Long Time Horizon Steering for Sample-Based Planning of Dynamic Systems

T. M. Caldwell and N. Correll

***Abstract*—We propose a novel method for performing steering and distance computations in sample-based planning for dynamic systems that is applicable to RRT-like motion planning algorithms. In order to maintain the differential constraint to the dynamics, these operations must solve optimal control problems which are expensive to compute. Similar to conventional approaches, we approximate distance and steering through system linearization and solving linear quadratic regulation and tracking problems. A number of improvements are made. We linearize about zero-control trajectories as opposed to single states. We employ a trust region that assesses a maximal time horizon for each linearization online instead of choosing a conservative one a priori. We show how these methods can be implemented efficiently through precomputation and caching redundancies in the linear optimal control for many executions on a single linearization. For example, assessing nearest neighbor does not require any additional integrations or simulations. These improvements make long time horizons viable, allowing for meaningful distance computations and steering between distant states. The methods are demonstrated on a 1-link and 2-link pendulum on a cart.**

I. Introduction

Planning for dynamic systems is trajectory exploration through nonconvex state spaces. These non-convexities often include configuration space obstacles and regions of inevitable collision [10]. While sample-based planning algorithms differ in many ways they commonly conduct many short trajectory explorations to connect or nearly connect a large number of stochastically sampled states with the goal of connecting a start state to a final state. Earlier algorithms like RRT [10] guarantee with probability one that if a connecting trajectory exists, that the algorithm will find it. Newer algorithms like RRT*, PRM*, SST*, and FMT* [7, 8, 9, 11, 12] guarantee asymptotic convergence to the globally optimal connecting trajectory. For dynamic systems, planners are in need of computationally efficient methods to *compute distances*—e.g. for assessing nearest neighbors—and to *steer*—i.e. to extend to sampled states. Furthermore, each method must be able to handle numerical issues like sensitivities to initial conditions so that the steering trajectories can have a significant time horizon.

The methods proposed in this paper are complementary to [2, 13, 14]. For planning dynamic systems, distance computation and steering are solutions to optimal control problems. They are often approximated through linearizing the nonlinear dynamics and solving linear quadratic regulation (LQR) or tracking (LQT) problems [2, 13]. Both [2, 13] linearize around a single state and so the system is time-invariant. In comparison, we linearize around the zero-control or "free" trajectory

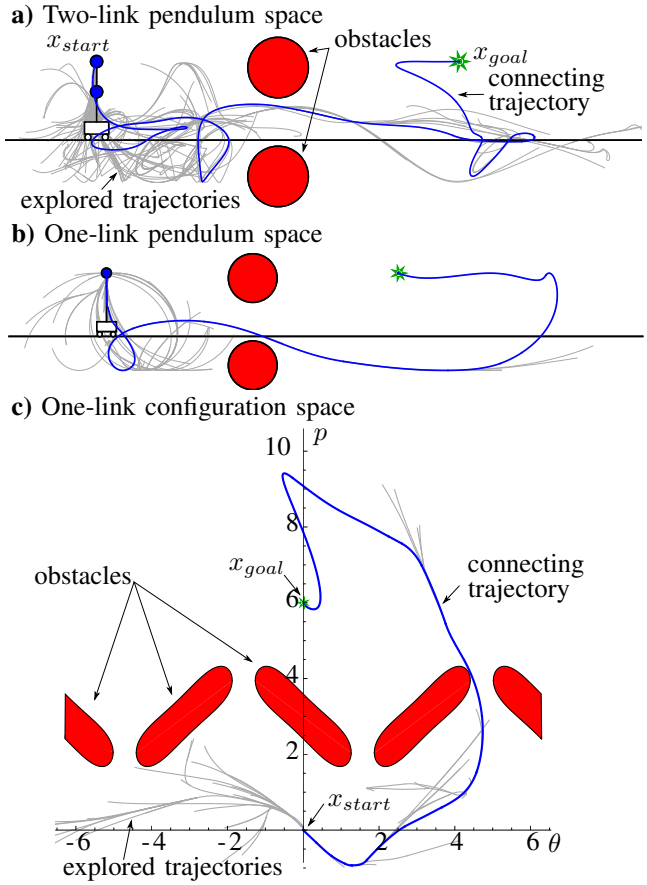

Fig. 1. Connecting and explored trajectories for 2-link **a** and 1-link **b** pendulum problems. **c)** Connecting and explored trajectories for 1-link pendulum problem in (p, θ) configuration space.

so that the linearization remains a good approximation for longer time horizons. Since our approximate system is time-varying we formulate the LQR and LQT so that through precomputation and caching they can be solved efficiently for many distance and steering executions from the same state.

Perez et al. [13] note that the distance computation is inaccurate for states far apart—i.e. states outside linearized region. For this reason, we propose a trust region-like approach to expand and contract a maximal time horizon online which provides a level of assurance that all LQR and LQT solutions remain in regions where the linearization dominates. Additionally, the trust region avoids needing a conservative maximal time horizon established for the full space.

Both [2, 14] set up the linear quadratic optimal control problem as an optimal state transfer without a running state

cost. As Hauser [4] indicates, the solution to such a problem is open-loop, which can be numerically intractable as the conditioning of the reachability Gramian becomes poor. Indeed, we show that the reachability Gramian's condition number approaches machine precision for the n-link pendulum on a cart in the examples, and as such, the methods in [2, 14] are infeasible. Instead, we propose implementing a feedback loop and performing the same state transfer procedure on the closed-loop system, which has a reachability Gramian with improved conditioning.

The state and trajectory solutions to the LQR problems are infeasible—i.e. trajectories for the linear system can not be trajectories of a nonlinear system. In order to steer, we implement the trajectory functional projection operator in [3, 5] to project approximate trajectories to feasible trajectories. We consider exact and inexact steering. Inexact steering finds a trajectory that propagates from an initial state toward a desired state while exact steering finds a trajectory that connects the two states. Exact steering requires solving a boundary value problem. We provide a gradient calculation for exact steering which in a steepest descent approach iterates through inexact steering problems with solution trajectories that converge to a trajectory that solves the exact steering problem. A similar approach is taken in [4].

The proposed distance computation and steering methods in the paper are contributions to general planners. The methods are designed so that long time horizon exploring is viable. This is reflected in the examples where a small number of vertices of an RRT are needed to plan a 1-link and a 2-link pendulum on a cart through a corridor of obstacles. The systems and connecting and explored trajectories are depicted in Figure 1. The pendulum on a cart is under-actuated and unstable. The 2-link pendulum on a cart is chaotic.

The paper is organized as follows: Section II reviews planning for dynamic systems and introduces nearest neighbor, inexact steering and exact steering. Section III provides results for exactly and inexactly steering affine linear systems. Section IV extends the linear approximation results to nonlinear systems through a projection operator and a trust region. Section V provides an RRT algorithm with the steering and distances methods proposed in the paper as well as applies it to plan a 1-link and 2-link pendulum on a cart through a corridor of obstacles.

II. Review of Planning for Dynamic Systems

We wish to provide tools for kinodynamic planning where the dynamic system may be nonlinear, under-actuated, and unstable. The kinodynamic planning problem is to find a state and control trajectory (x, u) that connects a start state x_{start} to a goal state x_{goal} that satisfies the differential constraint given by the system dynamics

$$\dot{x}(t) = f(x(t), u(t)). \tag{1}$$

Set the space of *feasible state and control trajectories* $\mathcal{S}_{x_{start}}[0, t_{final}]$ with final time $t_{final} > 0$ as the space of all trajectories (x, u) for which Eq. 1 is satisfied—i.e. where for each $t \in [0, t_{final}]$, $x(t) - x_{start} - \int_0^t f(x(\tau), u(\tau)) = 0$.

