Multi-Query Feedback Motion Planning with LQR-Roadmaps

Anirudha Majumdar
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA, USA
Email: anirudha@seas.upenn.edu

Mark Tobenkin and Russ Tedrake
MIT Computer Science and Artificial Intelligence Lab
Cambridge, MA, USA
Email: {mmt,russt}@mit.edu

***Abstract*—Here we present an algorithm which addresses the need to produce high performance, provably stable feedback controllers for constrained nonlinear systems with goals and constraints that are not fully specified until runtime. Our approach is to precompute a multi-query directed "roadmap", with each segment representing a locally optimal trajectory of the system and a continuous family of verified finite-time invariant sets, or "funnels", associated with an LQR controller computed for the trajectory. The result is a library of parameterized local feedback skills which can be efficiently assembled into a provably stable feedback controller at runtime that takes the system from any point in a bounded region in state space to any stabilizable goal state. The paper makes a number of technical contributions, including formulations for exact algebraic verification using Sums-of-Squares optimization of Lyapunov stability for sets of stabilizable goal states - exploiting the state-dependent riccati equations (SDRE) - and for parameterized finite-time invariance around a trajectory, as well as a "roadmap" construction algorithm that provides probabilistic feedback coverage for any runtime goal in the stabilizable set of the system. We demonstrate our approach with a number of numerical examples.**

I. Introduction

What would it take to say that the control problem was completely solved for a difficult nonlinear system? In this paper, we provide an algorithm which takes as input a description of a smooth nonlinear system and constraints, then *compiles* and *verifies* a parameterized feedback control system which can take the system from any initial condition to any reachable, stabilizable goal. For any particular goal state, we provide a controller which requires only lightweight computation at runtime, and also provide a Lyapunov certificate proving that the system is stable for that goal.

This work is inspired by the challenging control problems presented by model underactuated systems, such as the Cart-Pole system, the Acrobot, the Furuta Pendulum, and others[13, 11, 5, 14]. For systems of this complexity, the algorithm provided here can generate probabilistically complete plans with strong guarantees of success.

Our approach is to generate a sparse "roadmap" of locally optimal trajectories and stabilizable goals for the system. Each of these is stabilized with local feedback - in this paper we focus primarily on feedback design using the linear quadratic regulator (LQR), but other stabilization techniques with algebraic solutions could also be used. We then compute a Lyapunov function which verifies the stability of the local controller over some finite region of state space surrounding the nominal trajectory or goal; these functions can be visualized as "funnels" around the nominal trajectories. The result is a network of funnels which can transport the system through state space from a wide variety of initial conditions to a wide variety of final conditions. The randomized roadmap construction algorithm we provide efficiently covers the state space with these funnels, reusing existing funnels when possible to encourage sparsity.

II. Background

This work is directly motivated by recent work on the "LQR-Trees" algorithm for feedback motion planning[15]. Inspired by Burridge, Rizzi, and Koditschek's picture of feedback motion planning as sequential composition of locally valid feedback policies[3], or funnels, LQR-Trees combines local feedback controller design and verification with randomized trajectory motion planning in order to cover the state space with a feedback controller for a *single goal*. In the motion planning literature, this is related to the concept of a "single-query" planner[9, 10]. In order to stabilize a different goal, the algorithm would have to be restarted from scratch and build a brand new tree.

In the trajectory motion planning literature, a natural extension to the single-query planning algorithms are the "multi-query" algorithms, such as the Probabilistic Roadmaps (PRM) algorithm[7], which build a roadmap of feasible-paths between collision free configurations. These algorithms reuse feasible trajectories so that new queries, consisting of a start state and a goal state, can be performed by simply connecting the start and goal to the existing roadmap, then performing graph search. Although the original PRM was for undirected graphs, directed graphs are also possible[8].

Extending the LQR-Trees algorithm to the multi-query case is particularly exciting because each branch of an LQR-Tree is relatively expensive to compute - the verification step requires solving large convex optimization problems - so reusing these branches for different goals is very appealing. However, constructing valid roadmaps is significantly more challenging, as we must ensure that every path through the graph results in an verified combination of the local Lyapunov functions. Accomplishing this requires the development of new machinery for creating parameterized controllers and funnels, which we develop in the following sections. As a result of this work, we can obtain a strong result - a feedback controller and proof of stability that we can drive the system to any stabilizable goal in the roadmap.

There are other methods which attempt to produce strong guaranteed stability with goals and constraints specified at runtime. For linear systems with quadratic costs and affine constraints, model-predictive control (MPC)[4] can provide guaranteed convergence, even for the case of bounded uncertainty[2], but these guarantees do not extend to the case of a nonlinear plant. Similarly, Hofmann and Williams [6] develops a rich language for specifying and verifying spatial and temporal plans for linear (or feedback-linearized) systems, by constructing flow-tubes which are analagous to the funnels we construct here. Tomlin et al. [17] develop techniques for safety verification and planning on nonlinear systems using the concepts of reachability; this verification is based on elegant solutions to Hamilton-Jacobi equations on a discretization of the state space. By contrast, our verification is computed algebraically on the nonlinear system using Sum-of-Squares (SOS) optimization[12]; while still expensive, these techniques scale polynomially with the dimension of the state space.

III. Approach

In the following sections, we develop the machinery required to make the multi-query algorithm possible. The particular design decisions we have made favor extensive precomputation in order to achieve efficient runtime performance; as a rule we say that no complex optimization procedures should be required at runtime.

In Section IV, rather than try to densely sample individual goal states, we provide machinery which can produce a continuous family of feedback controllers and associated regions of attractions for a volume of the stabilizable goal space. This construction permits a stronger notion of completeness, and actually affords a major advantage in planning because, as we will see in Section IV-D, we can develop large regions of attraction for connected sets of stabilizable goals.

