Sampling-based Motion Planning for Uncertain High-dimensional Systems via Adaptive Control

Christos K. Verginis[1], Dimos V. Dimarogonas[1], and Lydia E. Kavraki[2]

[1] KTH Royal Institute of Technology, Stockholm, 100 44, Sweden,
{cverginis,dimos}@kth.se
[2] Rice University, Houston, TX 77005, USA
kavraki@rice.edu

Abstract. This paper considers the problem of safe motion planning for high-dimensional holonomic robots with 2nd-order uncertain dynamics. We integrate sampling-based motion planning techniques with traditional adaptive feedback control and address difficulties encountered in planning for such systems. More specifically, we develop a feedback control scheme that tracks a given reference trajectory within certain bounds, while simultaneously compensating for potential uncertainty in the dynamical parameters of the robot (masses, moments of inertia) and external disturbances. Employing this result, we are able to cast the problem of kinodynamic motion planning to a geometric one, which can be usually solved more efficiently since it does not take into account the robot dynamics (a.k.a. differential constraints). Intuitively, we use a geometric planner to obtain a high-level safe path for the robot, and a low-level adaptive feedback control algorithm to execute it while taking into account the robot dynamics. Experimental results validate the proposed approach.

Keywords: motion planning, robotic manipulation, adaptive control

1 Introduction

Robot motion planning is one of the fundamental problems in the field of robotics, with numerous applications such as exploration of unknown environments, autonomous driving, robotic manipulation, and autonomous warehouses [1,2]. The topic has been concerning researchers for several decades; early works propose efficient discrete as well as feedback-based algorithms to solve the problem in low-dimensional spaces (e.g., [3,4]).

The problem becomes significantly more complex as the dimension of the configuration space increases. Randomized planning has been introduced to overcome the respective scalability issues; [5,6,7] introduce the notions of probabilistic roadmaps (PRM) and random trees (RRT, EST), respectively, which constitute efficient and probabilistically complete solutions to multiple- and single-query, respectively, high-dimensional motion planning problems. The intuition behind these algorithms is the addition of random sampled states of the free space to a discrete graph/tree, promoting the search of the unexplored free space.

Furthermore, although the initial works derive geometric solutions in the configuration space, trees have been extended to kinodynamic planning, where the robot dynamics (a.k.a. differential constraints) $\dot{x} = f(x, t, u)$ are taken into account [6,8,9,10]. In these algorithms, the robot dynamics are simulated forward in time, possibly by randomly sampling inputs, in order to find a feasible path. Except for the randomized inputs, the incremental step as well as the duration of this forward simulation are often also chosen randomly. In high dimensional spaces, this randomness might require excessive tuning of the aforementioned parameters in order to find a solution in a reasonable amount of time. Along the lines of PRM, [11] and [12] introduce the notion of LQR-trees, which constitute trees of trajectories that probabilistically cover the state space. In that way, every controllable initial condition belongs to the region of attraction (funnel) of a trajectory and is thus driven to the goal via local linearization of the dynamics and optimal feedback control. Dynamics linearization and reachability sets were also recently used to develop an optimal kinodynamic algorithm, namely R3T [13].

A potential drawback of the aforementioned algorithms on kinodynamic motion planning is their strong dependence on the robot dynamics, which in general might be uncertain/unknown. Therefore, the robot model used in standard forward simulation-based kinodynamic algorithms might deviate from the actual dynamics, outputting hence paths that might be colliding with obstacles or difficult to be realized by the actual robotic system. Similar to the LQR-trees, [14] proposes an algorithm that builds trees of funnels based on the (known) bounds of model disturbances, restricted however to polynomial robot dynamics. Planning under uncertainty has been also considered in a stochastic framework and via belief trees [15,16,17,18]. These approaches, however, usually deal with linearized dynamics, and/or propagate the uncertainties on the planning horizon, constraining excessively the free space.

In this paper, we propose a two-layer framework that integrates "intelligent" feedback control protocols with geometric motion planning for high-dimensional Lagrangian holonomic systems (e.g., robotic manipulators). Firstly, motivated by the difficulty of measuring accurately the robotic system's dynamical parameters (like masses, and moments of inertia) as well as potential external disturbances, we design a feedback control scheme that does not use any information on these parameters/disturbances (i.e., these are *unknown* to the controller). The control scheme is a variation of standard adaptive control design, and aims at achieving tracking of a given trajectory for the robot, which is assumed to obey 2nd-order dynamics. The tracking of the trajectory is achieved within certain bounds that stem from the aforementioned uncertainties/disturbances. These bounds create an implicit funnel around the trajectory, which can be further shrunk by appropriate tuning of the control parameters, the latter being a standard procedure in adaptive control design. This funnel is then incorporated in a RRT-like algorithm, which outputs a path connecting an initial configuration to the goal. The construction of the RRT and the employed control protocol guarantee that the robot will follow the derived path without colliding with the workspace obstacles.

In that way, by using appropriate feedback control, the proposed methodology "relieves" the sampling-based motion planner of the robot dynamics and their uncertainties, hence the problem of constructing a path becomes purely geometrical. The motion planner relies only on the performance of the control layer, encoded in the aforementioned bounds. Similar ideas were pursued in [19] and [20]; [19], however, just provides a general idea of interfacing the planning and control layers, without elaborating on a particular systematic control technique, while [20] considers mainly predictive controllers for linear systems, without avoiding the forward simulation of the available system model. The proposed framework in this work exhibits the following important characteristics: 1) The robot dynamics are not forward simulated and hence they are decoupled from the motion planner. Consequently, even though a 2nd-order system is considered, the motion planner is purely geometrical and depends on the geometry of the configuration space as well as the bounds of the robot uncertainties. 2) We do not resort to linearization of the dynamics and computation of basins of attraction around the output trajectories, since the designed feedback control protocol applies directly to the nonlinear model. Finally, the proposed algorithm is expected, in practice, to exhibit lower complexity than standard kinodynamic planning algorithms, since it is purely geometrical and does not simulate any differential equations. The proposed methodology is validated using a UR5 robotic manipulator in V-REP environment ([21]).

2 Problem Formulation

Consider a robotic system with state $(q, \dot{q}) \in \mathbb{T} \times \mathbb{R}^n \subset \mathbb{R}^{2n}$, $n \in \mathbb{N}$, representing its positions and velocities. Usual robotic structures (e.g., robotic manipulators) might consist of translational and rotational joints, which we define here as $q_{tr} \in \mathbb{R}^{n_{tr}}$ and $q_r \in [0, 2\pi)^{n_r}$, respectively, with $n_{tr} + n_r = n$, and hence $\mathbb{T} \coloneqq \mathcal{W}_{tr} \times [0, 2\pi)^{n_r}$, where $\mathcal{W}_{tr}$ is a closed subset of $\mathbb{R}^{n_{tr}}$. Without loss of generality, we assume that $q = [q_{tr}^\top, q_r^\top]^\top$. We consider that the equations of motion of the robot obey the standard 2nd-order Lagrangian dynamics

$$B(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) + d(t) = u, \tag{1}$$

where $B : \mathbb{T} \to \mathbb{R}^{n \times n}$ is the *positive definite* inertia matrix, $C : \mathbb{T} \times \mathbb{R}^6 \to \mathbb{R}^{n \times n}$ is the Coriolis term, $g : \mathbb{T} \to \mathbb{R}^n$ is the gravity vector, and $d : \mathbb{R}_{\geq 0} \to \mathbb{R}^n$ is a term modeling external disturbances and possibly model structural uncertainties and friction terms; $d(t)$ is assumed to be continuous and uniformly bounded by a *known* bound $\bar{d}$ as $\|d(t)\| \leq \bar{d}$, $\forall t \geq 0$ (which is a common assumption in such systems, see, e.g. [22]). Finally, $u \in \mathbb{R}^n$ is the vector of torques representing the control input. An important property of the terms of (1), which will be used later, is the following ([23]):

Property 1. The term $\dot{B}(q) - 2C(q, \dot{q})$ is skew-symmetric, i.e., $y^\top(\dot{B}(q) - 2C(q, \dot{q}))y = 0$, $\forall y \in \mathbb{R}^n$.

