RRT-QX: Real-Time Kinodynamic Motion Planning in Dynamic Environments with Continuous-Time Reinforcement Learning

George P. Kontoudis[a], Kyriakos G. Vamvoudakis[b], and Zirui Xu[c]

[a]Maryland Robotics Center, University of Maryland, College Park, MD, USA;
[b]Guggenheim Sch. of Aerospace Eng., Georgia Institute of Technology, Atlanta, GA, USA;
[c]Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, USA

ARTICLE HISTORY
Compiled May 2022

ABSTRACT
This chapter presents a real-time kinodynamic motion planning technique for linear systems with completely unknown dynamics in environments with unpredictable obstacles. The methodology incorporates: i) a sampling-based algorithm for path planning and fast replanning; and ii) continuous-time Q-learning for the solution of finite-horizon optimal control problems in real-time. The path planner produces a set of waypoints that dynamically change in time according to the unpredictably appearing obstacles, while the Q-learning controller is responsible for optimal waypoint navigation. The efficacy of the methodology has been validated with simulations.

1. Introduction

Substantial improvements in artificial intelligence, computing resources, and software tools have enabled tremendous capabilities to mobile robots and autonomous systems. The problem of navigation is a core topic in robotics and autonomous vehicles, as the majority of robotic applications require safe path planning and obstacle avoidance (Yang et-al., 2018). Ideally, a solution to this problem considers collision-free navigation in dynamic environments, computationally affordable algorithms for real-time implementation, and optimal control strategies. Such a challenging problem should be addressed in the continuous-time domain, as naive discretization of the system dynamics and of the policy space disregards critical information and leads to discretization errors (Lillicrap et al., 2015). In addition, dynamic environments impose time constraints to the motion planning problem, because collision-free navigation is only ensured for limited time frames (Nägeli, Alonso-Mora, Domahidi, Rus, & Hilliges, 2017). The latter necessitates a finite-horizon formulation to the optimal control problem. Moreover, system modeling is a challenging task with inevitable model simplifications and inaccuracies (Berkenkamp & Schoellig, 2015). Thus, a combination of optimal and adaptive control is needed. Finally, even if the system dynamics assumed to be known, the finite-horizon optimal control problem requires extensive offline computations to solve the Hamilton-Jacobi-Bellman equation (Lewis, Vrabie, & Syrmos, 2012).

Our aim in this work is to present a real-time kinodynamic motion planning technique for dynamic environments with unpredictably appearing obstacles. We address

CONTACT G. P. Kontoudis. Email: kont@umd.edu

the finite-horizon optimal control problem with completely unknown system dynamics. The unknown model is considered to be continuous-time linear time-invariant.

Motion planning in high-dimensions has been addressed with probabilistic roadmaps (PRM) (Kavraki, Svestka, Latombe, & Overmars, 1996) and rapidly-exploring random trees (RRT) (Kuffner & LaValle, 2000; LaValle, 1998). These algorithms are probabilistically complete, but not optimal. The work of Karaman & Frazzoli (2011) proposed a variation of RRT based on rewiring, namely RRT*. The latter was proved to probabilistically complete and asymptotically optimal. These methods do not incorporate realistic system dynamics and instead use simple dynamics.

The problem of kinodynamic motion planning is introduced in (Donald, Xavier, Canny, & Reif, 1993). Kinodynamic RRT (LaValle & Kuffner, 2001) employs the dynamical model of the system, but the proposed control strategy is selected either randomly or by testing multiple controls and selecting the best. LQR-trees (Tedrake, Manchester, Tobenkin, & Roberts, 2010) is a feedback motion planning algorithm that utilizes optimization tools. This method requires significant computations to solve the Riccati equation. A combination of linear quadratic regulator (LQR) and RRT* is proposed in (Perez, Platt, Konidaris, Kaelbling, & Lozano-Perez, 2010). In particular, the authors formulate a free-final-state, infinite-horizon optimal control problem with minimum energy cost and a heuristic extension of the RRT*. This algorithm incorporates the system dynamics and enforces extensive offline computations. Kinodynamic RRT* (Webb & Van Den Berg, 2013) is an asymptotically optimal motion planner for known linear time-invariant systems. The authors formulate a finite-horizon optimal control problem of fixed-final-state and free-final-time with minimum fuel-time performance. However, kinodynamic RRT* yields an open-loop controller and the computation of the continuous reachability Gramian requires significant offline computation. The authors in (Li,Cui, Li, & Xu, 2018) proposed a near optimal kinodynamic motion planning technique, that is named NoD-RRT. The methodology utilizes neural network approximation and RRT. NoD-RRT achieved reduced computational complexity and enhanced performance for nonlinear systems, comparing to RRT and kinodynamic RRT*. Yet, their framework is model-based and requires offline computations. In (Kontoudis & Vamvoudakis, 2019a), the authors presented RRT-Q*, an online, model-free kinodynamic motion planning framework which computes approximately optimal control policies for motion planning in static environments. RRT-Q* combines continuous-time Q-learning, RRT*, and local replanning in relatively small spaces. The latter has been robustified in (Kontoudis & Vamvoudakis, 2019b). In (Chiang, Hsu, Fiser, Tapia, & Faust, 2019), the authors combined reinforcement learning (RL) and RRT for kinodynamic motion planning. RL is used to learn obstacle avoiding policies and supervised learning to predict the time to reach a state and guide the growth of the tree. However, this algorithm requires significant offline computations. All the aforementioned motion planning techniques can only deal with static environments.

Randomized kinodynamic motion planning in dynamic environments is introduced in (Hsu, Kindel, Latombe, & Rock, 2002). In (Bruce & Veloso, 2002), the execution-extended RRT is presented for real-time replanning. This algorithm stores selective nodes in a waypoint cache and performs an iterative, adaptive cost search on the forward tree, towards efficient replanning in dynamic domains. Dynamic RRT (Ferguson, Kalra, & Stentz, 2006) trims the invalid leafs of the tree when a collision occurs due to new obstacle configuration and grows the rest tree from goal to the current configuration of the robot. The authors in (Otte & Frazzoli, 2014, 2016) present the RRT$^{\text{X}}$, an asymptotically optimal motion planning algorithm for both static and dynamic environments. RRT$^{\text{X}}$ has the ability to perform quick online replanning. In (Allen &

Pavone, 2019), the authors proposed an online kinodynamic motion planning algorithm which was experimentally validated in dynamic indoor environments. The technique requires the model of the system and its efficiency depends on the offline training of reachability sets. Kontoudis, Xu, & Vamvoudakis (2020) combined event-triggered Q-learning and RRT$^{\text{X}}$ to address the kinodynamic motion planning problem in dynamic domains. In addition, cognitive hierarchy along with RRT$^{\text{X}}$ and Q-learning has been used to multi-robot motion planning in human-crowded environments Netter, Kontoudis, & Vamvoudakis (2021).

