A Quadratic Regulator-Based Heuristic for Rapidly Exploring State Space

Elena Glassman and Russ Tedrake

***Abstract*— Kinodynamic planning algorithms like Rapidly-Exploring Randomized Trees (RRTs) hold the promise of finding feasible trajectories for rich dynamical systems with complex, nonconvex constraints. In practice, these algorithms perform very well on configuration space planning, but struggle to grow efficiently in systems with dynamics or differential constraints. This is due in part to the fact that the conventional distance metric, Euclidean distance, does not take into account system dynamics and constraints when identifying which node in the existing tree is capable of producing children closest to a given point in state space. We show that an affine quadratic regulator (AQR) design can be used to approximate the exact minimum-time distance pseudometric at a reasonable computational cost. We demonstrate improved exploration of the state spaces of the double integrator and simple pendulum when using this pseudometric within the RRT framework, but this improvement drops off as systems' nonlinearity and complexity increase. Future work includes exploring methods for approximating the exact minimum-time distance pseudometric that can reason about dynamics with higher-order terms.**

I. INTRODUCTION

Kinodynamic motion planning algorithms attempt to find feasible trajectories for a dynamical system from a start state to a goal state while respecting constraints on position, velocity, and/or acceleration. The problem is believed to be at least PSPACE-hard [1], however a number of randomized algorithms have been proposed which can achieve fast average-time performance for a large variety of problems [2], [3], [4], [5], [7].

A common theme running through many path-planning algorithms is some notion of distance in the space in which trajectories lie. In algorithms that attempt to create roadmaps, paths are found between *neighboring* nodes. In the Rapidly Exploring Random Tree (RRT) algorithm, nodes of a tree are grown toward randomly selected goals; only the node that is *closest* to the randomly selected goal is expanded [4], [5].

The distance function that maps two points to a distance score can be defined however the user sees fit. It is a pseudometric, since it does not need to meet the formal requirements of a metric, such as symmetry. It provides the user with the opportunity to incorporate his/her prior knowledge about the problem: he/she defines what makes two nodes neighbors in a roadmap, or what makes a point close enough to a goal state for a path to be considered complete, or to which nodes it is least costly to steer the system, from some specified initial state (i.e., cost-to-go).

E. Glassman is with the Department of Electrical Engineering and Computer Science, MIT, Cambridge, MA 02139, USA elg@mit.edu

R. Tedrake is the X Consortium Associate Professor of Electrical Engineering and Computer Science, MIT, Cambridge, MA 02139, USA russt@mit.edu

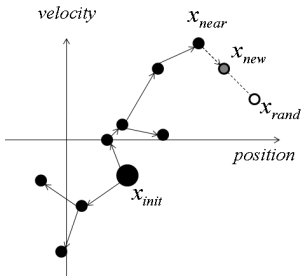


Fig. 1. Illustration of one iteration of growing an RRT, adapted from [8].

```
1: procedure BUILD_RRT(x_init)
2:     T.init(x_init);
3:     for k = 1 to K do
4:         x_rand ← RANDOM_STATE();
5:         EXTEND(T, x_rand)
6:     end for
7:     Return T
8: end procedure
9: procedure EXTEND(T, x)
10:    x_near ← NEAREST_NEIGHBOR(x, T);
11:    if NEW_STATE(x, x_near, x_new, u_new) then
12:        T.add_vertex(x_new);
13:        T.add_edge(x_near, x_new, u_new);
14:    end if
15: end procedure
```

Fig. 2. The basic algorithm for constructing RRTs, adapted from [8].

The performance of the RRT, a particularly popular and simple randomized path-planning algorithm that is currently one of the most promising methods for planning in phase space and for solving other problems with differential constraints [13], can vary greatly as a function of the definition of distance [8]. LaValle and Kuffner asserted that an exact, quickly computable distance pseudometric would address remaining barriers to more efficiently exploring state space. The basic RRT algorithm is shown in Fig. 2, and illustrated in Fig. 1. In this work, the EXTEND function creates multiple x_{new} (child) nodes using each of a set of specified actions, and the child node which is closest to x_{rand}, according to the NEAREST_NEIGHBOR function, is added to the RRT. The definition of distance has its effect by determining which node the NEAREST_NEIGHBOR function returns for the RRT to extend.