Note that the dynamical terms $B(q)$, $C(q,\dot{q})$, $g(q)$ of (1) depend on the dynamical parameters of the robot, i.e., its mass and moment of inertia. These parameters are assumed to be *unknown* to the user/designer, and hence they cannot be used in the planning and control modules. The same applies to the function $d(t)$. Nevertheless, as will be shown later, having satisfying estimates for these terms renders the planning module for the robot less conservative in terms of collision checking.

We consider that the robot operates in a workspace $\mathcal{W} \subset \mathbb{R}^3$ filled with obstacles occupying a closed set $\mathcal{O} \subset \mathbb{R}^3$. We denote the set of points that consist the volume of the robot at configuration q as $\mathcal{A}(q) \subset \mathbb{R}^3$. The collision-free space is defined as the open set $\mathcal{A}_{\text{free}} \coloneqq \{q \in \mathbb{T} : \mathcal{A}(q) \cap \mathcal{O} = \emptyset\}$. Our goal is to achieve safe navigation of the robot to a predefined goal region $Q_g \subset \mathcal{A}_{\text{free}}$ from an initial configuration $q(0) \in \mathcal{A}_{\text{free}}$ via a path $\boldsymbol{q}_{\text{p}} : [0,\sigma] \to \mathcal{A}_{\text{free}}$ satisfying $\boldsymbol{q}_{\text{p}}(0) = q(0)$ and $\boldsymbol{q}_{\text{p}}(\sigma) \in Q_g$, for some positive σ.

The problem we consider is the following:

Problem 1. Given $q(0)) \in \mathcal{A}_{\text{free}}$ and $Q_g \subset \mathcal{A}_{\text{free}}$, respectively, design a control trajectory $u : [0, t_f] \to \mathbb{R}^n$, for some finite $t_f > 0$, such that the solution $q^*(t)$ of (1) satisfies $q^*(t) \in \mathcal{A}_{\text{free}}$, $\forall t \in [0, t_f]$, and $q^*(t_f) \in Q_g$.

The feasibility of Problem 1 is established in the following assumption.

Assumption 1 *There exists a (at least twice differentiable) path* $\boldsymbol{q}_{\text{p}} : [0,\sigma] \to \mathcal{A}_{\text{free}}$ *such that* $\boldsymbol{q}_{\text{p}}(0) = q(0)$ *and* $\boldsymbol{q}_{\text{p}}(\sigma) \in Q_g$.

3 Main Results

We present here the proposed solution for Problem 1. Our methodology follows a two-layer approach, consisting of a robust trajectory-tracking control design and a higher-level sampling-based motion planner. Firstly, we use an adaptive control protocol that compensates for the uncertain dynamical parameters of the robot and forces the system to evolve in a funnel around a desired trajectory, whose size depends on the initial estimates of the dynamical parameters and the bound of the external disturbances. Secondly, we develop a geometric sampling-based motion planner that uses this funnel to find a collision free trajectory from the initial to the goal configuration. Intuitively, the robust control design helps the motion planner procedure, which does not have to take into account the complete dynamics (1). Section 3.1 presents the control design and 3.2 provides the motion planner.

3.1 Control Design

Control of uncertain dynamical systems, i.e., systems whose dynamical equations contain terms that are not known to the designer, has gained a large amount of attention during the last decades and numerous results have been developed for

a large variety of models. In the special case where these terms are constant, the control design is tackled by using adaptive feedback control ([24,25]).

Motivated by real life scenarios, this work considers parametric uncertainty of the robotic system dynamics, i.e., we consider that the masses and moments of inertia of the robotic system are unknown to the user/designer and hence cannot be used in the feedback control design. Let $\theta \in \mathbb{R}^l$, $l \in \mathbb{N}$, be a vector of terms involving these unknown dynamical parameters, which appear in the terms B, C, g of (1). Conveniently, the dynamics (1) can be linearly parameterized with respect to θ. More specifically, the terms that form the left-hand side of (1) can be written as [23, Ch. 7]

$$B(a)d + C(a,b)c + g(a) = Y(a,b,c,d)\theta, \tag{2}$$

$\forall a \in \mathbb{T}, b, c, d \in \mathbb{R}^{3n}$, where $Y(\cdot)$ is a matrix independent of θ and hence available for the control design. Let $q_\mathrm{d} \coloneqq [q_{\mathrm{d},tr}^\top, q_{\mathrm{d},r}^\top]^\top : [0, t_f] \to \mathbb{T}$ be a reference trajectory, with $q_{\mathrm{d},tr} \in \mathbb{R}^{n_{tr}}$ and $q_{\mathrm{d},r} \in [0, 2\pi)^{n_r}$ being its translational and rotational parts, respectively. Such a trajectory will be the output of the sampling-based motion planning algorithm that will be developed in the next section. We wish to design the control input u of (1) such that $q(t)$ converges close to $q_\mathrm{d}(t)$, despite the uncertainty in θ. Adaptive control is a widely known procedure when it comes to robotic manipulators with uncertainties (e.g., [26,23,22]). In this work we show how such a design can be used in the motion planning of a robotic system with dynamical uncertainties by developing a suitable variant of the standard adaptive control scheme.

We start by defining the appropriate error metric between $q = [q_{tr}^\top, q_r^\top]^\top$ and $q_\mathrm{d} = [q_{\mathrm{d},tr}^\top, q_{\mathrm{d},r}^\top]^\top$, which represents their distance. Regarding the translational part, we define the standard Euclidean error $e_{tr} \coloneqq q_{tr} - q_{\mathrm{d},tr}$. For the rotation part, however, the the same error $e_r \coloneqq q_r - q_{\mathrm{d},r}$ does not represent the minimum distance metric, since q_r evolves on the n_r-dimensional sphere, and its use might cause conservative or infeasible results in the planning layer. Hence, unlike standard adaptive control scheme for robotic manipulators, which drive the Euclidean difference $e_r(t)$ to zero (e.g., [22,26]), we use the chordal metric $d_C(x,y) \coloneqq 1 - \cos(x,y) \in [0,2]$, $\forall x, y \in [0, 2\pi)$, or $\bar{d}_C(x,y) \coloneqq \sum_{j\in\{1,\ldots,\ell\}} d_C(x_j, y_j)$ for vectors $x = [x_1, \ldots, x_\ell], y = [y_1, \ldots, y_\ell] \in [0, 2\pi)^\ell$. Nevertheless, note that rotational joints subject to upper and/or lower mechanical limits evolve in $\mathbb{R}$ rather than the unit circle and should hence be included in q_{tr} instead of q_r.