Optimal control (Lewis, Vrabie, & Syrmos, 2012) can be efficiently merged with adaptive control (Ioannou & Sun, 2012) by employing principles of reinforcement learning (Sutton, & Barto, 2018), and approximate dynamic programming (Busoniu, Babuska, De Schutter, & Ernst, 2010; Lewis, Vrabie, & Vamvoudakis, 2012; Powell, 2007; Vrabie, Vamvoudakis, & Lewis, 2013). In (Watkins & Dayan, 1992), a solution to Markovian systems was proposed with the use of discrete-time Q-learning. The authors in (Mehta & Meyn, 2009), presented a connection of Q-learning with nonlinear control based on the observation that the Q-function is related to the Hamiltonian. A solution to the model-free, infinite horizon optimal control problem for continuous-time linear time-invariant systems is presented in (Vamvoudakis, 2017).

The remainder of this paper is organized as follows. In Section 2 we formulate the problem, Section 3 discusses the optimal control problem, Section 4 provides a model-free formulation based on Q-learning, and Section 5 presents the structure of RRT-Q*. Section 6 illustartes the efficiency of our method through simulations and Section 7 concludes the chapter.

The notation here is standard. The set of real numbers is denoted $\mathbb{R}$, the set of all positive real numbers $\mathbb{R}^{+}$, the set of $n \times m$ real matrices $\mathbb{R}^{n \times m}$, and the set of natural numbers $\mathbb{N}$. The notation $(\cdot)^{\intercal}$ and $(\cdot)^{-1}$ denote the transpose and inverse operator respectively. The superscript $\star$ denotes the optimal solutions of a minimization problem. The notations $\underline{\lambda}(A)$ and $\overline{\lambda}(A)$ denote the minimum and maximum eigenvalues of the matrix A respectively. We denote $\text{vech}(A)$, $\text{vec}(A)$, and $\text{mat}(A)$ the half-vectorization, vectorization, and matrization of a matrix A respectively. The Minkowski sum of two sets is denoted $\oplus$. A positive and semi-positive definite A matrix is denoted by $A \succ 0$ and $A \succeq 0$ respectively. The notation $U \otimes V$ denotes the Kronecker product of two vectors and $\| \cdot \|$ denotes the L_2 norm.

2. Problem Formulation

Let a linear time-invariant continuous-time system,

$$\dot{x}(t) = Ax(t) + Bu(t),\ x(0) = x_0,\ t \geq 0,$$

where $x(t) \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state vector, $u(t) \in \mathbb{R}^m$ is the control input, and $A \in \mathbb{R}^{n \times n}$, $B \in \mathbb{R}^{n \times m}$ are the unknown plant and input matrix respectively. To drive our system from an initial state x_0 to a final state $x(T) = x_{\text{r}}$, we define the difference between the state $x(t)$ and the state x_{r}, as our new state $\bar{x}(t) \coloneqq x(t) - x_{\text{r}}$. The final time is denoted by $T \in \mathbb{R}^{+}$. The new system yields,

$$\dot{\bar{x}}(t) = A\bar{x}(t) + Bu(t),\ \bar{x}_0 = x_0 - x_{\text{r}},\ t \geq 0. \tag{1}$$

Consider an energy cost function,

$$J(\bar{x}; u; t_0, T) = \phi(T) + \frac{1}{2}\int_{t_0}^{T} (\bar{x}^\intercal M \bar{x} + u^\intercal R u)\, \mathrm{d}\tau,\ \forall t_0, \tag{2}$$

where $\phi(T) = \frac{1}{2}\bar{x}^\intercal(T)P(T)\bar{x}(T)$ is the terminal cost with $P(T) = P_\mathrm{T} \in \mathbb{R}^{n\times n} \succ 0$ the final Riccati matrix, $M \in \mathbb{R}^{n\times n} \succeq 0$ and $R \in \mathbb{R}^{m\times m} \succ 0$ user defined matrices that penalize the states and the control input respectively.

Assumption 1. The unknown pairs (A,B) and ($\sqrt{M}$, A) are controllable and detectable respectively.

We are interested in finding an optimal control $u^\star$ such that it satisfies $J(\bar{x}; u^\star; t_0, T) \leq J(\bar{x}; u; t_0, T),\ \forall \bar{x}, u$, which can be described by the minimization problem $J(\bar{x}; u^\star; t_0, T) = \min_u J(\bar{x}; u; t_0, T)$ subject to (1). In other words, we want to obtain the optimal value function $V^\star$ that is defined by,

$$V^\star(\bar{x}; t_0, T) \coloneqq \min_u \left(\phi(T) + \frac{1}{2}\int_{t_0}^{T} (\bar{x}^\intercal M \bar{x} + u^\intercal R u)\, \mathrm{d}\tau\right), \tag{3}$$

but without any information about the system dynamics.

Consider the known obstacle closed space $\mathcal{X}_\mathrm{obs} \subset \mathcal{X}$. For multiple obstacles, the obstacle space is defined $\mathcal{X}_\mathrm{obs} \coloneqq \bigcup_{l=1}^{N_o} \mathcal{X}_{\mathrm{obs},l}$, where $N_o \in \mathbb{N}$ is the total number of obstacles. Thus, the free space is an open space $\mathcal{X}_\mathrm{free} = (\mathcal{X}_\mathrm{obs})^\complement = \mathcal{X}\backslash\mathcal{X}_\mathrm{obs}$. In dynamic environments, the obstacle space $\mathcal{X}_\mathrm{obs}$ and the free space $\mathcal{X}_\mathrm{free}$ evolve in time. We define the unpredictable variation of the obstacle space as $\Delta\mathcal{X}_\mathrm{obs} \coloneqq f(\mathcal{X}_\mathrm{obs}; t)$, where $f(\cdot)$ is unknown, and $\Delta\mathcal{X}_\mathrm{obs} = \{\varnothing\}$ indicates no obstacle changes in the environment. We use RRT$^\mathrm{X}$ for path planning. RRT$^\mathrm{X}$ constructs a graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. As a slight abuse of notation, we will refer to nodes $v \in V$ as states $x \in \mathcal{X}$. The planner provides an optimal sub-tree that contains the planned path $\pi(x_{0,k}, x_{\mathrm{r},k}; t) \in \mathbb{R}^{2(K\times n)}$, where $k = 1, \ldots, K$, $K \in \mathbb{N}$ is the number of boundary value problems (BVPs). Each BVP is described by the initial and desired state $(x_{0,k}, x_{\mathrm{r},k})$. Since the obstacle space $\mathcal{X}_\mathrm{obs}$ evolves in time, π is also a function of time, and thus K also change accordingly.