In Section V-A, we develop tools for computing families of finite-time positive invariance (i.e. funnels) around trajectories, by requiring a particular form of exponential stability from the trajectory stabilization. This formulation allows us to rescale the size of funnels, as is required in order to adapt funnels to runtime constraints, and to reuse funnels for driving the system to different goals (each of which might have a different size verified region of attraction).

Finally, in Section VI, we develop the algorithm for combining these locally stabilized and verified solutions into a roadmap of controllers which can drive the system from any initial condition to any stabilizable final condition.

IV. Simultaneous Stabilization of Goal Sets

Standard region-of-attraction analysis using SOS verifies stability to a single goal[18]. In this Section, we develop a method for computing controllers and the corresponding regions of attraction for a continuously parameterized *set* of goal states using a *single* SOS program, and the solution of a small number of Lyapunov equations.

We are given a controlled nonlinear system:

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

with $x \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$. We assume we are given a set of stabilizable goals, parameterized by $\delta \in P \subset \mathbb{R}^p$. These goals are given by continuous function $x_g : P \mapsto \mathbb{R}^n$ and another function $u_g : P \mapsto \mathbb{R}^m$ is given such that $f(x_g(\delta), u_g(\delta)) = 0$ for all $\delta \in P$. Finally, for simplicity, we assume $0 \in P$. More detail as to the admissible parameterizations is given in the next section. Below we design a linear feedback gain K_δ such that $x_g(\delta)$ is a exponentially stable equilibrium of (1) with $u = \pi_\delta(x) := u_g(\delta) - K_\delta(x - x_g(\delta))$.

To design our controller we exploit linearized control. Recall that the solution to the problem of stabilizing a linear system, $\dot{\bar{x}} = A\bar{x} + B\bar{u}$, about the origin, subject to the quadratic cost

$$J(\bar{x}) = \int_0^\infty \left[\bar{x}^T(t) Q \bar{x}(t) + u^T(t) R u(t)\right] dt$$
$$Q = Q^T \geq 0, R = R^T > 0.$$

is given by the cost-to-go $J^\star = \bar{x}^T S \bar{x}$, where S is the positive definite solution to the Algebraic Riccati Equation:

$$0 = Q - SBR^{-1}B^T S + SA + A^T S. \tag{2}$$

The corresponding optimal feedback policy for the linear system is given by $\bar{u}^\star = -R^{-1}B^T S \bar{x} = -K\bar{x}$.

Setting $A = A_\delta$ and $B = B_\delta$ with:

$$A_\delta = \frac{\partial f}{\partial x}(x_g(\delta), u_g(\delta)), \qquad B_\delta = \frac{\partial f}{\partial u}(x_g(\delta), u_g(\delta)).$$

and taking K_δ to be the resulting feedback gain, K, already provides a continuous family of controllers which stabilize each $\{x_g(\delta)\}_{\delta \in P}$. However, to allow for simultaneous verification of all of the controllers, we seek a suboptimal, algebraic parameterization of the controllers.

We address this task using a variant of the state-dependent Riccati equation (SDRE) [1]. We look to approximately satisfy the Riccati equation (2) for the perturbed linear system with

$$A = A_0 + \Delta A, \quad B = B_0 + \Delta B,$$

recovering a controller for (A_δ, B_δ) with $\Delta A = A_\delta - A_0$, $\Delta B = B_\delta - B_0$. We expand ΔA and ΔB in some basis:

$$\Delta A = \sum_{i=1}^{n^2} a_i E_i, \quad \Delta B = \sum_{j=1}^{nm} b_j F_j$$

where E_i and F_j are linearly independent matricies spanning $\mathbb{R}^{n \times n}$ and $\mathbb{R}^{n \times m}$ respectively.

We now write the Riccati solution and optimal controllers as a Taylor expansion in the vectors a and b:

$$S(a, b) = S_0 + \sum_i a_i S_i^{(a)} + \sum_j b_j S_j^{(b)} + \dots,$$

$$K(a,b) = R^{-1}(B_0 + \Delta B)^T S(a,b).$$

Substituting A, B and $S(a,b)$ into (2) determines the terms of the expansion (e.g. $S_i^{(a)}$ and $S_j^{(b)}$) as solutions of independent Lyapunov equations.

Let K_δ and S_δ be the truncations of $K(a,b)$ and $S(a,b)$ where a and b are determined by A_δ and B_δ. This feedback law and quadratic form are polynomials in A_δ and B_δ.

A. Simultaneous Verification

We now provide conservative esimates of the region of attraction for each $x_g(\delta)$ using techniques from robust verification with Sum-of-Squares optimization [18]. We use the approximate Riccati solution S_δ to provide a parameterized candiate Lyapunov function:

$$V(\delta, x) = (x - x_g(\delta))^T S_\delta (x - x_g(\delta)).$$

with the derivative along solutions given by:

$$\frac{d}{dt}V(\delta, x) = 2(x - x_g(\delta))^T S_\delta f(x, \pi_\delta(x)).$$

For each $\delta \in P$, we aim to find a maximal $\rho : P \mapsto \mathbb{R}_+$ such that:

$$V(\delta, x) \leq \rho(\delta) \implies \frac{d}{dt}V(\delta, x) \leq -cV(\delta, x)$$

which verifies exponential stability (here $c > 0$ is a fixed positive constant.

By construction S_δ and K_δ are polynomials in δ. When f, x_g, and u_g are polynomials and P is semialgebraic set, defined by polynomial equations and inequalities:

$$P = \{\delta \mid \max_i g_i(\delta) \leq 0, h(\delta) = 0\},$$

with $g : \mathbb{R}^p \mapsto \mathbb{R}^{p_g}$ and $h : \mathbb{R}^p \mapsto \mathbb{R}^{p_h}$, we can directly address this problem using Sum-of-Squares programming. The next section gives an example from a broader class of systems which can still be addressed using SOS programming.