Additionally, the state and control must remain in the set of *allowable state and control* (X, U) throughout the full trajectory where the allowable state subset $X \subset \mathbb{R}^n$ and control subset $U \subset \mathbb{R}^m$ are compact. The set X is free of all obstacles and as such can be nonconvex. The non-allowable states may include configuration space obstacles and regions of inevitable collision [10]. The boundaries of the set U may be the saturation limits of the system actuators. It is worth noting that a feasible trajectory $(x, u) \in \mathcal{S}_{x_{start}}[0, t_{final}]$ may not be allowable.

Planning algorithms either find a path that connects a start state to an end state or find the optimal path. An optimal path minimizes a cost function

$$J(x, u, t_{final}) = \int_0^{t_{final}} \ell(x(\tau), u(\tau), \tau) d\tau + \ell_f(t_{final}) \tag{2}$$

where ℓ is the running cost and ℓ_f is the terminal cost. The optimal path is the one that minimizes J:

Sample-based planning algorithms find a feasible or optimal path by generating a graph $G = (V, \mathcal{E})$ where the vertices V are explored states $x \in X$ and the edges $\mathcal{E}$ are state and control trajectories that connect two states in V—i.e. $(x, u; t_h) \in \mathcal{E}$ implies for time horizon $t_h > 0$, $x(0) \in V$, $x(t_h) \in V$ and $(x, u) \in \mathcal{S}_{x(0)}[0, t_h]$.

Methods based on RRT build the graph through two functions, nearest neighbor and steering. We consider both exact and inexact steering. Exact steering connects an initial and desired state while inexact steering only propagates the system toward the desired state.

A. Nearest Neighbor

The nearest neighbor calculation relies on the choice of distance function $d : X \times X \to \mathbb{R}$ between the states $x_0 \in V$ and the sampled state x_{samp}. For dynamic systems Euclidean distance is a poor choice, even for linear time-invariant systems [2]. Instead, as [10] suggests, the ideal distance function is the cost of the optimal trajectory that transfers the state x_0 to or near x_1. Here, optimality is given by solving an optimal control problem for a specified cost function J where J may have the same form as Eq. 2.

$$d(x_0, x_1) = \min_{t_h > 0; (x,u) \in \mathcal{S}_{x_0}[0, t_h]} J. \tag{3}$$

The constraint $\mathcal{S}_{x_0}[0, t_h]$ to feasible trajectories makes this minimization an optimal control problem which can be computationally expensive especially in the context of planning. As such, [2, 13] approximate the distance through a linearization and the solution through LQR, which is also the approach taken in this paper. However, we additionally propose a trust region-like approach to ensure that the solution trajectory remains in a region where the linearization is a good approximation of the nonlinear dynamics.

B. Inexact Steering

For inexact steering, a type of shooting method is often employed to transfer the system from an initial state $x_0 \in X$ to a neighborhood of a desired state $x_1 \in X$. In [8] a constant control selected through bisection is applied for a short time horizon. In [11, 12] a number of piecewise constant controls and time horizons are randomly selected until one satisfies desired criteria.

In our approach, we linearize the dynamics around the zero-control trajectory, solve a minimal control energy or LQT problem, and project the linear solution to the trajectory manifold that satisfies the system's dynamics. The zero-control trajectory is the solution to:

$$\dot{x}_{zero}(t) = f(x_{zero}(t), 0), \text{ s.t. } x_{zero}(0) = x_0. \tag{4}$$

Through the Taylor expansion about $(x_{zero}, 0)$, the affine linear approximation $(\tilde{x}, \tilde{u})$ is

$$\begin{array}{c} \tilde{x}(t) = x_{zero}(t) + z(t), \text{ and } \tilde{u}(t) = v(t), \\ \text{where } \dot{z} = A(t)z(t) + B(t)v(t), \text{ s.t. } z(0) = 0 \end{array} \tag{5}$$

where $A(t) = \frac{\partial}{\partial x(t)} f(x(t), u(t))|_{(x(t),u(t))\to(x_{zero}(t),0)}$ and $B(t) = \frac{\partial}{\partial u(t)} f(x(t), u(t))|_{(x(t),u(t))\to(x_{zero}(t),0)}$ are time-varying. For the affine linear dynamics, we consider two time-varying linear quadratic optimization problems. The first is the minimal control energy problem while the second has an additional cost associated with the error of the final state with the desired state, which is a LQT problem. Finally, we use the trajectory functional projection operator proposed and analyzed in [5] to map the solutions to the space of feasible trajectories $\mathcal{S}_{x_0}[0, t_h]$.

C. Exact steering

Exact steering calls for solving a boundary value problem to compute a feasible trajectory $(x, u) \in \mathcal{S}_{x_0}[0, t_h]$ that transfers the system from a state $x_0 \in X$ to a state $x_1 \in X$ in $t_h > 0$ time. One technique is to solve the optimal state transfer problem [4]:

$$\begin{array}{c} \min\limits_{(x,u)\in\mathcal{S}_{x_0}[0,t_h]} \int_0^{t_h} \ell(x(\tau), u(\tau)) d\tau \\ \text{s.t. } x(0) = x_0 \text{ and } x(t_h) = x_1. \end{array}$$

Inspired by [4], we propose a method to compute an exact steering trajectory by iteratively solving inexact steering problems for different desired states. In other words, in order to exactly steer x_0 to x_1, the approach computes an $x_2 \in X$ for which the inexact steering trajectory from x_0 to x_2 is the exact steering trajectory from x_0 to x_1.

III. Affine LTV State Exploration

When a system is affine linear it is possible to find a subset of states in X that are reachable from any initial state for a bounded control effort. Later, we will extend the affine linear reachability results to nonlinear system path planning through a local linearization about the zero-control trajectory and a projection.

Suppose the system with state and control $(\tilde{x}, \tilde{u})$ evolves according to

$$\tilde{x}(t) := \tilde{x}_T(t) + z(t) \text{ and } \tilde{u}(t) := v(t)$$

where $\tilde{x}_T$ is some time-varying translation from the origin and z is the solution to the linear differential equation

$$\dot{z}(t) = A(t)z(t) + B(t)v(t), \text{ s.t. } z(0) = 0 \tag{6}$$

where $A(t) \in \mathbb{R}^{n\times n}$ and $B(t) \in \mathbb{R}^{n\times m}$ are piecewise continuous. Through the state-transition matrix (STM) $\Phi(t, \tau)$ of $A(t)$, the solution to the linear differential equation is

$$z(t) = \int_0^t \Phi(t, \tau)B(\tau)v(\tau)d\tau. \tag{7}$$

The STM is the solution to the matrix differential equation

$$\frac{\partial}{\partial\tau}\Phi(t, \tau) = -\Phi(t, \tau)A(\tau), \ \Phi(t, t) = Id_{n\times n} \tag{8}$$

where $Id_{n\times n}$ is the $n \times n$ identity matrix. We wish to find the states $\tilde{x}' \in X$ such that there exists a control v constrained to some set $\mathcal{V} \subset \mathcal{U}$ which transfers $\tilde{x}(0) = \tilde{x}_T(0)$ to $\tilde{x}'$ in t_h time. Define the set reachable at t_h with control constrained to $\mathcal{V}$ as

$$\begin{array}{l} \tilde{X}_{\mathcal{V}} := \{\tilde{x}' \in \mathbb{R}^n | \exists v \in \mathcal{V} \\ \quad \text{where } \tilde{x}' = \tilde{x}_T(t) + \int_0^{t_h} \Phi(t, \tau)B(\tau)v(\tau)d\tau\}. \end{array}$$

Consider the control constrained by the control energy

$$\|v\|_R := \frac{1}{2}\int_0^{t_h} v(\tau)^T R(\tau)v(\tau)\, d\tau \tag{9}$$

where $R(\cdot)$ is a piecewise continuous matrix with $R(\tau) = R(\tau)^T > 0$ symmetric positive-definite. The control set with bounded control energy is

$$\mathcal{V}_{\delta v} := \{v \in \mathcal{U} | \frac{1}{2}\int_0^{t_h} v(\tau)^T R(\tau)v(\tau)\, d\tau < \delta v\} \tag{10}$$

where $\delta v > 0$ and $t_h > 0$.