We seek to drive the system to a desired state, without any knowledge of the system dynamics. For the k-th BVP, let us define the initial distance as the distance from the initial state $x_{0,k}$ to the desired state $x_{\mathrm{r},k}$,

$$D_0(\bar{x}_{0,k}) \coloneqq \|x_{0,k} - x_{\mathrm{r},k}\| = \|\bar{x}_{0,k}\|,\ \forall \bar{x}_0 \in \mathbb{R}^n, \tag{4}$$

and the relative distance to $x_{\mathrm{r},k}$,

$$D(\bar{x}) \coloneqq \|x - x_{\mathrm{r},k}\| = \|\bar{x}\|,\ \forall \bar{x} \in \mathbb{R}^n. \tag{5}$$

Since we address the free-final state finite-horizon optimal control problem, the controller at final time T approximates the desired state x_r, i.e., $x(T)$ converges to a close neighborhood around x_r (Bryson, 1975; Lewis, Vrabie, & Syrmos, 2012). In order to reduce the navigation time, we assume that the desired state is reached when the system enters the close neighborhood of the desired final state x_r. That is to say, when $D(\bar{x}) \leq \rho D_0(\bar{x}_{0,k})$, where ρ is the user-defined admissible window, the robot is

considered to have reached the desired state $x_{\mathrm{r},k}$. Subsequently, the system proceeds to the next $(k+1)$-th problem.

Moreover, since the system dynamics are unknown, when RRT$^{\mathrm{X}}$ calculates the collision-free path π, it can only adopt straight lines as edges in E. However, the actual trajectory of the robot is curved due to the kinodynamic constraints (1) and the optimal performance (2). Thus, the actual trajectory deviates from the nominal trajectory provided by the RRT$^{\mathrm{X}}$, and collisions may occur in near-to-obstacle areas. To address this issue, we introduce an obstacle augmentation strategy. More specifically, the algorithm computes at every time instance the kinodynamic distance,

$$D_{\mathrm{rob}}(\bar{x}) \coloneqq \frac{|\bar{x}_{0,k} \times \bar{x}|}{D_{0,k}}, \tag{6}$$

which represents present the current deviation of the robot's current position from the nominal trajectory, i.e. corresponding straight path determined by $(x_{0,k},\, x_{\mathrm{r},k})$. Then, an augmented obstacle space $\mathcal{X}_{\mathrm{obs}}^{\mathrm{aug}}$ is obtained from,

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

where $\mathcal{X}_{\mathrm{kin}}$ is the space of a compact set bounded by a circle with radius the maximum kinodynamic distance $D_{\mathrm{rob}}^{\mathrm{kin}}$. When the maximum kinodynamic distance is updated, RRT$^{\mathrm{X}}$ provides a new path based on the the newest augmented obstacle space (7).

3. Finite-Horizon Boundary Value Problem

Let us define the Hamiltonian with respect to (1) and (3) as,

$$\mathcal{H}(\bar{x}; u; \lambda) \coloneqq \frac{1}{2}(\bar{x}^{\intercal} M \bar{x} + u^{\intercal} R u) + \lambda^{\intercal}(A\bar{x} + Bu),\ \forall \bar{x}, u, \lambda.$$

In order to solve the finite-horizon optimal control problem (3), we use the sweep method (Bryson, 1975) and we set $\lambda = \frac{\partial V^{\star}}{\partial \bar{x}}$. Thus, the Hamilton-Jacobi-Bellman (HJB) equation yields,

$$-\frac{\partial V^{\star}}{\partial t} = \frac{1}{2}(\bar{x}^{\intercal} M \bar{x} + u^{\intercal} R u) + \frac{\partial V^{\star\,\intercal}}{\partial \bar{x}}(A\bar{x} + Bu),\ \forall \bar{x}.$$

Since our system (1) is linear, we write the value function in a quadratic form as,

$$V^{\star}(\bar{x}; t) = \frac{1}{2}\bar{x}^{\intercal} P(t) \bar{x},\ \forall \bar{x}, t \geq t_0, \tag{8}$$

where $P(t) \in \mathbb{R}^{n \times n} \succ 0$ is the Riccati matrix that solves the differential Riccati equation,

$$-\dot{P}(t) = M + P(t)A + A^{\intercal} P(t) - P(t) B R^{-1} B^{\intercal} P(t). \tag{9}$$

Hence, the optimal control gets the form of,

$$u^{\star}(\bar{x}; t) = -R^{-1} B^{\intercal} P(t) \bar{x},\ \forall \bar{x},\, t. \tag{10}$$

Theorem 3.1. *Suppose that there exists a $P(t) \succ 0$ that satisfies the Riccati equation* (9) *with a final condition given by P_T, and the control obtained by,*

$$u(\bar{x};t) = -R^{-1}B^{\intercal}P(t)\bar{x}. \tag{11}$$

Then, the control input (11) *minimizes the cost given in* (3)*, and the origin is a globally uniformly asymptotically stable equilibrium point of the closed-loop system.*

Proof. The proof follows from (Kontoudis & Vamvoudakis, 2019a). □

4. Model-Free Formulation

Let us now define the following Q-function as,

$$\begin{aligned}\mathcal{Q}(\bar{x};u;t) :=& V^{\star}(\bar{x};t) + \mathcal{H}(\bar{x};u;\frac{\partial V^{\star}}{\partial t},\frac{\partial V^{\star}}{\partial \bar{x}}) \\ =& V^{\star}(\bar{x};t) + \frac{1}{2}\bar{x}^{\intercal}M\bar{x} + \frac{1}{2}u^{\intercal}Ru + \bar{x}^{\intercal}P(t)(A\bar{x}+Bu) + \frac{1}{2}\bar{x}^{\intercal}\dot{P}(t)\bar{x},\end{aligned} \tag{12}$$

where $\mathcal{Q}(\bar{x};u;t) \in \mathbb{R}$ is an action-dependent value.

Next, we define the augmented state $U := [\bar{x}^{\intercal}\ u^{\intercal}]^{\intercal} \in \mathbb{R}^{(n+m)}$ to express the Q-function (12) in a compact form as,

$$\mathcal{Q}(\bar{x};u;t) = \frac{1}{2}U^{\intercal}\begin{bmatrix}Q_{\mathrm{xx}}(t) & Q_{\mathrm{xu}}(t)\\ Q_{\mathrm{ux}}(t) & Q_{\mathrm{uu}}\end{bmatrix}U =: \frac{1}{2}U^{\intercal}\bar{\mathcal{Q}}(t)U, \tag{13}$$

where $Q_{\mathrm{xx}}(t) = \dot{P}(t) + P(t) + M + P(t)A + A^{\intercal}P(t)$, $Q_{\mathrm{xu}}(t) = Q_{\mathrm{ux}}^{\intercal}(t) = P(t)B$, and $Q_{\mathrm{uu}} = R$, with $\mathcal{Q} : \mathbb{R}^{n+m} \times \mathbb{R}^{(n+m)\times(n+m)} \to \mathbb{R}$. Using the stationarity condition $\partial\mathcal{Q}(\bar{x};u;t)/\partial u = 0$, we find a model-free expression of the optimal control $u^{\star}$ (10) as,

$$u^{\star}(\bar{x};t) = \arg\min_{u} \mathcal{Q}(\bar{x};u;t) = -Q_{\mathrm{uu}}^{-1}Q_{\mathrm{ux}}(t)\bar{x}. \tag{14}$$

Lemma 4.1. *The value of the minimization $\mathcal{Q}^{\star}(\bar{x};u^{\star};t) := \min_u \mathcal{Q}(\bar{x};u;t)$ is the same with the optimal value $V^{\star}$ in* (8) *of the minimization problem* (3)*, where $P(t) \succ 0$ is the Riccati matrix found from* (9)*.*