We parameterize $\rho(\delta)$ as a fixed degree polynomial in δ. We additionlly search over a "multiplier" polynomial $\nu : \mathbb{R}^n \times \mathbb{R}^p \mapsto \mathbb{R}$ of fixed degree. The ideal form of the SOS program is:

$$\begin{aligned} \underset{\rho,\nu}{\text{maximize}} \quad & \int_P \rho(\delta)\, d\delta \\ \text{subject to} \quad & \|x - x_g(\delta)\|^2 (V(\delta,x) - \rho(\delta)) \\ & + \nu(x,\delta)\left(\frac{d}{dt}V(\delta,x) + cV(\delta,x)\right) \geq 0 \\ & \rho(\delta) \geq 0 \qquad \forall \delta \in P, x \in \mathbb{R}^n \end{aligned}$$

Enforcing these constraints only for $\delta \in P$ is handled by introducing further multipliers [12]. Depending on the complexity of P, the integral may need to be approximated.

One can verify that any feasible solution of the above satisfies that for $\delta \in P$ we have $\rho(\delta) \geq 0$ and $V(x,\delta) \geq \rho(\delta)$ whenever $\frac{d}{dt}V(\delta,x) + cV(\delta,x) = 0$ and $x \neq x_g(\delta)$. Combined with the knowledge that the Hessian of $\frac{d}{dt}V(\delta,x) + cV(\delta,x)$ is negative definite in a neighborhood about $x_g(\delta)$, this certifies the exponential stability of the closed loop system.. A similar SOS program can be used to verify this second condition. If the program is infeasible, P must be reduced in size.

Note that although the feedback and Lyapunov candidates are approximations of the local optimal control solutions, the verification procedure here is exact. Thus, issues such as convergence of the Taylor series used to represent the Lyapunov functions do not affect the validity of the guarantees obtained from the SOS program.

B. Numerical Examples

The procedure described above for designing and verifying controllers for regions in the stabilizable space of a nonlinear system was tested on the Van der Pol oscillator, $\ddot{x} = \dot{x}(1 - x^2) + x + u$. We took $P = [-0.5, 0.5]$ (i.e. $g(\delta) = (\delta - 0.5)(\delta + 0.5)$) and $x_g(\delta) = [\delta; 0]$.

Figure 1 compares regions of attraction estimated for sample goal states via Sum-of-Squares verification with the LQR solution against the approximate controllers proposed above. For $\delta = 0$ the approximations are poor due to the restriction of ρ being polynomial. Higher-order expansion of S_δ and K_δ result in improved performance, and low-order approximations perform well.

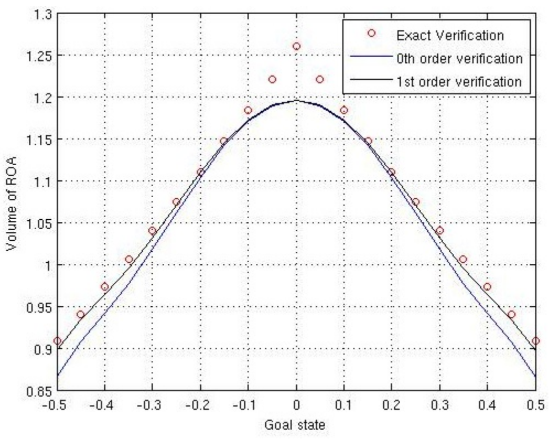


Fig. 1. Volumes of regions of attraction for the Van der Pol oscillator for various order expansions of the control. All verifications are exact.

C. Stabilizable Manifolds

In the Van der Pol example considered above, the set of stabilizable points lie along the x-axis in state space. However, in general, the set of stabilizable points is a complicated, disconnected manifold in the configuration space (derivative variables are always zero at a stabilizable point), defined by $0 = f(x,u)$, plus any input and state constraints. In many cases, this constraint can be simplified to a tractable form. For (potentially over- or under- actuated) robots defined by the manipulator equations:

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = Bu,$$

the constraint reduces to

$$Bu = G(q), \quad \dot{q} = 0.$$

These can be folded into the robust verification by adding equality constraints via Lagrange multipliers.

Consider, for example, the Acrobot as described in Tedrake [14] with torque limits $u_{min} \leq u \leq u_{max}$. The constraint reduces to

$$m_1 l_{c1} s_1 + m_2(l_1 s_1 + l_{c2} s_{1+2}) = 0, u_{min} \leq m_2 g l_{c2} s_{1+2} \leq u_{max},$$

which corresponds to two disconnected stabilizable sets - with the center of mass directly above or below the shoulder. These can be converted to SOS constraints by changing coordinates (of the entire problem) to polynomial, using for instance a stereographic projection: $\sin(\theta_i) = \frac{1-q_i^2}{1+q_i^2}, \cos(\theta_i) = \frac{2q_i^2}{1+q_i^2}$.

D. Control for Path-Connected Goal Sets

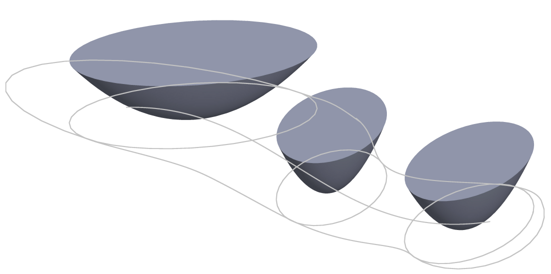


Fig. 2. Illustration of a path of stabilizable goals and the associated region of attraction. Each goal has a quadratic Lyapunov function (three pictured). By switching control laws along the path the system can be moved from any one point's basin of attraction to any other basin in finite time.