We are now ready to define a suitable distance metric for $\mathbb{T}$ as follows: for $x \coloneqq [x_{tr}^\top, x_r^\top]^\top$, $y \coloneqq [y_{tr}^\top, y_r^\top]^\top \in \mathbb{T}$ we define $d_\mathbb{T}$ as $d_\mathbb{T}(x,y) \coloneqq \|x_{tr} - y_{tr}\|^2 + \bar{d}_C(x_r, y_r)$. Note, however, that the chordal metric induces a limitation with respect to tracking on the unit sphere. Consider $d_C(q_{r_j}, q_{\mathrm{d},r_j}) = 1 - \cos(e_{r_j})$, where we further define $e_{r_j} \coloneqq q_{r_j} - q_{\mathrm{d},r_j}$ as the jth element of e_r, $j \in \{1, \ldots, n_r\}$. Differentiation yields $\dot{d}_C(q_{r_j}, q_{\mathrm{d},r_j}) = \sin(e_{r_j})\dot{e}_{r_j}$, $\forall j \in \{1, \ldots, n_r\}$, which is zero when $e_{r_j} = 0$ or $e_{r_j} = \pi$. The second case is an undesired equilibrium, which implies that the point $e_{r_j} = 0$ cannot be stabilized from *all* initial conditions using a continuous controller. This is an inherent property of dynamics on the

unit sphere due to topological obstructions ([27]). In the following, we devise a control scheme that, except for driving $q(t)$ to $q_{\mathrm{d}}(t)$, guarantees that $e_{r_j}(t) \neq \pi$, $\forall t \in (0, t_f]$, provided that $e_{r_j}(0) \neq 0$, $\forall j \in \{1, \dots, n_r\}$. To do that, we define the mapping

$$\mathsf{H}(x, y) \coloneqq \left[\tan\left(\frac{x_1 - y_1}{2}\right), \dots, \tan\left(\frac{x_{n_r} - y_{n_r}}{2}\right)\right]^\top \in \mathbb{R}^{n_r} \tag{3}$$

for vectors $x = [x_1, \dots, x_{n_r}]^\top$, $y = [y_1, \dots, y_{n_r}]^\top \in [0, 2\pi)^{n_r}$, as well as the signal $\eta_r \coloneqq \mathsf{H}(q_r, q_{\mathrm{d},r})$. Note that η_r is not defined when $e_{r_j} = \pi$ for some $j \in \{1, \dots, n_r\}$, which we exploit in the control design. Next, we define first the reference signals for $\dot{q}_{tr}, \dot{q}_r$ as $\alpha \coloneqq [\alpha_{tr}^\top, \alpha_r^\top]^\top$, with

$$\alpha_{tr} \coloneqq -K_{tr} e_{tr} + \dot{q}_{\mathrm{d},tr},\ \alpha_r \coloneqq [\alpha_{r_j}] \coloneqq \left[\dot{q}_{\mathrm{d},r_j} - k_{r_j} \cos\left(\frac{e_{r_j}}{2}\right) \sin\left(\frac{e_{r_j}}{2}\right)\right], \tag{4}$$

where $K_{tr} \in \mathbb{R}^{n_{tr} \times n_{tr}}$ is a symmetric positive definite gain matrix, $k_{r_j} > 0$ are positive gain constants, $\forall j \in \{1, \dots, n_r\}$, and $[\cdot]$ here denotes the stacked vector operator. Such velocities would achieve asymptotic stability for the system in hand and hence we define the velocity error signal $e_v = \dot{q} - \alpha$, which we also wish to drive to zero. Moreover, since the parameter vector θ cannot be used in the control design, we define the respective estimate $\hat{\theta} \in \mathbb{R}^l$, as well as the error $e_\theta \coloneqq \hat{\theta} - \theta \in \mathbb{R}^l$. The term $\hat{\theta}$ is time-varying and evolves according to the "adaptation" law, which will be designed along with the feedback control u. Note also that, although we use the term "estimate", the evolution of $\hat{\theta}$ is independent of any actual estimations of θ. To design the control scheme and guarantee that $e_{r_j}(t) \neq \pi$, we define the positive definite Lyapunov function

$$V(x, t) \coloneqq \tfrac{1}{2}\|e_{tr}\|^2 + \tfrac{1}{2}\|\eta_r\|^2 + \tfrac{1}{2\underline{k}_v} e_v^\top B(q) e_v + \tfrac{1}{2} e_\theta^\top \Gamma^{-1} e_\theta,$$

where $x \coloneqq [e_{tr}^\top, \eta_r^\top, e_v^\top, e_\theta]^\top$, $\underline{k}_v$ is a positive constant, and $\Gamma \in \mathbb{R}^{\ell \times \ell}$ is a positive definite constant matrix, both to be used in the control design. We assume that $e_{r_j}(0) \neq \pi$ and hence $V(x(0), 0)$ is bounded. Differentiation of V, use of Property 1 and the dynamics' linear parameterization (2) and substitution of (4) as well as $\dot{q} = \alpha + e_v$ yields

$$\dot{V} = -e_{tr} K_{tr} e_{tr} - \eta_r^\top K_r \eta_r + \tfrac{1}{\underline{k}_v} e_v^\top (u + e_x - Y_\alpha \theta - d(t)) + e_\theta^\top \Gamma^{-1} \dot{\hat{\theta}},$$

where $K_r \coloneqq \mathrm{diag}\{k_{r_1}, \dots, k_{r_{n_r}}\}$, $e_x \coloneqq \left[e_{tr}^\top, \frac{\tan\left(\frac{e_{r_1}}{2}\right)}{\cos\left(\frac{e_{r_1}}{2}\right)^2}, \dots, \frac{\tan\left(\frac{e_{r_{n_r}}}{2}\right)}{\cos\left(\frac{e_{r_{n_r}}}{2}\right)^2}\right]^\top$, and $Y_\alpha \coloneqq Y(q, \dot{q}, \alpha, \dot{\alpha})$, with $Y(\cdot)$ as given in (2). We design next the control and adaptation law u, $\dot{\hat{\theta}}$, respectively, as

$$u = Y_\alpha \hat{\theta} - K_v e_v - e_x, \quad \dot{\hat{\theta}} = -\Gamma \left(\frac{1}{\underline{k}_v} Y_\alpha^\top e_v + \sigma_\theta \hat{\theta}\right). \tag{5}$$

By substituting these in $\dot{V}$ and setting $\underline{k}_v$ as the minimum eigenvalue of K_v, we obtain after straightforward algebraic manipulations

$$\dot{V} \leq -\underline{k}_{tr}\|e_{tr}\|^2 - \underline{k}_r\|\eta_r\|^2 - \tfrac{1}{2}\|e_v\|^2 - \tfrac{\sigma_\theta}{2}\|e_\theta\|^2 + d_x,$$

where $\underline{k}_{tr}$, $\underline{k}_r$ are the minimum eigenvalues of the matrices K_{tr} and K_r, respectively, and $d_x \coloneqq \frac{\bar{d}^2}{2\underline{k}_{v_1}^2} + \frac{\sigma_\theta}{2}\|\theta\|^2$.