Proof. The proof follows from (Kontoudis & Vamvoudakis, 2019a). □

4.1. Actor/Critic Network

A critic approximator is designed to approximate the Q-function in (13) as,

$$\mathcal{Q}^{\star}(\bar{x};u^{\star};t) = \frac{1}{2}U^{\intercal}\begin{bmatrix}Q_{\mathrm{xx}}(t) & Q_{\mathrm{xu}}(t)\\ Q_{\mathrm{ux}}(t) & Q_{\mathrm{uu}}\end{bmatrix}U := \frac{1}{2}\mathrm{vech}(\bar{\mathcal{Q}}(t))^{\intercal}(U\otimes U),$$

where $\mathrm{vech}(\bar{\mathcal{Q}}(t)) \in \mathbb{R}^{\frac{(n+m)(n+m+1)}{2}}$. The half-vectorization operation exploits the symmetric properties of the $\bar{\mathcal{Q}}$ matrix to reduce the computations. Then, by setting

$\nu(t)^\intercal W_{\mathrm{c}} \coloneqq 1/2\mathrm{vech}(\bar{\mathcal{Q}}(t))$ we obtain,

$$\mathcal{Q}^\star(\bar{x}; u^\star; t) = W_{\mathrm{c}}^\intercal \nu(t)(U \otimes U),$$

where $W_{\mathrm{c}} \in \mathbb{R}^{\frac{(n+m)(n+m+1)}{2}}$ is the critic weight estimator vector, and $\nu(t) \in \mathbb{R}^{\frac{(n+m)(n+m+1)}{2} \times \frac{(n+m)(n+m+1)}{2}}$ is a radial basis function of appropriate dimensions that depends explicitly on time. Since the ideal weight estimates are unknown, we employ an adaptive estimation technique (Ioannou & Sun, 2012) to approximate the Q-function,

$$\hat{\mathcal{Q}}(\bar{x}; u; t) = \hat{W}_{\mathrm{c}}^\intercal \nu(t)(U \otimes U), \tag{15}$$

where $\hat{W}_{\mathrm{c}}\nu(t) \coloneqq \frac{1}{2}\mathrm{vech}(\hat{\bar{\mathcal{Q}}}(t))$.

By using a similar way of thinking for the actor we assign $\mu(t)^\intercal W_{\mathrm{a}} \coloneqq -Q_{\mathrm{uu}}^{-1} Q_{\mathrm{ux}}(t)$ to write,

$$u^\star(\bar{x}; t) = W_{\mathrm{a}}^\intercal \mu(t)\bar{x},$$

where $W_{\mathrm{a}} \in \mathbb{R}^{n \times m}$ is the actor weight estimator vector, $\mu(t) \in \mathbb{R}^{n \times n}$ is a radial basis function of appropriate dimensions that depends explicitly on time. The actor by using current weight estimates yields,

$$\hat{u}(\bar{x}; t) = \hat{W}_{\mathrm{a}}^\intercal \mu(t)\bar{x}. \tag{16}$$

Remark 1. The approximation errors of the critic and the actor approximators described in (15) and (16) respectively, vanish as the system (1) is linear. To this end, we exploit the whole space and not just a compact set. With this structure, the approximations will converge to the optimal policies, and hence the superscript $\star$, that denotes the ideal values of the adaptive weight estimation, render similarly with the optimal solutions.

Next, we adopt an integral reinforcement learning approach (Vrabie, Vamvoudakis, & Lewis, 2013) that let us express the Bellman equation as,

$$V^\star(\bar{x}(t); t) = V^\star(\bar{x}(t - \Delta t); t - \Delta t) - \frac{1}{2}\int_{t-\Delta t}^{t} (\bar{x}^\intercal M \bar{x} + u^{\star\intercal} R u^\star)\, \mathrm{d}\tau, \tag{17}$$

$$V^\star(\bar{x}(T); T) = \frac{1}{2}\bar{x}^\intercal(T) P(T) \bar{x}(T), \tag{18}$$

where $\Delta t \in \mathbb{R}^+$ is a small fixed value, i.e. resolution. By following Lemma 4.1, where we proved that $\mathcal{Q}^\star(\bar{x}; u^\star; t) = V^\star(\bar{x}; t)$, we can write (17) and (18) as,

$$\mathcal{Q}^\star(\bar{x}(t); u^\star(t); t) = \mathcal{Q}^\star(\bar{x}(t - \Delta t); u^\star(t - \Delta t); t - \Delta t) - \frac{1}{2}\int_{t-\Delta t}^{t} (\bar{x}^\intercal M \bar{x} + u^{\star\intercal} R u^\star)\, \mathrm{d}\tau,$$

$$\mathcal{Q}^\star(\bar{x}(T); T) = \frac{1}{2}\bar{x}^\intercal(T) P(T) \bar{x}(T).$$

Next, we define the errors $e_{\mathrm{c}_1}, e_{\mathrm{c}_2} \in \mathbb{R}$, that we seek to drive to zero by appropriately

tuning the critic weights of (15). Define the first critic error e_{c_1} as,

$$\begin{aligned} e_{c_1} \coloneqq & \hat{\mathcal{Q}}(\bar{x}(t); \hat{u}(t); t) - \hat{\mathcal{Q}}(\bar{x}(t-\Delta t); \hat{u}(t-\Delta t); t-\Delta t) \\ & + \frac{1}{2}\int_{t-\Delta t}^{t} (\bar{x}^\intercal M \bar{x} + \hat{u}^\intercal R \hat{u})\, \mathrm{d}\tau \\ = & \hat{W}_c^\intercal \Big(\nu(t)(U(t) \otimes U(t)) - \nu(t-T)(U(t-\Delta t) \otimes U(t-\Delta t)) \Big) \\ & + \frac{1}{2}\int_{t-\Delta t}^{t} (\bar{x}^\intercal M \bar{x} + \hat{u}^\intercal R \hat{u})\, \mathrm{d}\tau, \end{aligned} \tag{19}$$

Intrinsic dynamics are included in (19), which can be evaluated by taking the time derivative,

$$\dot{p} = \bar{x}^\intercal(t) M \bar{x}(t) - \bar{x}^\intercal(t-\Delta t) M \bar{x}(t-\Delta t) + \hat{u}^\intercal(t) R \hat{u}(t) - \hat{u}^\intercal(t-\Delta t) R \hat{u}(t-\Delta t).$$

The second critic error is defined by,

$$e_{c_2} \coloneqq \frac{1}{2}\bar{x}^\intercal(t) P(T) \bar{x}(t) - \hat{W}_c^\intercal \nu(t)(U(t) \otimes U(t)).$$