Verifying continuous families of goals allows a simple, but very important observation: we can produce a simple controller for which the region of attraction of any goal in the family is the union of the LQR regions of attraction for every goal in the family. Figure 2 provides an illustration.

The previous design and verification technique provides us with a family of goals $G = x_g(P)$ such that:

1) Each $x_g(\delta)$ with $\delta \in P$ is stabilized by the feedback policy $u = \pi_\delta(x)$.
2) There exists a continuous functions $V : P \times \mathbb{R}^n \mapsto \mathbb{R}_+$ and $\rho : P \mapsto \mathbb{R}_+$ such that $V(\delta, x_g(\delta)) = 0$ and $V(\delta, \cdot)$ is a Lyapunov function verifying trajectories in:
$$\Omega_\delta = \{x \mid V(\delta, x_g) \leq \rho(\delta)\},$$
converge exponentially to $x_g(\delta)$ when the policy $\pi_\delta(\cdot)$ is applied.

We have the following Lemma.

Lemma 4.1: Let $V(\cdot,\cdot)$ and $\pi_\cdot(\cdot)$ be functions satsifying the conditons above. If there exists a continuous path $p : [0,1] \mapsto P$ such that $p(0) = \delta$ and $p(1) = \delta'$ then there exists a controller which can take any initial conditon from Ω_δ to $\Omega_{\delta'}$ in finite time.

The proof is omitted due to space constraints, but involves finitely many switches between control laws $\pi_{p(t_i)}(\cdot)$ for increasing $t_i \in [0,1]$.

Hence, this "switching controller" allows us to traverse a path-connected set in the stabilizable region of the system, an important ability that allows us to achieve the goal of reaching *any* stabilizable point.

V. Exponential Stability Along Trajectories

Building a multi-query feedback motion plan efficiently requires reusing partial plans to arrive at different goal states. To do this effectively requires different stability constraints than those proposed in [15]. Figure 6a illustrates the new situation which must be accounted for. In this figure "Funnel 1" has already been planned, and belongs to the roadmap. "Funnel 2" is being added to the roadmap. We see the initial (larger) outlet of Funnel 1 does not fit inside the inlet of Funnel 2 (here due to centering, but more commonly due to shape or size). The stability constraints described below will allows us to certify that every rescaling of Funnel 1 is also finite-time invariant, so that once Funnel 2 is computed we can associate with it the shrunken funnel. Note that this rescaling can be different for every funnel Funnel 1 flows into, and is trivial to compute.

A. Controller Design

This section describes the computation of regions of stability for LQR-stabilized trajectories of the system. For each nominal trajectory $x_0 : [t_0, t_f] \mapsto \mathbb{R}^n$ we design a time-varying controller using LQR. We linearize the dynamics of the system about the trajectory:

$$A(t) = \frac{\partial f}{\partial x}(x_0(t), u_0(t)), B(t) = \frac{\partial f}{\partial u}(x_0(t), u_0(t))$$

The control law is obtained by solving a Riccati differential equation:

$$-\dot{S}(t) = Q - S(t)B(t)R^{-1}B^T S(t) + S(t)A(t) + A(t)^T S(t)$$

with final value conditions $S(t) = S_0$. The quadratic functional:

$$V_0(t,x) = (x - x_0(t))^T S(t)(x - x_0(t))$$

is our locally valid Lyapunov candidate. In particular S_0 provides flexibility in choosing the geometry of the "outlet" of a funnel. It can, for example, be chosen to be contained in the basin of attraction of a stabilized equilibirum.

B. Controller Verification

For each nominal trajectory $x_0 : [t_0, t_f] \mapsto \mathbb{R}^n$ we want to verify solutions which begin nearby remain nearby. To do so, we construct a Lyapunov function $V : [t_0, t_f] \times \mathbb{R}^n \mapsto \mathbb{R}_+$ with $V(t, x_0(t)) = 0$ such that:

$$\frac{\partial V}{\partial t}(t,x) + \frac{\partial V}{\partial x}(t,x)f(t,x) < 0 \quad (3)$$

for all (t,x) with:

$$x \neq 0, \quad \text{and } V(t,x) \leq 1 \quad \text{and} \quad t \in [t_0, t_f].$$

In the region (3) is valid the function $V(t, x(t))$ decreases with time whenever $x(t)$ is a trajectory of the system. For any $c \in (0,1]$ we see that $V(t, x(t)) < c$ for a given $t \in [t_0, t_f]$ implies that $V(\tau, x(\tau)) < c$ for all $\tau \in [t, t_f]$. We can imagine the surfaces where $V(t,x) = c$ as a "funnel" which will not be piereced by trajectories.

Below we describe an optimization approach similar to [15] and [16] for finding such Lyapunov functions around trajectories of nonlinear systems exploiting the Sum-of-Squares technique [12].

Design procedures for time-varying linear feedback laws, like the LQR controller described above, generally provide a quadratic "cost-to-go" or Lyapunov function, say $V_0(t,x)$. When examining the fully nonlinear dynamics, this function is only guaranteed to be valid for infinitesimal regions around a trajectory. We exploit $V_0(t,x)$ as a candiate Lyapunov function, but make use of an additional time-varying rescaling of this function as follows:

$$V(t,x) = \frac{V_0(t,x)}{\rho(t)}$$

where $\rho : [t_0, t_f] \mapsto \mathbb{R}$ is a differentiable, strictly positive function to be determined. An equivalent condition to (3) is:

$$\rho(t)\left(\frac{\partial V_0}{\partial t}(t,x) + \frac{\partial V_0}{\partial x}(t,x)f(t,x)\right) < \frac{d\rho}{dt}(t)V_0(t,x) \quad (4)$$

for all (t,x) with:

$$V_0(t,x) \leq \rho(t) \quad \text{and} \quad t \in [t_0, t_f].$$

Our objective is to maximize $\rho(t)$ while ensuring (4) holds.

