adjusted implementation is shown in Algorithm 1.

V. Level of Rationality Estimation

We next propose a framework to estimate the level of rationality of each agent in an environment. We consider an environment containing N agents, each driven with the proposed framework with either level-0 or level-1 rationality. Note that, all higher levels can be expressed by level-0 and level-1. In addition, we consider an algorithm observing the environment that is only familiar with each agent's state, velocity, and trajectory at any time t.

In order to identify the level of rationality of each agent, we consider the tendencies of each level of rationality. Since level-1 agents react online to the potential motion of agents to avoid collisions, they need to often rapidly replan in congested environments. Conversely, level-0 agents rarely need to replan their trajectory. Considering this, the algorithm first seeks to estimate the series of BVPs that each agent i followed through the environment as part of its path π_i. Then, agent i compares the maximum KD during

Algorithm 1 Bounded Rational 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; $\mathcal{X}_{\text{obs}}$ - obstacle space; $\mathcal{X}_{\text{a}}$ - states of other agents; $\mathcal{X}$ - state space; L - level of rationality; t$_{\text{s}}$ - agent range time-frame
Output: $\hat{u}$ - control

```
1: $\alpha_{\text{a}}, \alpha_{\text{c}} \leftarrow$ Stability($M, R$)
2: $\mathcal{X}_{\text{obs}}^{\text{aug}} \leftarrow \mathcal{X}_{\text{obs}}$;
3: $D_{\text{rob}}, D_{\text{rob}}^{\text{kin}} \leftarrow 0$; $k \leftarrow 1$;
4: while $x_{\text{goal}} \neq x$ do
5:     if $L = 1$ then                    ▷ Predictive obstacle avoidance
6:         $\hat{\mathcal{X}}_{\text{obs}} \leftarrow$ PotentialStates($\mathcal{X}_a, L, t_s$);
7:         $\mathcal{X}_{\text{obs}}^{\text{tot}} \leftarrow \mathcal{X}_{\text{obs}} + \hat{\mathcal{X}}_{\text{obs}}$
8:     end if
9:     while NoCollision do
10:        $D_0 \leftarrow$InitialDistance($x_0$) (4);
11:        for $t \in T$ do
12:            if $D_{\text{rob}} > D_{\text{rob}}^{\text{kin}}$ then        ▷ Obstacle augmentation
13:                $D_{\text{rob}}^{\text{kin}} \leftarrow D_{\text{rob}}$;
14:                $\mathcal{X}_{\text{obs}}^{\text{aug}} \leftarrow$ Augment($\mathcal{X}_{\text{obs}}^{\text{tot}}, D_{\text{rob}}^{\text{kin}}$) (6);
15:            end if
                                                      ▷ Q-learning
16:            $\hat{W}_{\text{c}} \leftarrow$ Critic($M, R, \Delta t, \alpha_{\text{c}}, \bar{x}, \hat{u}$)
17:            $\hat{\mathcal{Q}}_i \leftarrow$ EstimateQ($\hat{W}_{\text{c}}, \bar{x}, \hat{u}$)
18:            $\hat{W}_{\text{a}} \leftarrow$ Actor($\hat{\mathcal{Q}}_i, \alpha_{\text{a}}, \bar{x}$)
19:            $\hat{u} \leftarrow$ Control($\hat{W}_{\text{a}}, \bar{x}$)
20:            return $\hat{u}$;
21:            $D_{\text{rob}} \leftarrow$ KinodynamicDist($x_{0,k}, \bar{x}, D_0$) (5);
22:            if $D \leqslant \rho D_0$ then          ▷ Terminal state evaluation
23:                $x_{0,k} \leftarrow x(t)$;
24:                $k \leftarrow k + 1$;
25:                break;
26:            end if
27:        end for
28:    end while                                  ▷ Dynamic planning
29:    $\mathcal{G}, \pi \leftarrow$ RRT$^{\text{X}}$($\mathcal{X}, \mathcal{X}_{\text{obs}}^{\text{aug}}, x_{\text{start}}, x_{\text{goal}}$);
30: end while
```

each of these BVPs with the expected maximum KD. If the measured maximum KD of agent j is larger than the predicted maximum KD, then agent i needs to significantly adjust its planned path online, which in turn implies that agent i has level-1 rationality. Conversely, if this never occurs, it implies that i is level-0.