The actor approximator error $e_a \in \mathbb{R}^m$ is defined by,

$$e_a \coloneqq \hat{W}_a^\intercal \mu(t) \bar{x} + \hat{Q}_{uu}^{-1} \hat{Q}_{ux}(t) \bar{x},$$

where $\hat{Q}_{uu}$, $\hat{Q}_{ux}$ will be obtained from the critic weight matrix estimation $\hat{W}_c$. By employing adaptive control techniques (Ioannou & Sun, 2012), we formulate the squared-norm of errors as,

$$K_1(\hat{W}_c, \hat{W}_c(T)) = \frac{1}{2}\|e_{c_1}\|^2 + \frac{1}{2}\|e_{c_2}\|^2, \tag{20}$$

$$K_2(\hat{W}_a) = \frac{1}{2}\|e_a\|^2. \tag{21}$$

4.2. Learning Methodology

The weights of the critic estimation matrix are obtained by applying a normalized gradient descent algorithm in (20),

$$\dot{\hat{W}}_c = -\alpha_c \frac{\partial K_1}{\partial \hat{W}_c} = -\alpha_c \left(\frac{1}{(1+\sigma^\intercal\sigma)^2}\sigma e_{c_1} + \frac{1}{(1+\sigma_f^\intercal\sigma_f)^2}\sigma_f e_{c_2} \right), \tag{22}$$

where $\sigma(t) \coloneqq \nu(t)(U(t) \otimes U(t) - U(t-\Delta t) \otimes U(t-\Delta t))$, $\sigma_f(t) = \nu(t)(U(t) \otimes U(t))$, and $\alpha_c \in \mathbb{R}^+$ is a constant gain that specifies the convergence rate. The critic tuning (22) guarantees that as $e_{c_1} \to 0$ and $e_{c_2} \to 0$ then $\hat{W}_c \to W_c$ and $\hat{W}_c(T) \to W_c(T)$.

Similarly, the weights of the actor estimation matrix $\hat{W}_a$ by applying a gradient descent algorithm in (21) yield,

$$\dot{\hat{W}}_a = -\alpha_a \frac{\partial K_2}{\partial \hat{W}_a} = -\alpha_a \bar{x} e_a^\intercal, \tag{23}$$

where $\alpha_{\mathrm{a}} \in \mathbb{R}^{+}$ is a constant gain that specifies the convergence rate. The actor estimation algorithm (23) guarantees that as $e_{\mathrm{a}} \to 0$ then $\hat{W}_{\mathrm{a}} \to W_{\mathrm{a}}$.

For the theoretical analysis we introduce the weight estimation error for the critic $\tilde{W}_{\mathrm{c}} \coloneqq W_{\mathrm{c}} - \hat{W}_{\mathrm{c}}$ and for the actor $\tilde{W}_{\mathrm{a}} \coloneqq W_{\mathrm{a}} - \hat{W}_{\mathrm{a}}$, with $\tilde{W}_{\mathrm{c}} \in \mathbb{R}^{\frac{(n+m)(n+m+1)}{2}}$, $\tilde{W}_{\mathrm{a}} \in \mathbb{R}^{n \times m}$. The estimation error dynamics of the critic yields,

$$\dot{\tilde{W}}_{\mathrm{c}} = -\alpha_{\mathrm{c}} \frac{1}{(1+\sigma^{\intercal}\sigma)^{2}} \sigma \sigma^{\intercal} \tilde{W}_{\mathrm{c}},$$

and the estimation error dynamics of the actor becomes,

$$\dot{\tilde{W}}_{\mathrm{a}} = -\alpha_{\mathrm{a}} \bar{x}\bar{x}^{\intercal} \mu(t)^{\intercal} \tilde{W}_{\mathrm{a}} - \alpha_{\mathrm{a}} \bar{x}\bar{x}^{\intercal} \frac{\mu(t) \tilde{Q}_{\mathrm{xu}} R^{-1}}{\|1 + \mu(t)^{\intercal}\mu(t)\|^{2}}, \tag{24}$$

where $\tilde{Q}_{\mathrm{xu}} \coloneqq \mathrm{mat}(\tilde{W}_{\mathrm{c}}[\frac{n(n+1)}{2} + 1 : \frac{n(n+1)}{2} + nm])$.

Lemma 4.2. *For any given control input $u(t) \in \mathcal{U}$ the estimation error dynamics of the critic* (24) *have an exponentially stable equilibrium point at the origin as follows,*

$$\|\tilde{W}_{c}\| \leq \|\tilde{W}_{c}(t_0)\| \kappa_1 e^{-\kappa_2 (t - t_0)},$$

where $\kappa_1, \kappa_2 \in \mathbb{R}^{+}$. In order to establish exponential stability, we require the signal $\Delta(t) \coloneqq \frac{\sigma(t)}{1+\sigma(t)^{\intercal}\sigma(t))}$ to be persistently exciting (PE) at $[t, t+T_{PE}]$, where $T_{PE} \in \mathbb{R}^{+}$ the excitation period, if there exists a $\beta \in \mathbb{R}^{+}$ such that $\beta I \leq \int_{t}^{t+T_{PE}} \Delta(\tau)\Delta^{\intercal}(\tau) d\tau$, where I is an identity matrix of appropriate dimensions.

Proof. The proof follows from (Vamvoudakis, 2017). □

The main stability theorem for the Q-learning framework is provided below.

Theorem 4.3. *Consider the linear time-invariant continuous-time system* (1)*, the critic, and the actor approximators given by* (15)*, and* (16) *respectively. The weights of the critic, and the actor estimators are tuned by* (22)*, and* (23) *respectively. The origin with state $\psi = [\bar{x}^{\intercal} \ \tilde{W}_{c}^{\intercal} \ \tilde{W}_{a}^{\intercal}]^{\intercal}$ is a globally uniformly asymptotically stable equilibrium point of the closed-loop system and for all initial conditions $\psi(0)$, given that the critic gain α_c is sufficiently larger than the actor gain α_a and the following inequality holds,*

$$0 < \alpha_a < \frac{2\underline{\lambda}(M + Q_{xu} R^{-1} Q_{xu}^{\intercal}) - \overline{\lambda}(Q_{xu} Q_{xu}^{\intercal})}{\delta \overline{\lambda}\left(\frac{\mu(t) R^{-1}}{\|1+\mu(t)^{\intercal}\mu(t)\|^{2}}\right)}, \tag{25}$$

with δ a constant of unity order.