We choose a piecewise polynomial basis for $\rho(t)$ and approximate $f(t,x)$ on locally by its Taylor expansion $\hat{f}(t,x)$. This allows us to formulate the condition above as a Sum-of-Squares program using the polynomial $\mathcal{S}$-procedure. For a piecewise-linear $\rho(t)$ the optimization can be done one knot point at a time via binary search by using Sum-of-Squares optimization to verify the condition (4) holds, as in [15]. For more complicated $\rho(t)$ a bilinear alternation technique similar to [16] applies. In the cited works only a single invariant set is guaranteed, i.e. it is only determined that solutions do not exit $V(t,x) \leq 1$.

VI. LQR-Roadmaps

A. The Algorithm

The mechanics of designing feedback controllers and associated funnels developed in the preceding sections can now be assembled into a roadmap with directed edges using the algorithm outlined in Algorithm 1. For each iteration of the algorithm, a random start state, x_s, and a random goal, x_g, are selected. If the goal is not yet stabilized by an existing time-invariant funnel, then a new family of funnels is constructed around this sample point using the procedure described in Section IV. The algorithm immediately attempts to connect the new funnels to the existing roadmap by optimizing a trajectory from the current roadmap to x_g. The algorithm then checks whether x_s is currently a region of state space covered by the branches of the roadmap which connect to x_g; we call this covered region $\mathcal{C}(x_g)$. If it is outside the region, then the algorithm uses trajectory optimization to attempt to connect x_s up to the current roadmap. If this trajectory optimization is successful, a TV-LQR controller and its associated region of stability is computed using the verification procedure described in Section V-A. Although individual trajectory optimization steps are never guaranteed to succeed (since they use only a local method), the algorithm is designed so that failed connection attempts will be retried again later, if necessary, with a different random seed for the trajectory optimization and with an ever expanding roadmap to potentially connect to.

Algorithm 1 LQR-Roadmap

```
1: $\mathcal{M}$ is empty roadmap
2: $\mathcal{G}$ is empty goal region
3: for $k$ = 1 to K do
4:    $(x_s, (x_g, u_g)) \Leftarrow$ random start state, goal state pair
5:    if $x_g \notin \mathcal{G}$ then
6:       $[\delta, K(\delta), S(\delta), \rho(\delta)]$ from time-invariant LQR, §IV
7:       $[t_f, x_0(t), u_0(t)]$ from trajectory optimization with
         $x_0(0) \in \mathcal{M}$ and $x_0(t_f) = x_g$
8:       $\mathcal{M}$.addNode($x_g, u_g, \delta, \mathbf{K}(\delta), \mathbf{S}(\delta), \rho(\delta)$)
9:       $\mathcal{G} \Leftarrow \mathcal{G} \cup \{x : x_g + \delta\}$
10:      if trajectory optimizer succeeds then
11:         $[K(t), S(t), \rho(t)]$ from time-varying LQR, §V-A
12:         Compute funnel scaling, $c$; $0 \leq c \leq 1$, §??
13:         $\mathcal{M}$.addTrajectory($t_f, x_0(t), u_0(t), K(t), S(t), \rho(t), c$)
14:      end if
15:   end if
16:   Compute $\mathcal{T}(x_g)$ := subset of $\mathcal{M}$ connected to $x_g$, §??
17:   $\mathcal{C}(x_g)$ := space covered by trimmed funnels on $\mathcal{T}(x_g)$
18:   if $x_s \notin \mathcal{C}(x_g)$ then
19:      $[t_f, x_0(t), u_0(t)]$ from trajectory optimization with
         $x_0(0) = x_s$ and $x_0(t_f) \in \mathcal{T}(x_g)$
20:      if trajectory optimizer succeeds then
21:         $[K(t), S(t), \rho(t)]$ from time-varying LQR, §V-A
22:         $\mathcal{M}$.addTrajectory($t_f, x_0(t), u_0(t), K(t), S(t), \rho(t)$)
23:      end if
24:   end if
25: end for
```

Note that Algorithm 1 produces a roadmap, $\mathcal{M}$ with exploitable structure. In particular, it contains no loops: connections are made from a random x_s to $\mathcal{M}$ and from $\mathcal{M}$ to a random x_g, but never from $\mathcal{M}$ to itself. This makes it particularly simple to quickly determine for any particular goal, x_g, which edges of $\mathcal{M}$ contain a path which can reach that goal. Furthermore, with only one exception, all of the time-varying feedback and funnels are described via the solution to a smooth continuous Riccati equation from a random start state, through branches of the roadmap, all of the way to the goal; e.g. along these trajectories there are no discontinuities in $K(t)$ or $S(t)$. The exception is at the point where a new goal state is connected to an existing roadmap - we affectionately call these trajectories "off-ramps" from the roadmap. For these off-ramps, there is no reason to expect the Riccati solution generated from the new goal state back along the trajectory to the roadmap to match with the solution $S(t)$ of the existing map. At this point, we must compute a scaling factor, c, by which all preceeding funnels must be shrunk in order to have the funnels on the roadmap land inside the funnels of the new off-ramp.

Exploiting this structure, and the form of stability verified in the time-varying trajectories, we can quickly compute the subset of $\mathcal{M}$ which connects to any stabilized x_g and the *trimmed* set of funnels which provide guarantees for driving the system to this goal. In fact, this subset is an LQR-Tree[15], which we denote $\mathcal{T}(x_g)$. Note that, if a branch of this tree passes through an intermediate time-invariant verified region on its path to x_g, that the funnel for this time-invariant region does not have to be trimmed, regardless of how small the inlet to the next branch is, because the system can get arbitrarily close to the nominal node before continuing on down the tree.