Therefore, $\dot{V}$ is negative when $\|e_{tr}\| \geq \sqrt{\frac{d_x}{\underline{k}_{tr}}}$, or $\|\eta_r\| \geq \sqrt{\frac{d_x}{\underline{k}_r}}$, or $\|e_v\| \geq \sqrt{2d_x}$, or $\|e_\theta\| \geq \sqrt{\frac{2d_x}{\sigma_\theta}}$. By using Theorem 4.18 of [28], we conclude that there exists a finite T such that the state will enter the set $\Omega_x \coloneqq \{x \in \mathbb{R}^{2n+\ell} : \|e_{tr}(t)\| \leq \sqrt{\frac{d_x}{\underline{k}_{tr}}}, \|\eta_r(t)\| \leq \sqrt{\frac{d_x}{\underline{k}_r}}, \|e_v(t)\| \leq \sqrt{2d_x}, \|e_\theta(t)\| \leq \sqrt{\frac{2d_x}{\sigma_\theta}}\}$ and remain there, i.e., $x(t) \in \Omega_x$, $\forall t \geq T$. If x starts in Ω_x, i.e., $x(0) \in \Omega$, then $x(t) \in \Omega_x$, $\forall t \in [0, t_f]$. If not, then by using T as the time when $x(t)$ enters Ω_x, it holds that $\dot{V}(x(t)) \leq 0$, and hence $V(x(t)) \leq V(x(0))$ and $\|e_{tr}(t)\| \leq 2V(x(0))$, $\|\eta_r(t)\| \leq V(x(0))$, $\forall t \in [0, T]$. Therefore, it holds that $\|e_{tr}(t)\| \leq \bar{e}_{tr} \coloneqq \max\left\{2V(x(0)), \sqrt{\frac{d_x}{\underline{k}_{tr}}}\right\}$, $\|\eta_r(t)\| \leq \bar{\eta}_r \coloneqq \max\left\{V(x(0)), \sqrt{\frac{d_x}{\underline{k}_r}}\right\}$, $\forall t \in [0, t_f]$. Moreover, since Ω_x is a compact set and $\dot{V} \leq 0$ outside of it, we conclude that $V(x(t))$ stays always bounded, for all $t \in [0, t_f]$. This implies that $\eta_r(t)$ also remains bounded and $e_{r_j}(t) \neq \pi$, $\forall t \in [0, t_f], j \in \{1, \ldots, n_r\}$. Note that this stems from the term e_x in the control law (5), which is infinity when $e_{r_j} = \pi$. Intuitively, as e_{r_j} approaches to π, for some $j \in \{1, \ldots, n_r\}$, the control law u increases to infinity to prevent that from happening. Nevertheless, the ultimate boundedness of the signals in Ω_x implies that $u(t)$, $\hat{\theta}(t)$, and $\dot{\hat{\theta}}(t)$ remain also bounded, $\forall t \in [0, t_f]$. The aforementioned discussion is summarized in the following theorem:

Theorem 1. *Consider the dynamics* (1)*, a reference trajectory* $q_\mathrm{d} : [0, t_f] \to \mathbb{T}$*, as well as the constant* $V_0 \coloneqq \frac{1}{2}\|e_{tr}(0)\|^2 + \|\eta_r(0)\|^2 + \frac{1}{2\underline{k}_v} e_v(0)^\top B(q(0)) e_v(0) + \frac{1}{2} e_\theta(0)^\top \Gamma^{-1} e_\theta(0)$*. Then, if* $e_{r_j}(0) \neq \pi$*,* $\forall j \in \{1, \ldots, n_r\}$*, the control protocol* (5) *guarantees that*

$$\|e_{tr}(t)\| \leq \bar{e}_{tr} \coloneqq \max\left\{2V_0, \sqrt{\frac{d_x}{\underline{k}_{tr}}}\right\}, \quad \|\eta_r(t)\| \leq \bar{\eta}_r \coloneqq \max\left\{V_0, \sqrt{\frac{d_x}{\underline{k}_r}}\right\}, \quad (6)$$

as well as the boundedness of all closed-loop signals, $\forall t \in [0, t_f]$*.*

Note that the disturbance term $d(t)$ prohibits the system from achieving asymptotic convergence, i.e., $\lim_{t\to\infty}(q(t) - q_\mathrm{d}(t)) = 0$. Nevertheless, Theorem 1 establishes a funnel around the desired trajectory q_d where the state $q(t)$ will evolve in. This funnel will be used as clearance in the motion planner of the subsequent section to derive a collision-free path to the goal region. Note however, that this funnel cannot be accurately known by the user/designer, since V_0 cannot be accurately known (the terms $B(q(0))$ and $e_\theta(0)$ contain the unknown terms θ). Lower and upper bounds of θ can be obtained, however,

since these involve mass and moments of inertia, which can be estimated by the geometry and the material of the links/motors. Hence, one can obtain an upper bound on V_0. On the same note, a conservative estimate of d_x, appearing in (6), can be obtained by estimating an upper bound of $d(t)$ (e.g., by testing suitable trajectories on the robot) and using the aforementioned upper bound of θ. Therefore, we can obtain an overestimate of the bounds in (6), which will be used in the motion planner of the next section. These bounds can be tightened by appropriate tuning of the gain constants, as elaborated in the next remark.

Remark 1. The collision-free geometric trajectory q_d of the motion planner will connect the initial condition $q(0)$ to the goal and hence it is reasonable to enforce $q_\mathrm{d}(0) = q(0)$. By also reasonably assuming that that $\dot{q}(0) = 0$, V_0 from Theorem 1 becomes $V_0 = \frac{1}{2\underline{k}_v}\dot{q}_\mathrm{d}(0)B(q(0))\dot{q}_\mathrm{d}(0)+\frac{1}{2}e_\theta(0)^\top \Gamma^{-1}e_\theta(0)$, which can be rendered arbitrarily small by choosing large values for the control gains $\underline{k}_v$ and Γ. In the same vein, choosing large values for $\underline{k}_v$, $\underline{k}_{tr}$, and $\underline{k}_r$ shrinks the constants $\sqrt{\frac{d_x}{\underline{k}_{tr}}}$ and $\sqrt{\frac{d_x}{\underline{k}_r}}$, respectively. Therefore, the size of the funnel dictated by (6) can become smaller by appropriate gain tuning. This will lead to less conservative solutions for the motion planner of the next section, as will be clarified in the next subsection. Nevertheless, it should be noted that too large gains might result in excessive control inputs that cannot be realized by the actuators in realistic systems. Finally, note that the incorporation of $\underline{k}_v$ in the adaptation law (5), which is not common in standard adaptive control techniques, has been included to create an extra degree of freedom for reducing the value of V_0. This, along with the tracking using the chordal metric d_C for the rotation part, constitute the differences of the proposed control scheme with respect to standard adaptive control for uncertain robotic systems.

3.2 Motion Planner

We describe here the construction of the sampling-based motion planner, referred to as Bounded-RRT or B-RRT, that drives the robot from an initial state to the goal, which follows similar steps as the standard geometric RRT algorithm. Before presenting the algorithm we define the extended-free space, which will be used to integrate the results from the feedback control of the previous subsection. In order to do that, we define first the open polyhedron as

$$\mathcal{P}(q,\delta) \coloneqq \{y = [y_{tr}^\top, y_r^\top]^\top \in \mathbb{T} : \|y_{tr} - q_{tr}\| < \delta_{tr}, \|\mathsf{H}(y_r, q_r)\| < \delta_r\}, \quad (7)$$

for $q = [q_{tr}^\top, q_r^\top]^\top \in \mathbb{T}$ and $\delta = (\delta_{tr}, \delta_r) \in \mathbb{R}^2$, where $\mathsf{H}(\cdot)$ is the metric introduced in (3). We define now the δ-extended free space $\bar{\mathcal{A}}_\mathrm{free}(\delta) \coloneqq \{q \in \mathbb{T} : \bar{\mathcal{A}}(q,\delta) \cap \mathcal{O} = \emptyset\}$, where $\bar{\mathcal{A}}(q,\delta) \coloneqq \bigcup_{x\in\mathcal{P}(q,\delta)} \mathcal{A}(x)$. Note that $\bar{\mathcal{A}}_\mathrm{free}((\delta_{1_{tr}}, \delta_{1_r})) \subseteq \bar{\mathcal{A}}_\mathrm{free}((\delta_{2_{tr}}, \delta_{2_r}))$ if $\delta_{1_{tr}} \geq \delta_{2_{tr}}$ and/or $\delta_{1_r} \geq \delta_{2_r}$.