Next, we consider an open-loop and a closed-loop form for the control and analyze their respective bounded reachable sets.

A. Open-Loop Exact Linear Shooting

Suppose v has the form

$$v(t) = -R^{-1}(t)B(t)^T\Phi(t_h, t)^T\eta \tag{11}$$

where $\eta \in \mathbb{R}^n$. The significance of this control form is that it is the *minimal-energy control* with control energy Eq. 9 (see [6] Theorem 11.4). This control is open-loop because of its independence on z.

By setting η, the control Eq. 11 transfers $\tilde{x}(0)$ to some $\tilde{x}' \in \mathbb{R}^n$ through Eq. 6. This procedure is shooting. The δv bounded control reachable set $\tilde{X}_{\mathcal{V}_{\delta v}}$ are the points $\tilde{x}' \in \mathbb{R}^n$ for which there exists an η for which both $v \in \mathcal{V}_{\delta v}$ and v transfers $\tilde{x}(0)$ to $\tilde{x}'$. This set $\tilde{X}_{\mathcal{V}_{\delta v}}$ is given in the following Lemma:

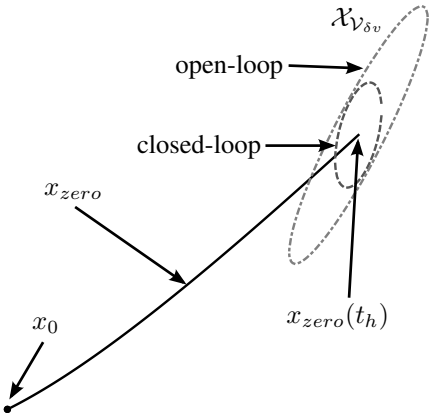

Fig. 2. Reachable set for open-loop and closed-loop double integrator for $x_0 = [0.5, 1]^T$, $\delta_v = 0.2$, $t_h = 1.0$.

Lemma 1: Supposing $A(\cdot)$ and $B(\cdot)$ are so that Eq. 6 is controllable on $[0, t_h]$ and that v has form Eq. 11, then

$$\begin{aligned}\tilde{X}_{\mathcal{V}_{\delta v}} = \{\tilde{x}' \in \mathbb{R}^n | \\ \tfrac{1}{2}(\tilde{x}' - \tilde{x}_T(t_h))^T W_0(t_h)^{-1}(\tilde{x}' - \tilde{x}_T(t_h)) < \delta v\}\end{aligned} \tag{12}$$

where the matrix $W_0(t)$, $t \in (0, t_h]$ is symmetric positive-definite and W_0 is the solution to:

$$\begin{aligned}\dot{W}_0(t) = A(t)W_0(t) + W_0(t)A(t)^T + B(t)R(t)^{-1}B(t)^T \\ \text{s.t. } W_0(0) = 0.\end{aligned}$$

Proof: The symmetric positive-definiteness of $W_0(\cdot)$ follows from $(A(\cdot), B(\cdot))$ controllable and $R(\cdot)$ symmetric positive-definite. Its existence is guaranteed since it is the solution to a linear differential equation. The integral form of $W_0(t)$ is:

$$W_0(t) = \int_0^t \Phi(t,\tau)B(\tau)R(\tau)^{-1}B(\tau)^T\Phi(t,\tau)^T d\tau.$$

An η is allowable if $v \in \mathcal{V}_{\delta v}$ (see Eq. 10). That is, η is allowable if

$$\begin{aligned}&\frac{1}{2}\int_0^{t_h} v^T(\tau)R(\tau)v(\tau)d\tau \\ &= \frac{1}{2}\eta^T \int_0^{t_h} \Phi(t_h,\tau)B(\tau)R^{-1}(\tau)B(t)^T\Phi(t_h,\tau)^T d\tau\eta \\ &= \tfrac{1}{2}\eta^T W_0(t_h)\eta < \delta v.\end{aligned} \tag{13}$$

Through the form of v, the solution to z for some $\eta \in \mathbb{R}^n$ is

$$\begin{aligned}z(t) &= -\int_0^t \Phi(t,\tau)B(\tau)R(\tau)^{-1}B(\tau)^T\Phi(t_h,\tau)^T d\tau\eta \\ &= -W_0(t)\Phi(t_h,t)^T\eta.\end{aligned}$$

Since $W_0(t_h)$ is positive-definite, it is invertible and the η which transfers $z(0) = 0$ to a desired z' is $\eta = -W_0(t_h)^{-1}z'$. Plugging η in Eq. 13 we arrive at the set of reachable z'. The reachable $\tilde{x}'$ are $\tilde{x}' = \tilde{x}'_T(t_h) + z'$ which is the set $\tilde{X}_{\mathcal{V}_{\delta v}}$. ■

The set of reachable states $\tilde{X}_{\mathcal{V}_{\delta v}}$ is an ellipsoid centered around $\tilde{x}_T(t_h)$, as seen in Figure 2 for the double integrator system. A planning algorithm can leverage Lemma 1 to choose a reachable state and compute the state and control trajectories $(\tilde{x}, \tilde{u})$ that transfer the linear system to that desired state. This exact steering procedure for affine linear systems is as follows:

Algorithm 1 (Open-Loop Exact Linear Steering):
For $\tilde{x}' \in \tilde{X}_{\mathcal{V}_{\delta v}}$:
1) $\eta \leftarrow -W_0(t_h)^{-1}(\tilde{x}' - \tilde{x}_T(tf))$
2) $\tilde{x}(t) \leftarrow \tilde{x}_T(t) - W_0(t)\Phi(t_h,t)^T\eta$, and
3) $\tilde{u}(t) \leftarrow -R(t)^{-1}B(t)^T\Phi(t_h,t)^T\eta$

The procedure relies on a well conditioned $W_0(\cdot)$, especially at t_h. The time-varying matrix W_0 is the reachability Gramian weighted by R^{-1}. It is guaranteed to be invertible when the system $(A(\cdot), B(\cdot))$ is controllable, but numerical inversion requires a well conditioned matrix, which is not the case for the n-link inverted pendulum on a cart analyzed in the examples section. The condition number at time $t_h = 1.0$, $\kappa(W_0(1.0))$, for 1, 2, and 3-link pendulum on the cart is:

$$\begin{aligned}1-link: &\quad \kappa(W_0(1.0)) = 1.32 \times 10^6 \\ 2-link: &\quad \kappa(W_0(1.0)) = 1.08 \times 10^8 \\ 3-link: &\quad \kappa(W_0(1.0)) = 3.33 \times 10^{15}.\end{aligned} \tag{14}$$

For the 3-link pendulum, the condition number approaches machine precision on many computing devices. When the conditioning is sufficiently poor, the numerical error between $\tilde{x}'$ and the computed $\tilde{x}(t)$ through Algorithm 1 invalidates the procedure. For unstable systems, the conditioning can be improved through a closed-loop control form with stabilizing feedback. The goal is to design a feedback so that Algorithm 1 becomes a sort of stable shooting algorithm.