B. Probabilistic Feedback Coverage

The LQR-Roadmap algorithm achieves a unique notion of completeness. We are given a bounded sampling region of stabilizable goals, $\mathcal{X}_G$, and a bounded sampling region of potential initial conditions, $\mathcal{X}_S$. Let $\mathcal{G}_k$ be the the goal region of stabilized goals at iteration k and $\mathcal{G}_\infty = \bigcup_{k=1}^{\infty} \mathcal{G}_k$. First, we demonstrate that $\text{cl}(\mathcal{G}_\infty) = \text{cl}(\mathcal{X}_G)$ w.p.1., where $\text{cl}(\cdot)$ denotes closure. We show a similar coverage of the points which can reach each x_g in $\mathcal{G}_\infty$ by the points the roadmap can stabilize to x_g. Using the terminology of [15], we say the algorithm achieves *probabilistic feedback coverage*. Due to space limitations, we only provide a sketch the proof here; it builds directly on the proof provided in [15].

Any open set in $\mathcal{X}_G$ which is not in $\mathcal{G}_k$ has a non-zero probability of being sampled from, stabilized, and added to the roadmap. Due to the robust verification over goals, a neighborhood of the sampled point will be added to $\mathcal{G}_k$ with each addition. Therefore, w.p.1, $\text{cl}(\mathcal{G}_\infty) = \text{cl}(\mathcal{X}_G)$ as $k \to \infty$.

As every $x_g \in \mathcal{G}_k$ shares a region of attraction and LQR-Tree, $\mathcal{T}(x_g)$ with every other point in the non-zero measure δ-region of time-invariant verification, the tree $\mathcal{T}(x_g)$ has a non-zero probability of being evaluated on every iteration of the algorithm. Denote $\mathcal{C}_k(x_g)$ as the region of state space covered by the (trimmed) $\mathcal{T}(x_g)$ after the kth iteration of the algorithm. Denote $\mathcal{R}(x_g)$ as the set of states from which x_g is reachable, and $\mathcal{R}_S(x_g)$ as the subset of $\mathcal{R}(x_g)$ contained in $\mathcal{X}_S$.

To arrive at the further claim we assume that a trajectory optimizer with suitably randomized initial seed will have a lower bounded probability of connecting an initial condition in $\mathcal{R}_S(x_g)$ to $\mathcal{T}(x_g)$ at any iteration. Every open subset in $\mathcal{R}(x_g)$ which is not in $\mathcal{C}_k(x_g)$ has a non-zero probability of being sampled. By our assumption on the trajectory optimizer there is a non-zero probability this point will be reconnected to the tree.

C. Numerical Example

Numerical simulations on a torque-limited pendulum help to illustrate the dynamics of the LQR-Roadmaps algorithm. The dynamics of the pendulum are given by $I\ddot{\theta}+b\dot{\theta}+mglsin(\theta) = \tau$, where $m = 1$, $l = 0.5$, $b = 0.1$, $I = 0.25$, $g = 9.8$ and $\tau_{max} = 3$. Figures 3(a) through 3(c) show the algorithm covering the region $[0, 2\pi] \times [-5, 5]$ in the 2-dimensional state space of the system with trajectories that grab points and distribute them to the stabilizable region in state space. Note that the torque limits imposed on the system limit the set of stabilizable points to two disconnected intervals along the x-axis. These two intervals are marked in black in the figure.

Figure 3(a) shows the first branch of the roadmap connecting a randomly chosen point in state space to a point in the stabilizable region. Once the two points have been connected, the algorithm proceeds by performing time-invariant verification for an interval of length 0.8 centered around the nominal goal point. This interval is shaded in red in the figure. Figure 3(b) shows another randomly chosen goal state being connected back up to the existing roadmap. This process continues till the space has been probabilistically covered.

Figure 4(a) through 4(c) show an LQR-Roadmap being constructed for the Cart-Pole system. This system is underactuated and has a 4-dimensional state space. Figure 4(a) shows three initial branches in the roadmap with trajectories and funnels projected in the $\dot{x}$ - x space. Figure 4(b) and 4(c) show the roadmap on the cartpole when it is close to filling up the space.

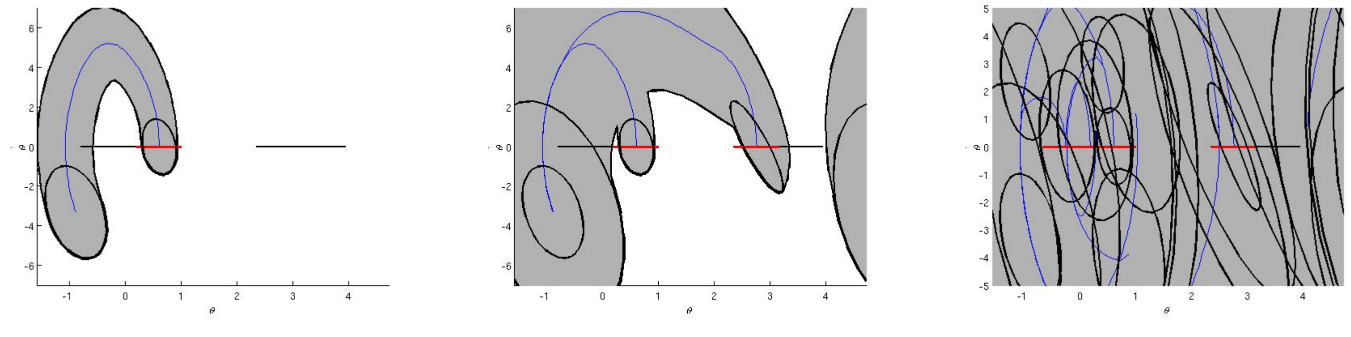
An important step in the LQR-Roadmaps algorithm is the "pruning" of funnels that follows the addition of a goal state that connects back up to the roadmap. This process is illustrated in Figures 5(b) and 5(c). Figure 5(b) shows a new "off-ramp" that has just been added to the roadmap and its corresponding funnel. Since the inlet of the funnel corresponding to this trajectory is smaller than the funnel of the trajectory to which it connects, pruning of trajectories must be carried out. Thus, the algorithm shrinks the funnel of the branch that the new trajectory connected up to along with every trajectory "upstream" from that branch. Figure 5(c) demonstrates this shrinking procedure.