Proof. The proof follows from (Kontoudis & Vamvoudakis, 2019a). □

5. Kinodynamic Motion Planning Framework

The motion planning structure comprises of four stages: i) dynamic planning; ii) Q-learning; iii) terminal state evaluation; and iv) obstacle augmentation, as shown in

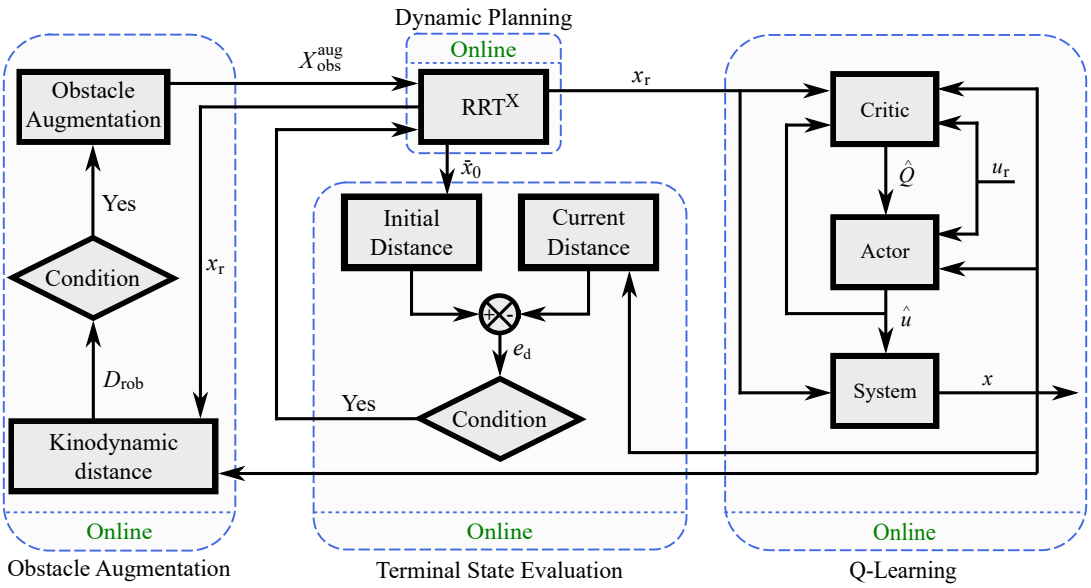

Figure 1. The motion planning structure consists of four stages. The algorithm runs clockwise, starting from the dynamic planning.

Fig. 1. The path planning is assigned to RRT$^{\mathrm{X}}$ which provides all the waypoints for navigation in a dynamic environment. Next, the online Q-learning and the terminal state evaluation are implemented. In parallel, we monitor the kinodynamic distance to augment the obstacle space. The implementation is presented in Algorithm 1

Dynamic Planning: The RRT$^{\mathrm{X}}$ contains not only the sub-tree—which stores the desired path—but also the search-graph of the initial planning process. In this way, the algorithm reuses the search-graph for a rewiring cascade whenever the domain changes. Consequently, information is transferred rapidly throughout the tree in the modified environment. Moreover, RRT$^{\mathrm{X}}$ maintains an ϵ-consistent[1] graph, which guarantees the quality of existing paths and allows for quick replanning. The neighborhood size at each node remains constant by selecting neighbors to maintain the runtime at each iteration. Therefore, RRT$^{\mathrm{X}}$ can provide a quick reaction in unpredictable dynamic environments as well as high-quality new paths.

Obstacle Augmentation: Since the model of the system is unknown, it is assumed that the robot traverses straight paths, given the waypoints from the RRT$^{\mathrm{X}}$. In addition, optimality in terms of path planning usually indicates narrow distance between the obstacles and the path. In our case, kinodynamic constraints (1) as well as the optimal performance (2) result in traversing curved paths, instead of the assumed straight-line paths. Thus, there exists a deviation from the assumed straight-line path and the traversed path of the robot. This deviation of paths may result in unsafe navigation with collisions. To address this problem, we introduce the concept of kinodynamic distance and follow an obstacle augmentation strategy (Kontoudis & Vamvoudakis, 2019a,b). Therefore, instead of considering the physical shape of the obstacles, their augmented shape is taken into account. The augmented obstacle space $\mathcal{X}_{\mathrm{obs}}^{\mathrm{aug}}$ is computed trough the Minkowski sum (7) based on the maximum kinodynamic distance $D_{\mathrm{rob}}^{\mathrm{kin}}$. Whenever new obstacles are detected, the obstacle augmentation precedes the replanning process to avoid collision.

[1] ϵ-consistency means that the cost-to-goal value is within ϵ of the minimum of sum distance-to-neighbor and the neighbor's cost-to-go.

Algorithm 1 RRT-Q^{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; $\mathcal{X}_{\mathrm{obs}}$ - obstacle space; $\mathcal{X}$ - state space

Output: $\hat{u}$ - control

$\alpha_{\mathrm{a}}, \alpha_{\mathrm{c}} \leftarrow \texttt{Stability}(M, R)$ (25);
$\mathcal{X}_{\mathrm{obs}}^{\mathrm{aug}} \leftarrow \mathcal{X}_{\mathrm{obs}}$;
$D_{\mathrm{rob}}, D_{\mathrm{rob}}^{\mathrm{kin}} \leftarrow 0$; $k \leftarrow 1$;
while $x_{\mathrm{goal}} \neq x$ **do**
  while *NoCollision* **do**
    $D_0 \leftarrow \texttt{InitialDistance}(x_0)$ (4);
    for $t \in T$ **do**
      if $D_{\mathrm{rob}} > D_{\mathrm{rob}}^{\mathrm{kin}}$ **then** ▷ Obstacle augmentation
        $D_{\mathrm{rob}}^{\mathrm{kin}} \leftarrow D_{\mathrm{rob}}$;
        $\mathcal{X}_{\mathrm{obs}}^{\mathrm{aug}} \leftarrow \texttt{Augment}(\mathcal{X}_{\mathrm{obs}}, D_{\mathrm{rob}}^{\mathrm{kin}})$ (7);
      end if
      $\hat{W}_{\mathrm{c}} \leftarrow \texttt{Critic}(M, R, \Delta t, \alpha_{\mathrm{c}}, \bar{x}, \hat{u})$ (22); ▷ Q-learning
      $\hat{\mathcal{Q}} \leftarrow \texttt{EstimateQ}(\hat{W}_{\mathrm{c}}, \bar{x}, \hat{u})$ (15);
      $\hat{W}_{\mathrm{a}} \leftarrow \texttt{Actor}(\hat{\mathcal{Q}}, \alpha_{\mathrm{a}}, \bar{x})$ (23);
      $\hat{u} \leftarrow \texttt{Control}(\hat{W}_{\mathrm{a}}, \bar{x})$ (16);
      return $\hat{u}$;
      $D_{\mathrm{rob}} \leftarrow \texttt{KinodynamicDistance}(x_{0,k}, \bar{x}, D_0)$ (6);
      if $D \leq \rho D_0$ **then** ▷ Terminal state evaluation
        $x_{0,k} \leftarrow x(t)$;
        $k \leftarrow k + 1$;
        break;
      end if
    end for
  end while
  $\mathcal{G}, \pi \leftarrow \texttt{RRT}^{\mathrm{X}}(\mathcal{X}, \mathcal{X}_{\mathrm{obs}}^{\mathrm{aug}}, x_{\mathrm{start}}, x_{\mathrm{goal}})$; ▷ Dynamic planning
end while

Q-Learning: At every k-pair of waypoints $(x_{0,k}, x_{\mathrm{r},k})$ of the planned path π, the proposed control law (16) is implemented to drive the system. The critic is used to assess the policy, and the actor to perform the policy update. The critic approximates the Q-function according to (15), where $\hat{W}_{\mathrm{c}}$ are the critic parameters that can be computed online by (22). The actor approximates the optimal control policy according to (16), where $\hat{W}_{\mathrm{a}}$ are the actor parameters following the tuning law (23).