B. Closed-Loop Exact Linear Steering

In order to make Algorithm 1 numerically tractable, we consider a closed-loop system with a better conditioned reachability Gramian than the open-loop system. The closed-loop linear system is (A_K, B) where $A_K(t) := A(t) - B(t)K(t)$ with time-varying feedback gain $K(t) \in \mathbb{R}^{m\times n}$. For now, we assume K is given. It can be computed as the optimal feedback gain of a LQR problem. With a properly designed K, the closed-loop system (A_K, B) should have better numerical properties—e.g. better conditioning of relevant matrices—than the open-loop system.

Let $\Phi_K(\cdot,\cdot)$ be the state-transition matrix corresponding to A_K. The closed-loop control form is

$$v(t) = -K(t)z(t) - R^{-1}(t)B(t)^T\Phi_K(t_h,t)^T\eta. \tag{15}$$

The closed-loop linear system dynamics are

$$\begin{aligned}\dot{z}(t) = A_K(t)z(t) - B(t)R(t)^{-1}B(t)^T\Phi_K(t_h,t)^T\eta, \\ \text{s.t. } z(0) = 0\end{aligned}$$

with solution

$$\begin{aligned}z(t) &= -\int_0^t \Phi_K(t,\tau)B(\tau)R(\tau)^{-1}B(\tau)^T\Phi_K(t_h,\tau)^T d\tau\eta \\ &= -W_K(t)\Phi_K(t_h,t)^T\eta\end{aligned}$$

where W_K is the reachability Gramian corresponding to A_K, weighted by R^{-1}, and has differential form:

$$\begin{aligned}\dot{W}_K(t) = A_K(t)W_K(t) + W_K(t)A_K(t)^T \\ +B(t)R(t)^{-1}B(t)^T \text{ s.t. } W_K(0) = 0.\end{aligned} \tag{16}$$

As with Lemma 1, we wish to find the reachable subset of $\mathbb{R}^n$ in t_h time with bounded control energy, except for the closed-loop form:

Lemma 2: Supposing $A(\cdot)$ and $B(\cdot)$ are so that Eq. 6 is controllable on $[0, t_h]$ and that v has form Eq. 15, then

$$\tilde{X}_{\mathcal{V}_{\delta v}} = \{\tilde{x}' \in \mathbb{R}^n | (\tilde{x}' - \tilde{x}_T(t_h))^T S_K(t_h)^{-1} (\tilde{x}' - \tilde{x}_T(t_h)) < \delta v\} \tag{17}$$

where the matrix $S_K(t)$, $t \in (0, t_h]$ is symmetric positive-definite and is the solution to: (omitting $(\cdot)$)

$$\begin{aligned} &\dot{S}_K = AS_K + S_K A^T \\ &+(KW_K - R^{-1}B^T)^T R(KW_K - R^{-1}B^T) \text{ s.t. } S_K(0) = 0. \end{aligned} \tag{18}$$

The proof of Lemma 2 is similar to that of Lemma 1 and for the sake of brevity is omitted.

Similar to Lemma 1, Lemma 2 finds that the reachable states with the closed-loop control form an ellipsoid centered at $\tilde{x}_T(t_h)$, except where the ellipsoid's axes are parameterized by $S_K(t_h)^{-1}$ as opposed to $W_0(t_h)^{-1}$.

The analog to Algorithm 1 is

Algorithm 2 (Closed-Loop Exact Linear Steering):
For $\tilde{x}' \in \tilde{X}_{\mathcal{V}_{\delta v}}$ and feedback gain K:
1) $\eta \leftarrow -W_K(t_h)^{-1}(\tilde{x}' - \tilde{x}_T(tf))$
2) $\tilde{x}(t) \leftarrow \tilde{x}_T(t) - W_K(t)\Phi_K(t_h, t)^T \eta$, and
3) $\tilde{u}(t) \leftarrow -K(t)z(t) - R(t)^{-1}B(t)^T \Phi(t_h, t)^T \eta$

The conditioning of W_K and S_K for the n-link cart pendulum is a significant improvement over W_0 (see Eq. 14):

$$\begin{aligned} 1-link: &\quad \kappa(W_K(1.0)) = 5.84 &&\quad \kappa(S_K(1.0)) = 3.80 \times 10^4 \\ 2-link: &\quad \kappa(W_K(1.0)) = 5.49 \times 10^2 &&\quad \kappa(S_K(1.0)) = 2.52 \times 10^6 \\ 3-link: &\quad \kappa(W_K(1.0)) = 1.03 \times 10^5 &&\quad \kappa(S_K(1.0)) = 2.57 \times 10^6. \end{aligned}$$

The closed loop form should be used when the open loop reachability Gramian is poorly conditioned.

C. Inexact Linear Steering

The reachability results provide the minimal energy control to a reachable set. When the desired state is not within the reachable set a new objective can be considered which weights the importance of tracking the desired state. Consider the cost function

$$\begin{aligned} J(\tilde{x}, \tilde{u}; t_h) := &\frac{1}{2}\int_0^{t_h} \tilde{u}^T(\tau) R(\tau) \tilde{u}(\tau)\, d\tau \\ &+\tfrac{1}{2}(\tilde{x}(t_h) - \tilde{x}_{des})^T P_1 (\tilde{x}(t_h) - \tilde{x}_{des}) \end{aligned} \tag{19}$$

where $P_1 = P_1 \geq 0$ is symmetric positive semi-definite. Parameterized by (z, v), the cost is

$$\begin{aligned} J(z, v; t_h) = &\frac{1}{2}\int_0^{t_h} v^T(\tau) R(\tau) v(\tau)\, d\tau \\ &+\tfrac{1}{2}(\tilde{x}_T(t_h) + z(t_h) - \tilde{x}_{des})^T P_1 (\tilde{x}_T(t_h) + z(t_h) - \tilde{x}_{des})\Big]. \end{aligned} \tag{20}$$

The problem is to find a state and control trajectory that satisfies the linear dynamics Eq. 6. That is, we wish to solve the problem

Problem 1: Solve

$$\begin{aligned} &\min_{z, v; t_h} J(z, v; t_h) \\ &\text{s.t. } \dot{z}(t) = A(t)z(t) + B(t)v(t),\ z(0) = 0. \end{aligned}$$

For a fixed t_h, this problem is a linear quadratic tracking LQT problem [1], the solution of which is well known. The optimal control is

$$\begin{aligned} v^\star(t) = &-K^\star(t) z^\star(t) \\ &-R(t)^{-1}B(t)^T \Phi_K(t, t_h) P_1 (\tilde{x}_T(t_h) - \tilde{x}_{des}) \end{aligned} \tag{21}$$

with optimal feedback gain $K^\star(t)$

$$K^\star(t) = R(t)^{-1}B(t)^T P(t) \tag{22}$$

where $P(t)$ is the solution to the Riccati equation

$$\begin{aligned} -\dot{P}(t) = &A(t)^T P(t) + P(t)A(t) \\ &-P(t)B(t)R(t)^{-1}B(t)^T P(t) \text{ s.t. } P(t_h) = P_1. \end{aligned} \tag{23}$$

Notice that $v^\star$ has the closed-loop form Eq. 15 except for a specific feedback gain $K(t) = K^\star(t)$ and where $\eta = P_1(\tilde{x}_T(t_h) - \tilde{x}_{des})$. The gain K in Eq. 15 can be chosen as the optimal feedback gain from this LQT problem. The $K(t)$ would depend on the choice of $\tilde{x}_{des}$ as well as P_1. If $P_1 = 0$—i.e. the cost function reduces to just the control energy—then $K^\star \equiv 0$—i.e. there is no feedback—and the optimal control is the open-loop control Eq. 11. As such, the procedure Algorithm 1 does indeed compute the minimal control energy trajectory that transfers the system to a reachable state.