VII. Discussion

In this section, we discuss some possible extensions to the LQR-Roadmaps algorithm, along with some of its limitations.

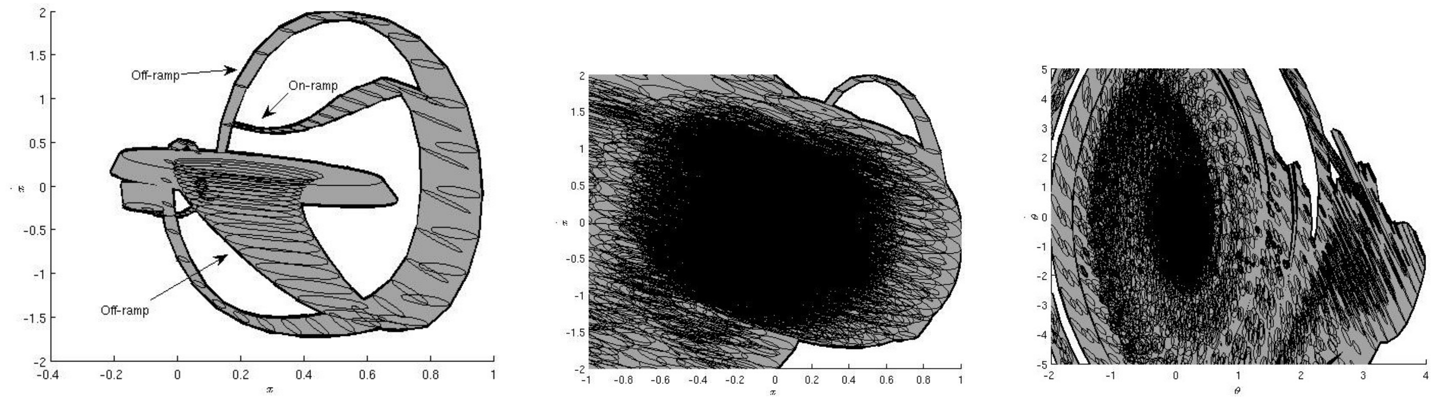
One exciting possible extension that could be easily added to the LQR-Roadmap algorithm in its current form would be to allow for the handling of runtime geometric constraints. The ability to "prune" funnels in a quick and easy way allows us to adjust the roadmap in order to avoid collisions with obstacles that were unknown at the roadmap construction phase. In addition, it may be possible to verify continuous families of trajectories contained in tubes around the nominal branches of the roadmap. Given a set of geometric constraints at runtime, a suitable trajectory from the tube around the nominal branch could be chosen in order to avoid collisions. It may also be possible to explicitly consider the space of possible constraints during the roadmap construction phase, especially for domain specific applications where one has a good idea of the kinds of constraints that may be encountered.

The algorithm presented here provides a strong notion of completeness: probabilistic feedback coverage. For high-dimensional state spaces where it is impractical to compile these plans for every starting and goal state, the algorithm can either cover a low-dimensional subset of the state space or



(a) 1 branch (b) 2 branches (c) 9 branches

Fig. 3. LQR-Roadmap construction for the pendulum. The two black intervals along the θ axis represent the stabilizable points of the torque-limited pendulum. The red intervals represent the stabilizable goal states for which regions of attraction have been computed. The gray regions around the blue trajectories are the regions of stability for the LQR controllers corresponding to the trajectories.



(a) (b) (c)

Fig. 4. LQR-Roadmaps on the underactuated Cart-Pole system, (a) with 3 initial branches and (b-c) close to filling the space. Note that the funnels depicted in the figure are slices of the funnels in 4-d projected down to the 2-dimensional subspaces, which appears artificially dense in the 2-d plot.

provide a verified partial plan for the high-dimensional system. Whenever a runtime goal is outside of the verified region, one can quickly solve for and apply LQR controller, then perform the verification and addition to the map offline.

The major limitation of the LQR-Roadmaps algorithm is the computational cost associated with the roadmap construction phase. The verification of LQR controllers via Sum-of-Squares optimization techniques is the major contributing factor to this computational cost. There have been results computing funnels for trajectories up to six state variables[16], and others report time-invariant regions-of-attraction using quadratic Lyapunov functions with considerably more dimensions[18].

Finally, a practical limitation of the LQR-Roadmaps algorithm is its reliance on a good plant model. One way to tackle the problem may be to compute robust feedback controllers and certificates for systems whose descriptions include bounded model parameter uncertainties. This robust verification fits directly into the Sum-of-Squares formulations presented here[18].

VIII. Conclusion

In this paper, we have presented the LQR-Roadmaps algorithm, which combines Sums-of-Squares techniques for computing regions of attraction for LQR controllers with randomized motion planning algorithms in order to create a multi-query “roadmap” that can take a non-linear system from any point in a bounded region in state space to any stabilzable point of the system. The technical contributions made in this paper include the exact algebraic verification using Sum-of-Squares optimization of Lyapunov stability around a parameterized goal and a scheme for verifying exponential stability around trajectories. We demonstrate our roadmap construction algorithm along with probabilistic feedback coverage on a two-dimensional system.