Terminal State Evaluation: A distance metric is employed to evaluate the terminal condition. At every Δt we compute the initial distance D_0 (4) and the relative distance D (5). When the relative distance drops below an admissible portion of the initial distance $D \leq \rho D_0$, then the algorithm assigns the current state as the new initial state $x_{0,k+1} = x(t)$ and proceeds to the next $(k+1)$-pair of waypoints.

Remark 2. The obstacle augmentation is a conservative strategy, because we are using the maximum kinodynamic distance to limit the free space. This means we are reserving space based on the worst case scenario that may appear only few times during the navigation. However, since we tackle the model-free problem without offline trials, that is a feasible methodology to ensure collision-free navigation.

6. Simulations and Results

In this section, we demonstrate the efficiency of the proposed motion planning technique in a dynamic environment with unpredictably appearing obstacles. We consider the continuous-time linear time-invariant system,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \ddot{x} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -.5 & 0 & -1.125 & 0 \\ 0 & -.5 & 0 & -1.125 \end{bmatrix} \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ .025 & 0 \\ 0 & .025 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}, \tag{26}$$

where x, y is the translation, $\dot{x}$, $\dot{y}$ the velocity, and $\ddot{x}$, $\ddot{y}$ the acceleration along the x and y axes respectively. The inputs forces are denoted f_1, f_2. The system in (26) represents an autonomous rover. More details about the parameters of (26) are discussed in (Kontoudis & Vamvoudakis, 2019a).

The autonomous rover has full state feedback and limited perception range. Thus, an accurate configuration of the obstacles can be detected only in the perception range. The environment is completely unknown and consists of obstacles that appear throughout the navigation. Since the environment is unknown, we suppose there are no obstacles other than the obstacles detected in the perception range. The proposed method measures the kinodynamic distance D_{rob} and updates the augmented obstacle space $\mathcal{X}_{\text{obs}}^{\text{aug}}$ at every time instance Δt. We set the finite horizon as $T = 10\,\text{s}$ for every run and the admissible window $\rho = 0.15$. The user-defined matrices are $M = 10I_4$, and $R = 2I_2$. The final Riccati matrix is $P(T) = 0.5I_4$. We set the actor and critic gains as $\alpha_{\text{c}} = 90$, and $\alpha_{\text{a}} = 1.2$ respectively, by following (25). The resolution is $\Delta t = 0.05\,\text{s}$. The initial values of the critic estimator vector $\hat{W}_{\text{c}}$ and the actor estimator vector $\hat{W}_{\text{a}}$ are randomly selected, except of the last three elements of $\hat{W}_{\text{c}}$ that need to be non-zero. These elements are the $\{\hat{W}_{\text{c}}\}_{19:21} = Q_{\text{uu}}$ values that is inverted in (14). Note that there are three elements, because the user defined matrix R is symmetric and we are also employing the half-vectorization operation in (15). For the implementation of the RRT$^{\text{X}}$ we use the package in (Otte, 2016).

The simulation is shown in Fig. 2, and a demonstrating video is available online[2]. The environment is a square with corners $(-20, -20)$ and $(20, 20)$ in meters. The start state is $x_{\text{start}} = (-15, -15, 0, 0)$ and the goal state $x_{\text{goal}} = (17, 17, 0, 0)$. The perception range is omnidirectional with radius of 8 m and is illustrated with a red dashed circle. The traversed path of the rover is shown in a red solid line and the RRT$^{\text{X}}$ path in a white line. The gray solid lines represent the search-tree of RRT$^{\text{X}}$. In every BVP, the rover moves toward the red dot, which denotes the desired state. The initial shape of the obstacles is denoted by blue polygons and the corresponding augmentation in green. The colored background represent the cost-to-go from every location to the final goal. Note that the cost-to-go is an underestimated value, as it is calculated with respect to straight-line paths. The maximum kinodynamic distance $D_{\text{rob}}^{\text{kin}}$, the current kinodynamic distance D_{rob}, and the cost-to-go of RRT$^{\text{X}}$ are shown in the left, right and bottom colorbars, respectively. The autonomous rover successfully avoids the obstacles throughout the navigation in an unpredictable dynamic environment using the proposed kinodynamic motion planning technique.

[2] `https://youtu.be/vNvOMTzxd0c`.

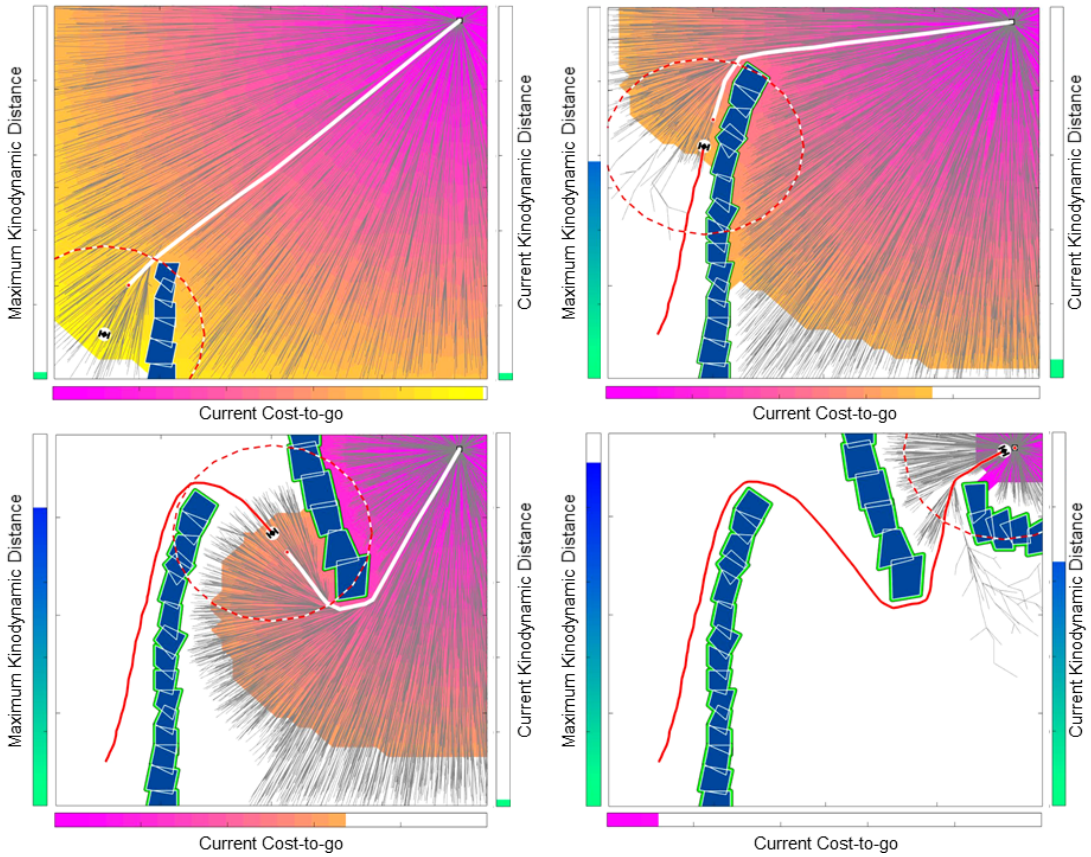

Figure 2. Various time frames of the autonomous rover collision-free navigation in an unpredictable dynamic environment.