The following algorithm computes the optimal cost $\tilde{J}^\star(t_h) := J(\tilde{x}^\star, \tilde{u}^\star, t_h)$, and trajectory $(\tilde{x}^\star, \tilde{u}^\star)$ for fixed t_h:

Algorithm 3 (Fixed t_h Inexact Linear Steering):
For $\tilde{x}_{des} \in X$ and $t_h > 0$:
1) $P(t) \leftarrow$ solve Riccati equation Eq. 23
2) $K^\star(t) \leftarrow R(t)^{-1}B(t)^T P(t)$
3) $z^\star(t) \leftarrow$ solve $\dot{z}^\star(t) = A_{K^\star}(t) - B(t)R(t)^{-1}B(t)^T \Phi_K(t, t_h) P_1(\tilde{x}_T(t_h) - \tilde{x}_{des})$
4) $\tilde{x}^\star(t) \leftarrow \tilde{x}_T(t) + z^\star(t)$, and
5) $\tilde{u}^\star(t) \leftarrow -K^\star(t)z^\star(t) - R(t)^{-1}B(t)^T \Phi_K(t, t_h) P_1(\tilde{x}_T(t_h) - \tilde{x}_{des})$
6) $\tilde{J}^\star(t_h) \leftarrow$ solve cost function Eq. 20
7) Return $\tilde{J}^\star(t_h)$ and $(\tilde{x}^\star, \tilde{u}^\star)$

In comparison to Algorithms 1 and 2, this algorithm computes a trajectory that tracks any state in X as opposed to just the states in a reachable set. However, the trajectory does not transfer the state to $\tilde{x}_{des}$ and as such is an inexact steering.

In order to solve for the optimal time horizon $t_h^\star$, which is called for in Problem 1, a numerical minimization like bisection can repeatedly execute Algorithm 3 to find the $t_h^\star$ with least $\tilde{J}^\star(t_h)$:

$$t_h^\star = \arg\min_{t_h} \tilde{J}^\star(t_h). \tag{24}$$

Every execution of Algorithm 3 solves the Riccati equation, the linear state equation, and integrates the running cost, which is computationally expensive. Next, we propose an efficient inexact linear steering algorithm that relies on precomputation and caching so that the algorithm does not rely on solving any differential equations or integrations. The precomputations are also invariant on $\tilde{x}_{des}$.

D. Efficient Inexact Linear Steering

For an affine linear system—i.e. for a set x_T, A and B—Algorithm 3 must be executed anew for distinct t_h and $\tilde{x}_{des}$. As we show here, inexact linear steering gains efficiency through precomputation and caching to remove redundancies from multiple algorithm execution.

As previously noted, the closed-loop control Eq. 15 has the same form as the solution to the LQT problem with optimal control Eq. 21. The efficiency is gained through precomputing and fixing a feedback gain $K(t)$ and solving for η as was done in Section III-B. In other words, the problem is minimize the cost Eq. 20 constrained to the closed-loop control Eq. 15:

Problem 2: With fixed $K(t)$, solve

$$\begin{aligned} & \min_{\eta; t_h} J(z, v, t_h) \\ \text{s.t.} \quad & \dot{z}(t) = A_K(t)z(t) \\ & \quad -B(t)v(t)R^{-1}(t)B(t)^T\Phi_K(t_h,t)^T\eta,\ z(0) = 0 \\ & v(t) = -K(t)z(t) - R^{-1}(t)B(t)^T\Phi_K(t_h,t)^T\eta. \end{aligned}$$

The solution to Problem 2 is not the same as the solution to Problem 1 unless the feedback gain $K(t)$ happens to be the optimal $K^\star(t)$.

The solution to Problem 2 is given in the following Lemma:

Lemma 3: For fixed $t_h > 0$ and K and supposing $A(\cdot)$ and $B(\cdot)$ are so that Eq. 6 is controllable on $[0, t_h]$, the solution to Problem 2 is

$$\eta^\star = P_{t_h}(\tilde{x}_T(t_h) - \tilde{x}_{des}) \tag{25}$$

where

$$P_{t_h} = (W_K(t_h)P_1W_K(t_h) + S_K(t_h))^{-1}W_K(t_h)P_1.$$

Additionally,

$$\begin{aligned} z^\star(t) &= -W_K(t)\Phi_K(t_h,t)^T\eta^\star, \\ v^\star(t) &= [K(t)W_K(t) - R(t)^{-1}B(t)^T]\Phi_K(t_h,t)^T\eta^\star \end{aligned} \tag{26}$$

and

$$\begin{aligned} &J(z^\star, v^\star; t_h) = \tfrac{1}{2}(\eta^\star)^T S_K(t_h)\eta^\star \\ &+\tfrac{1}{2}(\tilde{x}_T(t_h) - W_K(t_h)\eta^\star - \tilde{x}_{des})^T P_1(\tilde{x}_T(t_h) - W_K(t_h)\eta^\star - \tilde{x}_{des}). \end{aligned} \tag{27}$$

Proof: The integral form of z is Eq. 7:

$$\begin{aligned} z(t) &= \textstyle\int_0^t \Phi_K(t,\tau)B(\tau)v(\tau)d\tau \\ &= -\textstyle\int_0^t \Phi_K(t,\tau)B(\tau)R(\tau)^{-1}B(\tau)^T\Phi(t_h,\tau)^T d\tau\eta \\ &= -W_K(t)\Phi_K(t_h,t)^T\eta. \end{aligned}$$

Plugging $z(t)$ into $v(t)$,

$$v(t) = (K(t)W_K(t) - R(t)^{-1}B(t)^T)\Phi_K(t_h,t)^T\eta.$$

As such, $z(t)$ and $v(t)$ depend linearly on η. The control energy is

$$\begin{aligned} &\tfrac{1}{2}\textstyle\int_0^{t_h} v^T(\tau)R(\tau)v(\tau)\,d\tau \\ &= \tfrac{1}{2}\eta^T \textstyle\int_0^{t_h} \Phi_K(t_h,\tau)(K(\tau)W_K(\tau) - R(\tau)^{-1}B(\tau)^T)^T \\ &\quad \cdot R(\tau)(K(\tau)W_K(\tau) - R(\tau)^{-1}B(\tau)^T)\Phi_K(t_h,\tau)d\tau\eta \\ &= \tfrac{1}{2}\eta^T S_K(t_h)\eta. \end{aligned}$$