Remark 2. Since $\mathcal{A}_\mathrm{free}$ is open, there exist positive constants δ_{tr}, δ_r such that $Q_g \subset \bar{\mathcal{A}}_\mathrm{free}((\delta_{tr}, \delta_r))$ and the feasible path $\boldsymbol{q}_\mathrm{p}$ from Assumption 1 satisfies $\boldsymbol{q}_\mathrm{p}(\tau) \in \bar{\mathcal{A}}_\mathrm{free}((\delta_{tr}, \delta_r))$, $\forall \tau \in [0, \sigma]$.

The control scheme of the previous subsection guarantees that the robot can track a trajectory within the bounds (6). In other words, given a desired trajectory signal $q_{\mathrm{d}} : [t_0, t_f] \to \mathbb{T}$, the control algorithm (5) guarantees that $q(t) \in \bar{\mathcal{A}}_{\mathrm{free}}((\bar{e}_{tr}, \bar{\eta}_r))$, $\forall t \in [t_0, t_f]$, with $\bar{e}_{tr}$, $\bar{\eta}_r$ as defined in (6). Hence, the motion planner developed here takes that into account by producing trajectories that belong to the extended free space $\bar{\mathcal{A}}_{\mathrm{free}}((\bar{e}_{tr}, \bar{\eta}_r))$ [3]. The respective algorithm is presented in Algorithm 1. It is a variant of the standard RRT algorithm. The main difference, which constitutes the key point of the algorithm, is the procedure that aims to find a collision-free trajectory from a node on the tree towards the sampled point. In particular, the sampling of new nodes-points as well as the collision checker of the path between two nodes are carried out with respect to the extended free space $\bar{\mathcal{A}}_{\mathrm{free}}((\bar{e}_{tr}, \bar{\eta}_r))$. Moreover, the motion planner does not need to integrate the system dynamics (1) with sampled input values in order to design a feasible and collision-free robot trajectory. Instead, we use the established evolution funnel of Section (3.1) to design a collision-free trajectory for the robot, without involving the dynamics. This implies that the motion planner is purely geometrical.

Algorithm 1 B-RRT

procedure TREE
  $\mathcal{V} \leftarrow \{q(0)\}$; $\mathcal{E} \leftarrow \emptyset$; $i \leftarrow 0$
  while $i < N$ **do**
    $\mathcal{G} \leftarrow (\mathcal{V}, \mathcal{E})$;
    $q_{\mathrm{rand}} \leftarrow \mathsf{Sample}(i)$; $i \leftarrow i + 1$;
    $q_{\mathrm{nearest}} \leftarrow \mathsf{Nearest}(\mathcal{G}, q_{\mathrm{rand}})$;
    $q_{\mathrm{new}} \leftarrow \mathsf{Steer}(q_{\mathrm{nearest}}, q_{\mathrm{rand}})$;
    if $\mathsf{ObstacleFree}(q_{\mathrm{nearest}}, q_{\mathrm{new}})$ **then**
      $V \leftarrow V \cup \{q_{\mathrm{new}}\}$; $\mathcal{E} \leftarrow \mathcal{E} \cup \{(q_{\mathrm{nearest}}, q_{\mathrm{new}})\}$;

The functions that appear in Algorithm 1 are the following:

- $\mathsf{Sample}(i)$: Samples q_{rand} from a uniform distribution in the extended free space $\bar{\mathcal{A}}_{\mathrm{free}}((\bar{e}_{tr}, \bar{\eta}_r))$, where $\bar{e}_{tr}, \bar{\eta}_r$ are the constants from (6) that define the funnel polyhedron $\mathcal{P}(q_{\mathrm{d}}(t), (\bar{e}_{tr}, \bar{\eta}_r))$ (see (7)) around a reference trajectory $q_{\mathrm{d}}(t)$ that $q(t)$ can evolve in.
- $\mathsf{Nearest}(G, q)$: Finds the node q_{nearest} in the tree such that $d_{\mathbb{T}}(q_{\mathrm{nearest}}, q) = \min_{z \in \mathcal{V}} d_{\mathbb{T}}(z, q)$.
- $\mathsf{Steer}(q, z)$: Computes a point q_{new} lying on the straight line from z to q such that $d_{\mathbb{T}}(q, q_{\mathrm{new}}) = \epsilon$, where ϵ is a tuning constant that represents the incremental distance from q that to q_{new}.
- $\mathsf{ObstacleFree}(q, z)$: Checks whether the path $X_{\mathrm{Line}} : [0, \sigma] \to \mathbb{T}$, for some positive σ, from q to z is collision free with respect to the extended free space, i.e., check whether $q' \in \bar{\mathcal{A}}_{\mathrm{free}}((\bar{e}_{tr}, \bar{\eta}_r))$, $\forall q' \in X_{\mathrm{Line}}$.

[3] We keep the same notation $(\bar{e}_{tr}, \bar{\eta}_r)$, although only upper bounds of these values can be actually estimated and hence used by the planner

The difference hence of B-RRT with respect to the standard RRT algorithm is the use of the extended free space $\bar{\mathcal{A}}_{\text{free}}((\bar{e}_{tr}, \bar{\eta}_r))$ in the procedures of sampling new points (function Sample) and checking collisions of the path between two nodes (function ObstacleFree). As stated before, this stems from the control design of the previous section, which guarantees that the robot trajectory will evolve in $\bar{\mathcal{A}}_{\text{free}}((\bar{e}_{tr}, \bar{\eta}_r))$ with respect to a desired trajectory q_{d}.

The resulting smooth (at least twice cont. different.) path is endowed with time constraints to derive a timed trajectory $q_{\text{d}} : [0, t_f] \to \bar{\mathcal{A}}_{\text{free}}((\bar{e}_{tr}, \bar{\eta}_r))$, for some $t_f > 0$, which is given as the desired trajectory input to the control protocol designed in the previous section. The actual trajectory of the system $q(t)$ is guaranteed to track $q_{\text{d}}(t)$ in the funnel defined by $\bar{e}_{tr}, \bar{\eta}_r$. Since these bounds are taken into account in the design of the trajectory q_{d} by Algorithm 1, the system will remain collision free. Note also that t_f and hence the velocity of the formed trajectory q_{d} is chosen by the user. Therefore, the robot can execute the respective path in a predefined time interval.

The probabilistic completeness of the algorithm is stated in the next theorem.

Theorem 2. *Under Assumption 1 and for sufficiently high gains $\underline{k}_v$, $\underline{k}_{tr}$, $\underline{k}_r$, as introduced in eq.* (4)*,* (5)*, Algorithm 1 is probabilistically complete.*