For our proposed distance function, we use a finite-horizon affine quadratic regulator (AQR) to calculate optimal cost-

to-go functions of linearizations of the plant for multiple time horizons in order to locally approximate the optimal distance measure. In Section II, we place this in the context of previously proposed metrics. We elaborate on the problem formulation in Section III and give a more detailed explanation of our metric and our method of evaluating it in Section IV. In Section V, we show RRT trees grown on four benchmark systems and compare outcomes when using our distance function to the outcomes obtained when using a standard distance function. Finally, in Section VI, we discuss the results and future work.

II. RELEVANT LITERATURE

An ideal distance pseudometric is the optimal cost-to-go function [10]. Even if an exact pseudometric does not exist or cannot be computed efficiently, LaValle and Kuffner have argued that approximations will still dramatically improve performance. In the process of developing the idea, we found that LaValle and Kuffner have mentioned the possibility of using cost-to-go functions from applying optimal control to linearized systems, as part of a list of many possibilities, including Lyapunov functions, fitted spline curves, and steering methods [6].

There is literature on how to analytically or numerically solve for the optimal path [11],[12] but the optimal paths can only be computed for a small class of systems. If an approximate cost-to-go function meets several specific conditions, then it is a *feasible navigation function*. This function can be greedily descended to reach the goal state [24]. However, it is hard to find feasible navigation functions for many nonlinear and/or constrained systems.

Note also that if the cost-to-go function also satisfies the principle of optimality, it can be referred to as an optimal cost-to-go function, also known as an optimal value function in the value iteration and dynamic programming literature. The metric (in configuration space) induced by the cost of optimal paths between points, in the context of nonholonomic motion planning, is also called the nonholonomic, singular, Carnot-Caratheory, or sub-Riemannian metric [14].

Perhaps the most common pseudometric, which is in fact a metric, is that of Euclidean distance between two state space points as a function of their coordinates in that space:

$$dist(x, x') = \sqrt{\alpha \left(x_1 - x'_1\right)^2 + \ldots + \zeta \left(x_n - x'_n\right)^2} \quad (1)$$

The scaling factors can be used to encode domain knowledge about the relative significance of various state components. This metric works well on holonomic systems, in terms of coverage of the sampled space, but performs far more poorly in state space, which will be demonstrated in the section containing experimental results. It encodes no information about the constrained relationship between position and velocity. In the state space of a frictionless one-dimensional brick (double integrator) shown in Fig. 3, points A and B are equidistant from point C with respect to Euclidean distance. Yet, we know that the brick at point A is moving toward point C, while the second instance of the brick, at point B, is moving away from point C. Since the distance function determines which branch will be extended toward C, it makes intuitive sense to define distance so that A is in fact closer to C than B.

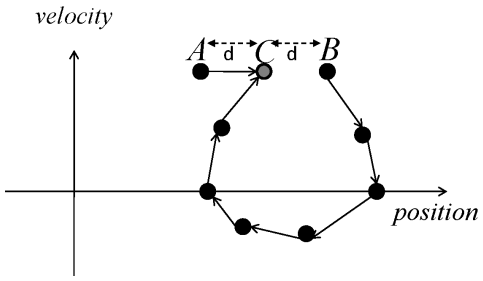


Fig. 3. Nodes A and B are equidistant to point C with respect to the Euclidean distance metric. However the children of node B, which has positive velocity, must be to the right of their parent, and therefore cannot get closer to C. In this case, Euclidean distance fails to distinguish between two nodes, one which is clearly the right choice for extending towards C, and the other which is not.

With a simple energy-based pseudometric, an RRT can find a very direct path for an underactuated simple pendulum to ascend to the unstable upright position, wasting very few tree nodes on spurious paths. The pseudometric is simply the difference in energy between two states. For energy-conservative systems in which (1) the goal lies on a set of connected states with the same energy, (2) the dynamics along that connected set bring that system to the goal state passively, and (3) no other states outside of that connected set have the same energy, this can be a very effective pseudometric.