The cost is thus

$$\begin{aligned} J = &\tfrac{1}{2}(\eta)^T S_K(t_h)\eta + \tfrac{1}{2}(\tilde{x}_T(t_h) - W_K(t_h)\eta - \tilde{x}_{des})^T \\ &\cdot P_1(\tilde{x}_T(t_h) - W_K(t_h)\eta - \tilde{x}_{des}) \end{aligned}$$

with optimal $\eta^\star$ where $\frac{\partial}{\partial\eta}J|_{\eta\to\eta^\star} = 0$:

$$\begin{aligned} \tfrac{\partial}{\partial\eta}J|_{\eta\to\eta^\star} = &[(S_K(t_h) + W_K(t_h)P_1W_K(t_h))\eta \\ &-W_K(t_h)P_1(\tilde{x}_T(t_h) - \tilde{x}_{des})]_{\eta\to\eta^\star} = 0. \end{aligned}$$

Since $S_K(t_h)$ and $W_K(t_h)$ are positive definite through the controllability assumption, $(S_K(t_h) + W_K(t_h)P_1W_K(t_h))$ is invertible. Solving for $\eta^\star$ results in Eq. 25. ∎

The lemma instructs how to do efficient fixed t_h inexact steering:

Algorithm 4 (Efficient Fixed t_h Inexact Linear Steering):
For $\tilde{x}_{des} \in X$ and $t_h > 0$:

1) $P_{t_h} \leftarrow (W_K(t_h)P_1W_K(t_h) + S_K(t_h))^{-1}W_K(t_h)P_1$
2) $\eta^\star \leftarrow P_{t_h}(\tilde{x}_T(t_h) - \tilde{x}_{des})$
3) $\tilde{x}^\star(t) \leftarrow \tilde{x}_T(t) - W_K(t)\Phi_K(t_h,t)^T\eta^\star$
4) $\tilde{u}^\star(t) \leftarrow [K(t)W_K(t) - R(t)^{-1}B(t)^T]\Phi_K(t_h,t)^T\eta^\star$
5) $\tilde{J}^\star(t_h) \leftarrow$ solve Eq. 27
6) Return $\tilde{J}^\star(t_h)$ and $(\tilde{x}^\star, \tilde{u}^\star)$

The algorithm is efficient compared to Algorithm 4 because it does not rely on solving any differential equations assuming certain functions have been precomputed. The functions to be precomputed and saved in memory are $K(t)$, $W_K(t)$, $S_K(t)$ and $\Phi_K(t_h^{max}, t)$ for a specified long time horizon $t \in [0, t_h^{max}]$, $t_h^{max} > 0$. As such, Algorithm 4 relies solely on matrix manipulations to return a fixed t_h inexact linear steering trajectory assuming $t_h < t_h^{max}$. Note that $\Phi_K(t_h, t) = \Phi_K(t_h^{max}, t_h)^{-1}\Phi_K(t_h^{max}, t)$. Also, $\Phi(t_h, t_h) = Id_{n\times n}$ and so $\tilde{x}^\star(t_h)$, $\tilde{u}^\star(t_h)$ and $\tilde{J}^\star(t_h)$ do not rely on precomputing Φ.

As with Algorithm 3, Algorithm 4 can be executed many times in a minimization procedure to solve for the optimal time horizon $t_h^\star$ through Eq. 24.

As of yet, the feedback gain is assumed to have been chosen in some way. A reasonable choice is the optimal feedback gain to the LQT problem, Problem 1, for the max time horizon $t_h = t_h^{max}$. It is the case that the solution to Problem 2 is equivalent to the solution to Problem 1 when $P_1 = P_{t_h}$. In other words, if $P_1 = P_{t_h}$, then $K = K^\star$, which will be the case at least at t_h^{max} for this choice of K.

IV. Extending to Nonlinear System

We extend the affine linear steering results in Section III to nonlinear dynamics $\dot{x}(t) = f(x(t), u(t))$ by projecting inexact and exact linear steering trajectories to feasible trajectories. For an initial state x_0 that may be a vertex of a graph generated by a planner, the dynamics are linearized about the zero-control trajectory for $t_h > 0$ time. The zero-control trajectory x_{zero} is given by Eq. 4 and the linearization is in Eq. 5.

A planner can pick which linear steering algorithm, Algorithms 1-4, to approximately transfer the system to a desired state x_{des}. The resulting approximate trajectories $(\tilde{x}, \tilde{u})$ are not feasible—i.e. $(\tilde{x}, \tilde{u})$ do not satisfy the system dynamics—unless the dynamics are linear.

When the approximate trajectories are within a region where the linearization is reasonable, then there are nearby feasible trajectories. The trajectory functional projection operator $\mathcal{P}$ proposed in [5] maps $(\tilde{x}, \tilde{u})$ to a feasible trajectory $(x, u) \in \mathcal{S}_{x_0}[0, t_h]$:

$$\begin{bmatrix} x \\ u \end{bmatrix} = \mathcal{P}\left(\begin{bmatrix} \tilde{x} \\ \tilde{u} \end{bmatrix}\right) := \begin{cases} \dot{x} = f(x, u) \\ u = \tilde{u} - K(x - \tilde{x}). \end{cases} \tag{28}$$

The projection is a feedback loop with gain K, reference signal $\tilde{x}$, and feedforward term $\tilde{u}$. The feedback gain may be chosen as the optimal feedback of an LQR or LQT problem.

A. Trust Region

Depending on the dynamics and the time horizon t_h, the projected trajectories (x, u) may not be near the approximate trajectories $(\tilde{x}, \tilde{u})$. In other words, it is possible that for

$$H := \frac{J(x, u, t_h)}{J(\tilde{x}, \tilde{u}, t_h)}$$

that $H >> 1$ where J is a cost on the state, control and final time with form Eq. 2. When $H >> 1$, the projected and approximate trajectories are far from each other and the projection is *untrustworthy*, in which case the results should be discarded.

This trust region-like approach assesses when the approximation is *trustworthy*. Each vertex $x_0 \in V$ in a graph generated by a planner has its own trust region. As such, the trust regions are state dependent. The region is restricted by a maximum allowable time horizon $\overline{t}_h$. When trustworthy, the trust region may be expanded by increasing $\overline{t}_h$. When untrustworthy, the trust region is contracted by decreasing $\overline{t}_h$. Since the linearization is around the zero-control trajectory, there is a sufficiently short time horizon for which any nonlinearities are inconsequential. The trust region approach avoids choosing a single conservative maximal horizon time for all X. Instead, the maximal time horizon is dictated by the local dynamics.

In order to contract or expand the trust region and to determine whether the projection is trustworthy or not, choose variables $0 < \underline{\alpha} < \overline{\alpha}$ and $\beta > 1$. Expand or contract according to the following chart:

$$\begin{array}{rcll} H < \underline{\alpha} & \Longrightarrow & \text{trustworthy (expand):} & \overline{t}_h \to \overline{t}_h \cdot \beta \\ \underline{\alpha} < H < \overline{\alpha} & \Longrightarrow & \text{trustworthy (no expand):} & \overline{t}_h \to \overline{t}_h \\ \overline{\alpha} < H & \Longrightarrow & \text{untrustworthy (contract):} & \overline{t}_h \to \overline{t}_h / \beta. \end{array}$$

The trust region approach allows a planner to make decisions based on approximate trajectories without projecting them to the feasible space. One of these decisions is computing the nearest neighbor.

B. Inexact Steering

Inexact steering transfers a vertex $x_0 \in V$ to a state near a desired state x_{des}. Suppose $K(t)$, $W_K(t)$, $S_K(t)$ and $\Phi_K(t_h^{max}, t)$ have been computed for the vertex x_0. Then, the efficient inexact steering trajectory is computed by projecting the approximate trajectories given by efficient inexact linear steering, Algorithm 4.