Proof. Assumption 1 and Remark 2 imply that there exist positive δ_{tr} and δ_r and (at least) one twice differentiable path $\boldsymbol{q}_{\text{p}} : [0, \sigma] \to \bar{\mathcal{A}}_{\text{free}}((\delta_{tr}, \delta_r))$ connecting q_0 and Q_g. As stated in Remark 1, by increasing the values of the control gains $\underline{k}_v$, $\underline{k}_{tr}$, $\underline{k}_r$, one can decrease the constants $\bar{e}_{tr}$, $\bar{\eta}_{tr}$ from eq. (6) such that $\bar{e}_{tr} < \delta_{tr}$, $\bar{\eta}_r < \delta_r$. Hence, Q_g satisfies $Q_g \subset \bar{\mathcal{A}}_{\text{free}}((\delta_{tr}, \delta_r)) \subset \bar{\mathcal{A}}_{\text{free}}((\bar{e}_{tr}, \bar{\eta}_r))$ and the feasible path satisfies $\boldsymbol{q}_{\text{p}}(\tau) \in \bar{\mathcal{A}}_{\text{free}}((\delta_{tr}, \delta_r)) \subset \bar{\mathcal{A}}_{\text{free}}((\bar{e}_{tr}, \bar{\eta}_r))$, $\forall \tau \in [0, \sigma]$, which guarantees the feasibility of Algorithm 1. Next, by following similar arguments with Lemma 2 of [29], one can prove that for any $q \in \bar{\mathcal{A}}_{\text{free}}((\bar{e}_{tr}, \bar{\eta}_r))$ and $\epsilon > 0$, it holds that $\lim_{i \to \infty} \mathbb{P}(D_{i,q} < \epsilon)$, where $D_{i,q}$ is the random variable associated with the minimum distance of the tree $\mathcal{G}$ to the point q (in terms of $d_{\mathbb{T}}$), after iteration i. Hence, the vertices $\mathcal{V}$ of $\mathcal{G}$ converge to the sampling distribution in $\bar{\mathcal{A}}_{\text{free}}((\bar{e}_{tr}, \bar{\eta}_r))$, which is assumed to be uniform. Therefore, since $\boldsymbol{q}_{\text{p}}$ lies in $\bar{\mathcal{A}}_{\text{free}}((\bar{e}_{tr}, \bar{\eta}_r))$, a subset of $\mathcal{V}$ converges to it and the proof follows.

3.3 Collision Checking in $\bar{\mathcal{A}}_{\text{free}}(\bar{e}_{tr}, \bar{\eta}_r)$

The proposed feedback control scheme guarantees that $q(t) \in \mathcal{P}(q_{\text{d}}(t), (\bar{e}_{tr}, \bar{\eta}_r))$ for any trajectory $q_{\text{d}}(t)$, formed by the several line segments X_{Line} that connect the nodes in $\mathcal{V}$ sampled in Algorithm 1. Therefore, checking whether the points $q_s \in X_{\text{Line}}$ belong to $\mathcal{A}_{\text{free}}$, as in standard motion planners [5,30], is not sufficient. For each such point $q_s \in X_{\text{Line}}$, one must check whether $z \in \mathcal{A}_{\text{free}}$, $\forall z \in \mathcal{P}(q_s, (\bar{e}_{tr}, \bar{\eta}_r))$, which is equivalent to checking if $q_s \in \bar{\mathcal{A}}_{\text{free}}(\bar{e}_{tr}, \bar{\eta}_r)$. There are two procedures that we use for that. Firstly, for each q_s, a finite number of points z can be sampled from a uniform distribution in $\mathcal{P}(q_s, (\bar{e}_{tr}, \bar{\eta}_r))$ and separately checked for collision. Then, for a sufficiently high number of such samples, and assuming a certain "fat"-structure of the workspace obstacles (e.g., there

are no long and skinny obstacles such as wires, cables and tree branches, etc., see ([31]) for more details), this approach can be considered to be complete, i.e., the resulting path will belong to the extended free space $\mathcal{A}_{\text{free}}$. Secondly, we calculate the limit poses of each link of the robot, based on the lower and upper bounds by the joints that affect it, as defined by $(\bar{e}_{tr}, \bar{\eta}_r)$. Subsequently, we compute the convex hull of these limit poses, which is expanded by an appropriate constant to yield an over-approximation of the swept volume of the potential motion of the link, as described in [32]. The resulting shape is then checked for collisions for each link separately.

4 Experimental Results

This section presents experimental results for a UR5 robot, which consists of 6 rotational degrees of freedom (see Fig. 1), using the V-REP environment ([21]). We assume that the first joint is free to move on the unit circle, i.e., $q_{r_1} \in [0, 2\pi)$, whereas the rest of the joints are restricted to $[-\pi, \pi]$ to avoid problematic configurations. We consider that the robot end-effector has to sequentially navigate

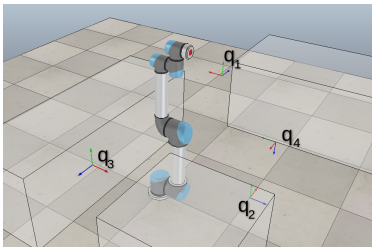

Fig. 1: A UR5 robotic arm in an obstacle-cluttered environment with 4 targets.

from its initial configuration $q_0 = [0, 0, 0, 0, 0, 0]$ to the following four target points (depicted in Fig. 1):

- Target 1: $T_1 = (-0.15, -0.475, 0.675)$ and orientation $(\frac{\pi}{2}, 0, 0)$, which yields the configuration $q_1 = [-0.07, -1.05, 0.45, 2.3, 1.37, -1.33]^\top$.
- Target 2: $T_2 = (-0.6, 0, 2.5)$ and orientation $(0, -\frac{\pi}{2}, -\frac{\pi}{2})$, which yields the configuration $q_2 = [1.28, 0.35, 1.75, 0.03, 0.1, -1.22]^\top$
- Target 3: $T_3 = (-0.025, 0.595, 0.6)$ and orientation $(-\frac{\pi}{2}, 0, \pi)$, which yields the configuration $q_3 = [-0.08, 0.85, -0.23, 2.58, 2.09, -2, 36]^\top$
- Target 4: $T_4 = (-0.525, -0.55, 0.28)$ and orientation $(\pi, 0, -\frac{\pi}{2})$, which yields the configuration $q_4 = [-0.7, -0.76, -1.05, -0.05, -3.08, 2.37]\top$

Regarding the collision checking in $\bar{\mathcal{A}}_{\text{free}}(\bar{e}_{tr}, \bar{\eta}_r)$ of the B-RRT algorithm, we check a finite number of samples around each point of the resulting trajectory q_{d} for collision. We run B-RRT with 10 and 50 such samples and we compared the results to a standard geometric RRT algorithm in terms of time per number of nodes. The results for 30 runs of the algorithms are given in Fig. 2 for the

four paths, in logarithmic scale. One can notice that the average nodes created do not differ significantly among the different algorithms. As expected, however, B-RRT requires more time than the standard geometric RRT algorithm, since it checks the extra samples in $\bar{\mathcal{A}}_{\text{free}}(\bar{e}_{tr}, \bar{\eta}_r)$ for collision. One can also notice that the time increases with the number of samples. However, more samples imply greater coverage of $\bar{\mathcal{A}}_{\text{free}}(\bar{e}_{tr}, \bar{\eta}_r)$ and hence the respective solutions are more likely to be complete with respect to collisions.

Since, in contrast to the standard geometric RRT, B-RRT implicitly takes into account the robot dynamics through the designed tracking control scheme and the respective extended free space $\bar{\mathcal{A}}_{\text{free}}(\bar{e}_{tr}, \bar{\eta}_r)$, we compare the results to a standard kinodynamic RRT algorithm that simulates forward the robot dynamics, assuming known dynamical parameters. In particular, we run the algorithm only for the first two joints, with initial and goal configurations at $(0, 0)$ and $(-\frac{\pi}{18}, \frac{\pi}{4})$ rad, respectively, and keep the other joints fixed at 0. For the forward simulation of the respective dynamics we chose a sampling step of 10^{-3} sec and total simulation time 30 sec for each constant control input. The termination threshold distance was set to 0.25 (with respect to the distance $d_{\mathbb{T}}$), i.e., the algorithm terminated when the forward simulation reached a configuration closer than 0.25 units to the goal configuration. The results for 10 runs of the algorithm are depicted in Fig. 3, which provides the execution time and number of nodes created in logarithmic scale. Note that, even for these simple case (planning for only two joints), the execution time is comparable to the B-RRT case of 50 samples in the fourth path scenario $q_3 \to q_4$. As pointed out in Section 1, this is justified by the fact that the inputs are randomized as well as the complex dynamics of the considered robotic system.