7. Conclusion

This paper proposed a real-time kinodynamic motion planning methodology for unpredictable dynamic environments. More precisely, we introduced a Q-learning control law to approximate the optimal policy of a continuous-time linear time-invariant system and we used a terminal state evaluation and an obstacle augmentation technique. We rigorously derived the Q-learning controller, so that global asymptotic stability of the equilibrium point is ensured. The simulations reveal that the autonomous rover can efficiently perform safe navigation with no collisions in an unknown dynamic domain. Our methodology is completely model-free without offline training and requires insignificant computations that facilitate the execution of the algorithm in real-time.

Funding

This work was supported in part, by NASA ULI, and by NSF under grant Nos. CAREER CPS-1851588 and S&AS 1849198.

References

Allen, R. E. & Pavone, M. (2019). A real-time framework for kinodynamic planning in dynamic environments with application to quadrotor obstacle avoidance. *Robotics and Autonomous Systems*, *115*, 174–193.

Berkenkamp, F. & Schoellig, A. P. (2015, July). Safe and robust learning control with Gaussian processes. In *IEEE European Control Conference (ECC)* (pp. 2496–2501).

Bruce, J. & Veloso, M. M. (2002, June). Real-time randomized path planning for robot navigation. In *Robot Soccer World Cup* (pp. 288–295). Springer, Berlin, Heidelberg.

Bryson, A. E. (1975). *Applied optimal control: Optimization, estimation and control.* CRC Press.

Busoniu, L., Babuska, R., De Schutter, B., & Ernst, D. (2010). *Reinforcement learning and dynamic programming using function approximators* (Vol. 39). CRC press.

Chiang, H. T. L., Hsu, J., Fiser, M., Tapia, L., & Faust, A. (2019). RL-RRT: Kinodynamic motion planning via learning reachability estimators from RL policies. *IEEE Robotics and Automation Letters*, *4*(4), 4298–4305.

Donald, B., Xavier, P., Canny, J., & Reif, J. (1993). Kinodynamic motion planning. *Journal of the ACM*, *40*(5), 1048–1066.

Ferguson, D., Kalra, N., & Stentz, A. (2006, May). Replanning with RRTs. In *IEEE International Conference on Robotics and Automation*, (pp. 1243–1248).

Hsu, D., Kindel, R., Latombe, J. C., & Rock, S. (2002). Randomized kinodynamic motion planning with moving obstacles. *International Journal of Robotics Research*, *21*(3), 233–255.

Ioannou, P. A. & Sun, J. (2012). *Robust adaptive control*. Courier Corporation.

Kavraki, L. E., Svestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*(4), 566–580.

Karaman, S. & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, *30*(7), 846–894.

Kontoudis, G. P. & Vamvoudakis, K. G. (2019). Kinodynamic motion planning with continuous-time Q-learning: An online, model-free, and safe navigation framework. *IEEE Transactions on Neural Networks and Learning Systems*, *30*(12), 3803–3817.

Kontoudis, G. P. & Vamvoudakis, K. G. (2019, July). Robust kinodynamic motion planning using model-free game-theoretic learning. In *American Control Conference* (pp. 273–278).

Kontoudis, G. P., Xu, Z., & Vamvoudakis, K. G. (2020, July). Online, Model-Free Motion Planning in Dynamic Environments: An Intermittent, Finite Horizon Approach with Continuous-Time Q-Learning. In *American Control Conference* (pp. 3873–3878).

Kuffner, J. J. & LaValle, S. M. (2000, April). RRT-connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation* (pp. 995–1001).

LaValle, S. M. (1998). Rapidly-exploring random trees: A new tool for path planning.

LaValle, S. M. & Kuffner Jr, J. J. (2001). Randomized kinodynamic planning. *international Journal of Robotics Research*, *20*(5), 378–400.

Li, Y., Cui, R., Li, Z., & Xu, D. (2018). Neural network approximation based near-optimal motion planning with kinodynamic constraints using RRT. *IEEE Transactions on Industrial Electronics*, *65*(11), 8718–8729.

Lewis, F. L., Vrabie, D., & Syrmos, V. L. (2012). *Optimal control*. John Wiley & Sons.

Lewis, F. L., Vrabie, D., & Vamvoudakis, K. G. (2012). Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers. *IEEE Control Systems Magazine*, *32*(6), 76–105.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., ... & Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint* arXiv:1509.02971.

Mehta, P. & Meyn, S. (2009, December). Q-learning and Pontryagin's minimum principle. In *IEEE Conference on Decision and Control* (pp. 3598–3605).

Nägeli, T., Alonso-Mora, J., Domahidi, A., Rus, D., & Hilliges, O. (2017). Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization. *IEEE Robotics and Automation Letters*, *2*(3), 1696–1703.

Netter, J., Kontoudis, G. P., & Vamvoudakis, K. G. (2021, December). Bounded Rational RRT-QX: Multi-Agent Motion Planning in Dynamic Human-Like Environments Using Cognitive Hierarchy and Q-Learning. In *IEEE Conference on Decision and Control* (pp. 3597–3602).

Otte, M. & Frazzoli, E. (2014). RRT$^{\mathrm{X}}$: Real-Time Motion Planning/Replanning for Environments with Unpredictable Obstacles. *Algorithmic Foundations of Robotics*, (pp. 461–478).

Otte, M. & Frazzoli, E. (2016). RRT$^{\mathrm{X}}$: Asymptotically optimal single-query sampling-based motion planning with quick replanning. *International Journal of Robotics Research*, *35*(7), 797–822.

Otte, M. (2016). RRT-X (dynamic obstacles). Retrieved from: `http://ottelab.com/html_stuff/code.html#RRTXcode`

Perez, A., Platt, R., Konidaris, G., Kaelbling, L., & Lozano-Perez, T. (2012, May). LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics. In *IEEE International Conference on Robotics and Automation* (pp. 2537–2542).

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality* (Vol. 703). John Wiley & Sons.

Sutton, R. S. & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Tedrake, R., Manchester, I. R., Tobenkin, M., & Roberts, J. W. (2010). LQR-trees: Feedback motion planning via sums-of-squares verification. *International Journal of Robotics Research*, *29*(8), 1038–1052.

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

Vrabie, D., Vamvoudakis, K. G., & Lewis, F. L. (2013). *Optimal adaptive control and differential games by reinforcement learning principles* (Vol. 2). IET.

Watkins, C. J. & Dayan, P. (1992). Q-learning. *Machine learning*, *8*(3-4), 279–292.

Webb, D. J. & Van Den Berg, J. (2013, May). Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics. In *IEEE International Conference on Robotics and Automation* (pp. 5054–5061).

Yang, G. Z., Bellingham, J., Dupont, P. E., Fischer, P., Floridi, L., Full, R., ..., Nelson, B. J. (2018). The grand challenges of Science Robotics. *Science robotics*, *3*(14).