Our future work will include exploring the dynamics of the LQR-Roadmap algorithm on systems of higher dimension and with more complicated stabilizable manifolds. Further, we plan to extend our current framework to handle geometric constraints that become available only at runtime.

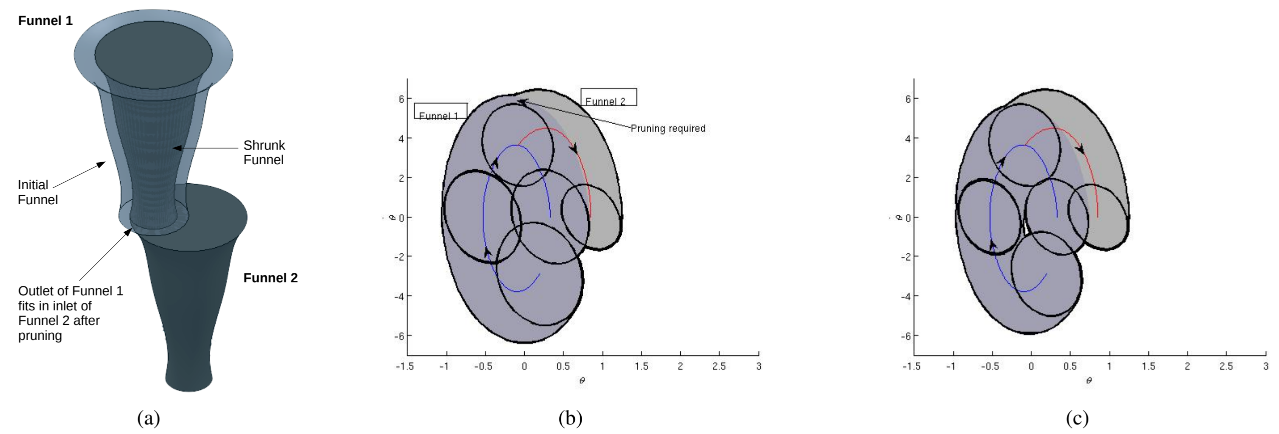


Fig. 5. Pruning of funnels. (b) Before pruning: The inlet of Funnel 2 lies inside Funnel 1; trajectories in Funnel 1 will not necessarily end up in Funnel 2. Hence, Funnel 2 (and all other funnels that connect up to it) needs to be pruned. (c) After pruning: The inlet of Funnel 2 fits snugly around Funnel 1.

References

[1] H. T. Banks, B. M. Lewis, and H. T. Tran. Nonlinear feedback controllers and compensators: a state-dependent riccati equation approach. *Computational Optimization and Applications*, 37(2):177–218, June 2007.

[2] Alberto Bemporad and Manfred Morari. Robust model predictive control: A survey. In A. Garulli and A. Tesi, editors, *Robustness in identification and control*, volume 245 of *Lecture Notes in Control and Information Sciences*, pages 207–226. Springer Berlin / Heidelberg, 1999. 10.1007/BFb0109870.

[3] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek. Sequential composition of dynamically dexterous robot behaviors. *International Journal of Robotics Research*, 18(6):534–555, June 1999.

[4] E. F. Camacho and Carlos Bordons. *Model Predictive Control*. Springer-Verlag, 2nd edition, 2004.

[5] Isabelle Fantoni and Rogelio Lozano. *Non-linear Control for Underactuated Mechanical Systems*. Communications and Control Engineering Series. Springer-Verlag, 2002. ISBN 1-85233-423-1.

[6] Andreas G. Hofmann and Brian C. Williams. Exploiting spatial and temporal flexibility for plan execution of hybrid, under-actuated systems. In *Proceedings of the American Association for Artificial Intelligence (AAAI)*, 2006.

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

[8] R. Kindel, D. Hsu, J.-C. Latombe, and S. Rock. Kinodynamic motion planning amidst moving obstacles. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 537 –543 vol.1, 2000.

[9] J.J. Kuffner and S.M. Lavalle. RRT-connect: An efficient approach to single-query path planning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 995–1001, 2000.

[10] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

[11] Richard M. Murray and John Hauser. A case study in approximate linearization: The acrobot example, April 1991.

[12] Pablo A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, May 18 2000.

[13] Mark W. Spong. Partial feedback linearization of underactuated mechanical systems. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, volume 1, pages 314–321, September 1994.

[14] Russ Tedrake. *Underactuated Robotics: Learning, Planning, and Control for Efficient and Agile Machines: Course Notes for MIT 6.832*. Working draft edition, 2010.

[15] Russ Tedrake, Ian R. Manchester, Mark M. Tobenkin, and John W. Roberts. LQR-Trees: Feedback motion planning via sums of squares verification. *International Journal of Robotics Research*, 29:1038–1052, July 2010.

[16] Mark M. Tobenkin, Ian R. Manchester, and Russ Tedrake. Invariant funnels around trajectories using sum-of-squares programming. *arXiv:1010.3013 [math.DS]*, 2010.

[17] Claire J. Tomlin, Ian M. Mitchell, Alexandre M. Bayen, and Meeko K. M. Oishi. Computational techniques for the verification and control of hybrid systems. In *Multidisciplinary Methods for Analysis Optimization and Control of Complex Systems*, Mathematics in Industry, pages 151–175. Springer Berlin Heidelberg, 2005.

[18] Topcu, U., Packard, A.K., Seiler, P., Balas, and G.J. Robust region-of-attraction estimation. *IEEE Transactions on Automatic Control*, 55(1):137 –142, Jan 2010.