A. BVP Estimation

As previously mentioned, an agent i in the environment constructs a global path composed of a series of BVPs $_i(x_{i,0,k}, x_{i,\mathrm{g},k}, t) \in \mathbb{R}^{2(K\times n)}$ for $k \quad 1, \quad , K$. As the agent traverses the k-th BVP of the path, it moves to the $k \quad 1$-th BVP after entering the pre-defined neighborhood of the goal state $x_{i,\mathrm{g},k}$. As the level estimation algorithm has no information about each agent's determined path, we seek to estimate the start and end location of each BVP to estimate the path. Moreover, since each BVP is a straight line, an agent is considered to be travelling on one BVP so long as its trajectory does not significantly change over time. In addition, BVP endpoints can be estimated by observing where the agent's trajectory changes over a sufficiently short period of time. To this end, we mark the initial state $x_{i,0}$ as the first BVP endpoint. We then denote the trajectory of agent i at time t as $v_i(t) \in \mathbb{R}^n$. We observe the agent online, and note the change of its trajectory over a short time frame t_a, and calculate the angle of change,

$$_i(t) \quad \cos^{-1} \frac{v_i(t) \quad v_i(t \quad t_a)}{||v_i(t)|| \, ||v_i(t \quad t_a)||}. \tag{12}$$

If the angle of change is found to exceed a user-chosen value , then $x_i(t)$ is marked as a BVP endpoint. Note that must be sufficiently large to avoid falsely interpreting an agent's slight trajectory adjustments due to kinodynamic constraints as a BVP endpoint. Increasing this threshold also means that some BVP endpoints will be missed, but the level estimation algorithm is searching primarily for endpoints resulting from online replanning to avoid new obstacles, which will result in significant trajectory changes.

After finding these endpoints, we measure the KD of the agent over each BVP, and compare it to the expected KD of the BVP. The expected KD of an agent over a BVP increases with the length of the BVP, and therefore if the measured KD exceeds the expected KD, it implies that the estimated BVP is shorter than it was originally planned to be. This in turn implies that the agent's trajectory significantly adjusted from what was originally expected, and that the agent in question is avoiding collisions and therefore is a level-1 agent. If this never occurs, it implies that the path was not significantly altered, and the agent in question is a level-0 agent.

VI. Simulation Results

For our simulation, we consider four agents placed in an environment with several environmental obstacles. Each agent is represented by the system (1), with plant and input matrices identical to those used in the simulation in [6] with $\mathbf{x} \quad [x_i \; y_i \; \dot{x}_i \; \dot{y}_i]$ as the state. The translations of each agent i are denoted as x_i, y_i, the velocities as $\dot{x}_i$, $\dot{y}_i$, and the accelerations as $\ddot{x}_i$, $\ddot{y}_i$. The inputs are forces denoted as f_1, f_2. We choose the finite horizon $T \quad 10\,\mathrm{s}$, and the admissible window $\quad 0.9$. The user-defined matrices for the cost function are $M \quad 10 I_4$, and $R \quad 2 I_2$. For the level estimation algorithm, we choose the BVP angle threshold $\quad 18^\circ$, the angle measurement time frame $t_a \quad 0.2\mathrm{s}$, and the KD error threshold $d \quad 0.05$. The critic and actor gains are set as $_{\mathrm{c}} \quad 90$ and $_{\mathrm{a}} \quad 1.2$.

Two simulations at different time frames are depicted in Fig. 2 and Fig. 3. We choose one agent to operate at our proposed level-1 rationality (shown in green), and the rest agents at level-0 (shown in gold). The generating artificial obstacles are shown in blacks squares, We then assign to each agent an initial state and a goal state in the environment, and allow our level estimation algorithm to determine the level of rationality of each agent in the environment. We repeated this experiment with a variety of initial and goal states for each agent. Over these simulations, we found that all agents were able to consistently avoid collision with the other agents. Our level estimation algorithm was also able to very consistently recognize the level-1 agent regardless of its starting position, as well as correctly recognize the level-0