Next, we illustrate the motion of the robot through the four target points via the control design of Section 3.1. For each sub-path ($q_i \to q_{i+1}$, $\forall i \in \{0, 1, 2, 3\}$) we fit a smooth timed trajectory $q_{\text{d}}(t) = [q_{\text{d},1}(t), \dots, q_{\text{d},6}(t)]^\top$ on the generated nodes, whose total time duration depends on the distance between successive nodes. The output sequence of points along with the interpolated trajectories are depicted in Fig. 4, where we have added an extra time offset after each $q_{\text{d}}(t)$. Note from the figure that the trajectories do not exactly fit the generated nodes, without however affecting the safety of the proposed scheme.

The estimates of the masses and inertias of the robot links and rotors, composing $\hat{\theta}$, were initialized at 60% of the actual values. Morever, in view of (6), we aim to impose an upper bound of 0.1 rad for each $|q_{r_j} - q_{\text{d},j}|$, $\forall j \in \{1, \dots, 6\}$. To that end, we choose the control gains as $k_{r_1} = \cdots = k_{r_6} = 0.005$, $K_v = \text{diag}\{[35, 65, 45, 20, 10, 0.5]\}$, and $\Gamma = 50\text{diag}\{\hat{\theta}(0)\}$, where the diag$\{z\}$ operator returns the diagonal $n \times n$ matrix whose diagonal values are the elements of the vector $z \in \mathbb{R}^n$. The results are depicted in Fig. 5 (a), which shows the error values $e_{r_j}(t) = q_{r_j}(t) - q_{\text{d},j}(t)$, $\forall j \in \{1, \dots, 6\}$, and all four paths. One can verify that the error values stay always bounded in the region $(-0.1, 0.1)$ rad, achieving thus the desired performance. For comparison purposes, we also simulate a PID controller of the form $u = -K_1 e_x - K_2(\dot{q} - \dot{q}_{\text{d}}) - K_3 \int e_x(\tau) d\tau$, where $K_1 = \text{diag}\{100, 1000, 1000, 100, 1, 1\}$, and $K_2 = K_3 = I_6$ are positive definite

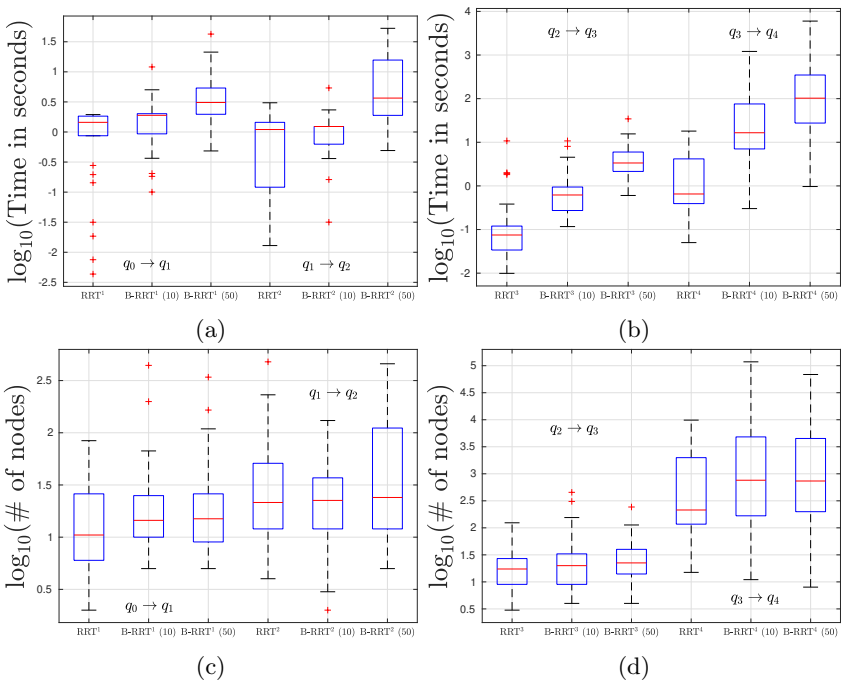

Fig. 2: Box plots showing the execution time (top) and the nodes (bottom) created of the three algorithms (in logarithmic scale) for the four paths (organized in two groups of two (left and right)); '+' indicate the outliers.

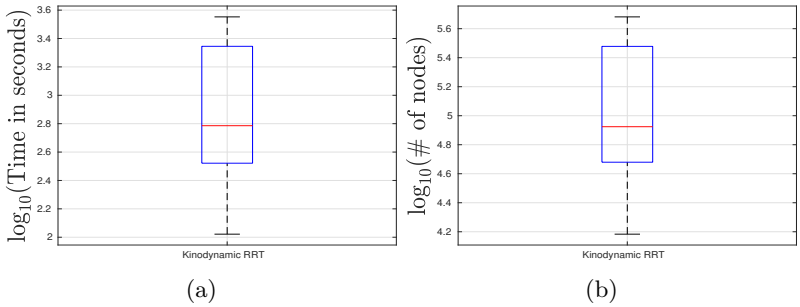

Fig. 3: Box plots showing the execution time (a) and number of nodes (b) created for the kinodynamic RRT in logarithmic scale (for the first two joints and the path $(0,0) \rightarrow (-\frac{\pi}{18}, \frac{\pi}{4})$).

gain matrices. The errors $e_{r_j}(t) = q_{r_j}(t) - q_{\mathrm{d},j}(t)$, $\forall j \in \{1,\ldots,6\}$, for the four paths are shown in Fig. 5 (b). Note that they exceed the interval $(-0.1, 0.1)$, which defined the clearance in the B-RRT algorithm, jeopardizing hence the actual trajectory of the robot. We acknowledge, nevertheless, that appropriate yet

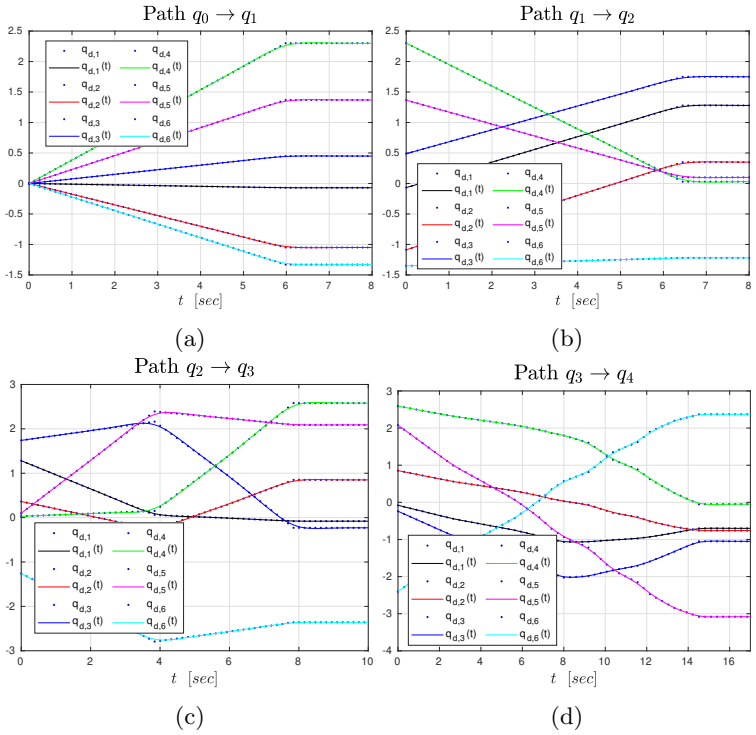

Fig. 4: The output sequence of points and the respective trajectories for the four paths.