Algorithm 5 (Efficient Inexact Steering):
For $x_0 \in X$ and $x_{des} \in X$:
1) $t_h^\star \leftarrow \arg\min_{t_h} \tilde{J}^\star(t_h)$
2) $\tilde{J}^\star(t_h) \leftarrow$ Algorithm 4 for t_h
3) $(\tilde{x}^\star, \tilde{u}^\star, \tilde{J}^\star(t_h^\star)) \leftarrow$ Algorithm 4 for $t_h^\star$
4) $(x, u) \leftarrow \mathcal{P}(\tilde{x}^\star, \tilde{u}^\star)$
5) return $(x, u, \tilde{x}, \tilde{u}, \tilde{J}^\star(t_h^\star), t_h^\star)$

C. Nearest Neighbor

The nearest neighbor computation chooses the vertex $x_0 \in V$ of the graph $G = (V, \mathcal{E})$ nearest a sampled state $x_{samp} \in X$ where nearest is specified by a distance function $d : X \times X \to \mathbb{R}$. The distance between two states x_0 and x_{samp}, $d(x_0, x_{samp})$, is the optimal cost to transfer the system from x_{samp} to or nearby x_{samp}, Eq. 3. Choose the cost J as the cost for inexact linear steering, Eq. 19, where the desired state is x_{samp}:

$$\begin{aligned} J(x, u; t_h) := \frac{1}{2}\int_0^{t_h} u^T(\tau) R(\tau) u(\tau)\, d\tau \\ + \tfrac{1}{2}(x(t_h) - x_{samp})^T P_1 (x(t_h) - x_{samp}). \end{aligned} \tag{29}$$

Ideally, the distance is $d(x_0, x_{samp}) = J(x^\star, u, t_h^\star)$ for the optimal feasible trajectory $(x, u) \in \mathcal{S}_{x_0}[0, t_h^\star]$. Since such a non-linear optimal control problem is slow to compute, we instead set $d(x_0, x_{samp}) = J(\tilde{x}^\star, \tilde{u}^\star, t_h^\star)$, which is the approximate distance given by inexact linear steering (see Section III-D and Algorithm 4)

The trust region paradigm provides an expectation that $J(\tilde{x}^\star, \tilde{u}^\star, t_h^\star) = J(x^\star, u, t_h^\star)$—i.e. $J(\tilde{x}^\star, \tilde{u}^\star, t_h^\star) < J(x^\star, u, t_h^\star)/\overline{\alpha}$. Since a planning algorithm steers from the nearest vertex, if its distance is untrustworthy, then the trust region for the nearest vertex is contracted and the results of the nearest neighbor are discarded.

D. Exact Steering

Exact steering finds a feasible trajectory (x, u) that connects an initial state x_0 to a desired state x_{des}. Let

$$E = \frac{1}{2}(x(t_h) - x_{des})^T P_1 (x(t_h) - x_{des})$$

be the error between the desired state and the state transferred to at time t_h. When an exact steering trajectory is found, $E = 0$. Inexact steering with desired state x_{des} will generate a feasible trajectory (x, u) though Algorithm 5 for which $E > 0$. However, for a different desired state, inexact steering may find a trajectory that solves the exact steering problem. Label the desired state for the inexact steering as x_{pull}. The problem is to find the state $x_{pull} \in X$ that "pulls" the inexact steering trajectory to x_{des}. In other words, inexact steering performs a mapping $g(x_{pull}, t_h) \to (x, u)$ where we wish to find a pair $x_{pull} \in \mathbb{R}^n$ and $t_h > 0$ for which $x(t_h) = x_{des}$.

The mapping g is $(x, u) = \mathcal{P}(\tilde{x}, \tilde{u})$ where the approximate trajectories $(\tilde{x}, \tilde{u})$ are given in Lemma 3, which assumes precomputations of x_{zero}, K, W_K, S_K and Φ_K.

The approach is to minimize E over x_{pull} and t_h constrained to g. For a fixed t_h, E can be minimized through

a numerical minimization routine like steepest descent with gradient $\frac{\partial}{\partial x_{pull}}E$. The gradient is given in the following lemma:

Lemma 4: For $t_h > 0$, the gradient of E with respect to $x_{pull} \in X$ where $(x, u) = g(x_{pull}, t_h)$ is

$$\frac{\partial}{\partial x_{pull}}E = (x(t_h) - x_{des})^T P_1 \gamma(t_h) \tag{30}$$

where $\gamma = \frac{\partial}{\partial x_{pull}}x$ is the solution to

$$\dot{\gamma}(t) = A_K(t)\gamma(t) + B(t)[\frac{\partial}{\partial x_{pull}}\tilde{u} - K(t)\frac{\partial}{\partial x_{pull}}\tilde{x}] \tag{31}$$

where

$$\frac{\partial}{\partial x_{pull}}\tilde{x}(t) = W_K(t)\Phi_K(t_h, t)^T P_{t_h}$$

and

$$\frac{\partial}{\partial x_{pull}}\tilde{u}(t) = -[K(t)W_K(t) - R(t)^{-1}B(t)]\Phi_K(t_h, t)^T P_{t_h}.$$

Proof: The partial derivative of E with respect to x_{pull} is Eq. 30. The partial derivative of the projection $(x, u) = \mathcal{P}(\tilde{x}, \tilde{u})$ Eq. 28 is

$$\frac{\partial \mathcal{P}}{\partial x_{pull}} = \begin{cases} \dot{\gamma}(t) = A(t)\gamma(t) + B(t)\mu(t),\ \gamma(0) = 0 \\ \mu(t) = \frac{\partial}{\partial x_{pull}}\tilde{u}(t) - K(t)[\gamma(t) - \frac{\partial}{\partial x_{pull}}\tilde{x}(t)] \end{cases}$$

where $\mu = \frac{\partial}{\partial x_{pull}}u$. Plugging $\mu(t)$ into $\dot{\gamma}(t)$ results in Eq. 31. The partials $\frac{\partial}{\partial x_{pull}}\tilde{x}(t)$ and $\frac{\partial}{\partial x_{pull}}\tilde{u}(t)$ are the partials of Eq. 26 with $\tilde{x} = x_{zero} + z$ and $\tilde{u} = v$ and $\frac{\partial}{\partial x_{pull}}\eta = -P_{t_h}$. ■

For a choice of $t_h > 0$, the gradient result in Lemma 4 can be used in a numerical minimization procedure to iteratively step to a x_{pull} for which $E = 0$.

V. Examples and Algorithm

Using the steering and nearest neighbor methods proposed in the paper we conduct an RRT to plan a trajectory that transfers an n-link pendulum on a cart through a corridor of obstacles to a goal state. The state is composed of the pendulum angles θ_i, their angular velocities $\dot{\theta}_i$, the cart position p and the cart velocity $\dot{p}$. The control is the force applied to the cart, accelerating it forward or backward. The pendulums are in inverted equilibrium for both the start and goal states, with cart position at $p_{start} = 0$m and $p_{goal} = 6$m, as seen in Figure 1a and b. The system must find a path to connect the start and goal without the pendulum heads colliding with the obstacles. The obstacles are circles of radius 0.6 at $(3, -0.85)$ and $(3, 0.85)$.