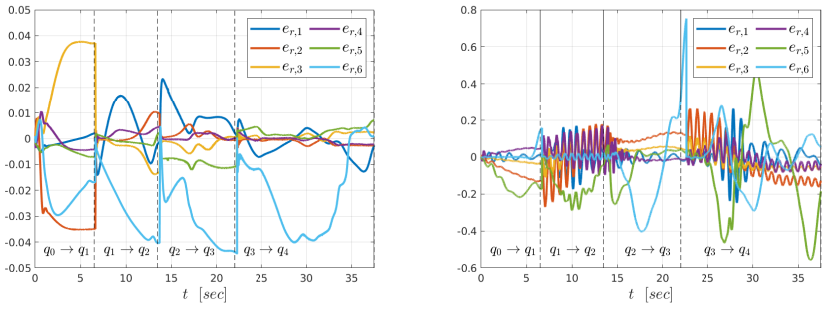

Fig. 5: The error values $e_{r_j}(t) = q_{r_j}(t) - q_{\mathrm{d},j}(t)$ for the adaptive controller (a) and the PID one (b).

tedious gain tuning might yield better results for a PID control scheme. A video illustrating the robot trajectory using the two control laws can be found here: https://youtu.be/y7bCoUoTlPA.

5 Discussion

We developed a two-layer motion planning scheme for 2nd-order systems with dynamic uncertainties. We integrated an adaptive control scheme that guarantees trajectory tracking withing certain bounds with an appropriately designed geometric RRT-based motion planner. The proposed methodology can solve efficiently the motion planning problem without neglecting the system dynamics. A drawback of the proposed scheme is the fact that the bounds (6) that the system trajectory evolves in (with respect to a desired trajectory) are not accurately known. Overestimates of the dynamical parameters as well as the external disturbances need to be obtained and gain tuning might be needed in order to achieve the desired bounds. Future efforts will be devoted in developing more sophisticated feedback control algorithms that provide *known* bounds as well as extending the B-RRT algorithm to account for optimality.

References

1. LaValle, S.M.: Planning algorithms. Cambridge university press (2006)
2. Choset, H.M., Hutchinson, S., Lynch, K.M., Kantor, G., Burgard, W., Kavraki, L.E., Thrun, S.: Principles of robot motion: theory, algorithms, and implementation. MIT press (2005)
3. Barraquand, J., Latombe, J.C.: Robot motion planning: A distributed representation approach. The International Journal of Robotics Research **10**(6), 628–649 (1991)
4. Rimon, E., Koditschek, D.E.: Exact robot navigation using artificial potential functions. IEEE Transaction on Robotics and Automation **8**, 501–518 (1992)
5. Kavraki, L.E., Svestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE transactions on Robotics and Automation **12**(4), 566–580 (1996)
6. Hsu, D., Latombe, J.C., Motwani, R.: Path planning in expansive configuration spaces. Proceedings of International Conference on Robotics and Automation **3**, 2719–2726 (1997)
7. LaValle, S.M.: Rapidly-exploring random trees: A new tool for path planning (1998)
8. LaValle, S.M., Kuffner Jr, J.J.: Randomized kinodynamic planning. The international journal of robotics research **20**(5), 378–400 (2001)
9. Şucan, I.A., Kavraki, L.E.: Kinodynamic motion planning by interior-exterior cell exploration. Algorithmic Foundation of Robotics VIII pp. 449–464 (2009)
10. Vidal, E., Moll, M., Palomeras, N., Hernández, J.D., Carreras, M., Kavraki, L.E.: Online multilayered motion planning with dynamic constraints for autonomous underwater vehicles. 2019 International Conference on Robotics and Automation (ICRA) pp. 8936–8942 (2019)
11. Tedrake, R., Manchester, I.R., Tobenkin, M., Roberts, J.W.: Lqr-trees: Feedback motion planning via sums-of-squares verification. The International Journal of Robotics Research **29**(8), 1038–1052 (2010)
12. Reist, P., Preiswerk, P., Tedrake, R.: Feedback-motion-planning with simulation-based lqr-trees. The International Journal of Robotics Research **35**(11), 1393–1416 (2016)

13. Wu, A., Sadraddini, S., Tedrake, R.: R3t: Rapidly-exploring random reachable set tree for optimal kinodynamic planning of nonlinear hybrid systems (2019)
14. Majumdar, A., Tedrake, R.: Funnel libraries for real-time robust feedback motion planning. The International Journal of Robotics Research **36**(8), 947–982 (2017)
15. Du Toit, N.E., Burdick, J.W.: Probabilistic collision checking with chance constraints. IEEE Transactions on Robotics **27**(4), 809–815 (2011)
16. Pairet, È., Hernández, J.D., Lahijanian, M., Carreras, M.: Uncertainty-based online mapping and motion planning for marine robotics guidance. IEEE/RSJ International Conference on Intelligent Robots and Systems pp. 2367–2374 (2018)
17. Bry, A., Roy, N.: Rapidly-exploring random belief trees for motion planning under uncertainty. 2011 IEEE international conference on robotics and automation pp. 723–730 (2011)
18. Agha-Mohammadi, A.A., Chakravorty, S., Amato, N.M.: Firm: Sampling-based feedback motion-planning under motion uncertainty and imperfect measurements. The International Journal of Robotics Research **33**(2), 268–304 (2014)
19. Le Ny, J., Pappas, G.J.: Sequential composition of robust controller specifications. IEEE International Conference on Robotics and Automation pp. 5190–5195 (2012)
20. Luders, B.D., Karaman, S., Frazzoli, E., How, J.P.: Bounds on tracking error using closed-loop rapidly-exploring random trees. Proceedings of the 2010 American Control Conference pp. 5406–5412 (2010)
21. Rohmer, E., Singh, S.P., Freese, M.: V-rep: a versatile and scalable robot simulation framework. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2013)
22. Tomei, P.: Robust adaptive control of robots with arbitrary transient performance and disturbance attenuation. IEEE transactions on automatic control **44**(3), 654–658 (1999)
23. Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G.: Robotics: modelling, planning and control. Springer Science & Business Media (2010)
24. Krstic, M., Kanellakopoulos, I., Kokotovic, P.: Nonlinear and Adaptive Control Design. Publisher: Wiley New York (1995)
25. Lavretsky, E., Wise, K.: Robust and Adaptive Control: With Aerospace Applications. Springer Science and Business Media (2012)
26. Slotine, J.J.E., Li, W.: On the adaptive control of robot manipulators. The international journal of robotics research **6**(3), 49–59 (1987)
27. Bhat, S., Bernstein, D.: A Topological Obstruction to Continuous Global Stabilization of Rotational Motion and the Unwinding Phenomenon. Systems and Control Letters **39**(1), 63–70 (2000)
28. Khalil, H.K.: Nonlinear Systems. Prentice Hall (2002)
29. Kuffner, J.J., LaValle, S.M.: Rrt-connect: An efficient approach to single-query path planning. Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065) **2**, 995–1001 (2000)
30. Karaman, S., Frazzoli, E.: Incremental sampling-based optimal motion planning. Robotics:Science and Systems (2010)
31. van der Stappen, A.F., Halperin, D., Overmars, M.H.: The complexity of the free space for a robot moving amidst fat obstacles. Computational Geometry **3**(6), 353–373 (1993)
32. Schulman, J., Duan, Y., Ho, J., Lee, A., Awwal, I., Bradlow, H., Pan, J., Patil, S., Goldberg, K., Abbeel, P.: Motion planning with sequential convex optimization and convex collision checking. The International Journal of Robotics Research **33**(9), 1251–1270 (2014)