We consider both 2-link (see Figure 1a) and 1-link (see Figure 1b) systems. The 2-link is composed of $n = 6$ states while the 1-link is composed of $n = 4$ states. Each pendulum head has mass 0.1 kg and the pendulum lengths are assumed massless. The 1-link pendulum length is 1 m long while both the lengths for the 2-link pendulum are 0.5 m long. The cart has mass 1 kg.

The path is found by executing RRT [10] with inexact steering (see Section IV-B) and a trust region (see Section IV-A). The choices for the algorithm parameters are $t_h^{max} = 1.0$s, $\overline{\alpha} = 3$, $\underline{\alpha} = 1.1$, and $\beta = 1.25$. The matrices R and P_1 in the cost J, Eqs. 19 and 29, are $R = [0.025]$ and $P_1 = Id_n$ the $n \times n$ identity matrix.

RRT formulated with efficient inexact steering through precomputing and with a trust region follows. The methods in this paper can also be implemented for more complex planners like RRT*.

Algorithm 6 (RRT with Efficient Inexact Steering and Trust Region):
for $x_{start} \in X$:

1) Precompute $x_{zero}, K, W_K, S_K, \Phi_K$ for x_{start} over $t \in [0, t_h^{max}]$. (Eqs. 4, 22, 16, 18, 8)
2) $N_{init} = \{x_{start}, \bar{t}_h, x_{zero}, K, W_K, S_K, \Phi_K\}$
3) $V \leftarrow \{N_{init}\}$; $\mathcal{E} \leftarrow \emptyset$
4) while x_{goal} not found:
5) $x_{samp} \leftarrow$ sample(x)
6) $N_{near} \leftarrow$ nearestneighbor(V, x_{samp}). (Sec IV-C)
7) $(x_{new}, u_{new}, \tilde{x}_{new}, \tilde{u}_{new}, \tilde{J}^\star, t_h^\star) \leftarrow$ Alg. 5 from N_{near}
8) $(x, u) \leftarrow \mathcal{P}(\tilde{x}, \tilde{u})$ (Eq.28)
9) $H \leftarrow J(x, u, t_h^\star)/\tilde{J}^\star$
10) if $H < \underline{\alpha}$: $N_{near}.\bar{t}_h \leftarrow \min(N_{near}.\bar{t}_h \cdot \beta, t_h^{max})$
11) if $\overline{\alpha} < H$: $N_{near}.\bar{t}_h \leftarrow N_{near}.\bar{t}_h/\beta$; Go to step 4
12) if $(x_{new}(t), u_{new}(t)) \in (X, U)$ for all $t \in [0, t_h^\star]$:
13) Precompute $x_{zero}, K, W_K, S_K, \Phi_K$ for $x_{new}(t_h^\star)$ over $t \in [0, t_h^{max}]$.
14) $N_{new} \leftarrow \{x_{new}(t_h^\star), N_{near}.\bar{t}_h, x_{zero}, K, W_K, S_K, \Phi_K\}$
15) $V \leftarrow V \cup \{N_{new}\}$; $\mathcal{E} \leftarrow \mathcal{E} \cup \{(x_{new}, u_{new}; t_h^\star)\}$
16) return $G = (V, \mathcal{E})$

The results of applying Algorithm 6 are in Figure 1, which shows the explored trajectories for the top pendulum in gray and the connecting trajectory in blue. The one-link pendulum on a cart found x_{goal} by the 115^{th} node while the two-link pendulum on a cart found it by the 335^{th} node. Both found allowable trajectories of max time horizon $t_h^{max} = 1.0$s.

VI. Discussion / Conclusion

There are two paradigms for sample-based planning of dynamic systems. The first, currently dominating, is to search with a shorter time horizon and simple steering and distance computations which is made up for through many samples and searches and generating a graph with many nodes—e.g. on the order of 10000 [8, 11, 12]. The second is to search with longer time horizons and to make informed steering and distance computations as we introduce in this paper. While this approach has the potential to solve hard planning problems using much fewer number of nodes—e.g, on the order of 100 as in this paper—linearizing around the free trajectory instead of a single state is more expensive. Yet, we believe that appropriate precomputing the reachability gramian and other relevant functions and caching redundant calculations for each node, together with an overall smaller number of nodes, will make the proposed approach be more efficient in the long run. In future work, we will implement the proposed steering and distance computations on the state of the art * planners, to quantitatively compare performance benefits of this trade-off.

Indeed, computing geodesics in the trajectory manifold (the equivalent of the Euclidian distance in kinematic systems) is common to both RRT and RRT*-type algorithms, making transferring the proposed methods straightforward.

REFERENCES

[1] B. D. O. Anderson and J. B. Moore. *Optimal Control: Linear Quadratic Methods*. Dover Publications, INC, 1990.

[2] E. Glassman and R. Tedrake. A quadratic regulator-based heuristic for rapidly exploring state space. *IEEE International Conference on Robotics and Automation*, pages 5021–5028, 2010. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=5509718.

[3] J. Hauser. A Projection Operator Approach to the Optimization of Trajectory Functionals. *IFAC Wolrd Congress*, 2002. URL http://www.nt.ntnu.no/users/skoge/prost/proceedings/ifac2002/data/content/03043/3043.pdf.

[4] J. Hauser. On the computation of optimal state transfers with application to the control of quantum spin systems. *American Control Conference*, pages 2169 – 2174, 2003. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1243395.

[5] J. Hauser and D. G. Meyer. The Trajectory Manifold of a Nonlinear Control System. *IEEE Conference on Decision and Control*, pages 1034–1039, 1998. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=760833.

[6] J. P. Hespanha. *Linear Systems Theory*. Princeton university press, 2009.

[7] L. Janson and M. Pavone. Fast Marching Trees: a Fast Marching Sampling-Based Method for Optimal Motion Planning in Many Dimensions . *International Symposium on Robotics Research*, 2013. URL http://www.stanford.edu/~pavone/papers/Janson.Pavone.ISRR13.pdf.

[8] J. H. Jeon, H. Karaman, and E. Frazzoli. Anytime computation of time-optimal off-road vehicle maneuvers using the RRT*. *IEEE Conference on Decision and Control*, pages 3276 – 3282, 2011. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6161521.

[9] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, pages 846–894, 2011. URL http://ijr.sagepub.com/content/30/7/846.short.

[10] S. M. Lavalle and J. J. Kuffner. Randomized Kinodynamic Planning. *The International Journal of Robotics Research*, pages 378–400, 2001. URL http://ijr.sagepub.com/content/20/5/378.short.

[11] Y. Li, Z. Littlefield, and K. E. Bekris. Sparse Methods for Efficient Asymptotically Optimal Kinodynamic Planning. *Workshop on the Algorithmic Foundations of Robotics*, 2014. URL http://www.cs.rutgers.edu/~kb572/pubs/stable_sparse_rrt_WAFR14_LLB.pdf.

[12] Z. Littlefield, Y. Li, and K. E. Bekris. Efficient sampling-based motion planning with asymptotic near-optimality guarantees for systems with dynamics. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1779 – 1785, 2013. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6696590.

[13] A. Perez, R. Platt, G. Konidaris, L. Kaelbling, and T. Lozano-Perez. LQR-RRT*: Optimal sampling-based motion planning with automatically derived extension heuristics. *IEEE International Conference on Robotics and Automation*, pages 2537 – 2542, 2012. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6225177.

[14] D. J. Webb and J. van den Berg. Kinodynamic RRT*: Asymptotically optimal motion planning for robots with linear dynamics. *IEEE International Conference on Robotics and Automation*, pages 5054–5061, 2013. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6631299.