In [18], the approximation of the ideal metric was designed using knowledge of the blimp dynamics and the underlying Lie group structure. [19] used RRTs to plan dynamic trajectories for helicopters by using cost-to-go functions from the unconstrained problem to solve for combinations of trim trajectories (maneuvers/motion primitives) in an environment with obstacles. The cost-to-go calculation was made even more tractable by exploiting the (problem-specific) symmetries and relative equilibria of helicopters, along with the construction of motion primitives. [20] uses the local first-order approximation of cost-to-go, where cost is time, and the calculation is simple enough that it only requires the Euclidean distance between state space points and evaluations of given system dynamics function.

There are two major themes that run through the published work on RRT algorithm modifications. These modifications are designed to make the RRT algorithm less sensitive to the quality of the pseudometric. The first theme is that of reducing repeated failed expansions of a node. For example, [21] collects collision information online, i.e., *constraint violation tendency* (CVT), and uses that to bias search. The high-level reasoning behind [20]'s modification is essentially the same, but the implementation uses a *history-based weighting* instead of the CVT. [13] proposes RRT-Blossum, which attempts to distinguish between types of child nodes, so that the beneficial types are added to the tree while the others are

ignored.

The second main theme in the work on RRT algorithm modification is adaptively biasing the sampling distribution towards regions of state space that are reachable by the tree's current set of nodes. [20] replaces the traditional uniform distribution of x_{rand} with a compactly supported Gaussian-based distribution centered around a point in region of interest. [25] developed a local reachability-guided sample biasing method.

III. PROBLEM FORMULATION

Following the RRT framework, we require the following components:

1) **State Space:** A $2n$-dimensional differentiable manifold, X, that denotes the state space. A state, $x \in X$, is defined as $x = (q, \dot{q})$, for $q \in C$, where C is the n-dimensional configuration space.
2) **Pseudometric:** A real-valued function, $\rho : X \times X \rightarrow [0, \infty)$, which specifies the cost of traveling from one point to another in X in accordance with a specified cost function.
3) **Boundary Value:** $x_{init} \in X$
4) **Constraint Satisfaction Detector:** A function, $D : X \rightarrow \{true, false\}$ which indicates when global constraints have been satisfied or violated.
5) **Inputs:** A set U of inputs containing all inputs that affect the state. In this work, it was a set of seven forces/torques linearly spaced between the system-specified input bounds.
6) **Equation of Motion:** The dynamics expressed as a differential equation $\dot{x} = f(x, u)$.
7) **Incremental Simulator:** A function for generating future states of the agent given the current state, the equations of motion, a time interval, and u over that time interval.

The primary objective of this work is to develop a pseudometric that increases the capability of the RRT to explore state space, compared to the standard Euclidean distance. This breaks down into two subproblems: determining the appropriate cost function for maximum state space coverage and developing an approximation of the optimal cost-to-go function that can be computed efficiently and scales well with the number of state variables. The pseudometric will be directly compared to the exact optimal cost-to-go function when the later function is known, and its impact on the ability of the RRT to explore state space will be assessed on four different dynamic systems.

IV. DESIGNING THE AQR-BASED DISTANCE HEURISTIC

Most RRTs in the existing literature, including the basic RRT algorithm we employ in this work, expand nodes forward for a fixed Δt. In order to be efficient in the number of nodes, we would like to solve a minimum-time problem from each existing node of the tree to x_{rand}, which boils downs to a minimum-time optimal control problem. Exact solutions are known for several types of systems, such as the bounded-input continuous-time double integrator and all discrete-time linear systems. Switching functions, which provide a visual cookbook for determining the time-optimal path for bounded-input continuous-time systems, potentially both nonlinear and linear, can be analytically calculated using the Pontryagin Minimum Principle, as shown in [27], when the dimension is small. For systems with three or more state variables, it becomes "generally difficult, if not impossible, to obtain an analytical expression for the switching hypersurface" [26].

Linear and quadratic programming (LP and QP) can be used to solve minimum-time optimal control problems for discrete-time systems with both bounded and unbounded inputs, but the solution only applies to getting from one specific state to another. Therefore it requires solving an LP or QP problem for every node in the tree, for each x_{rand}, which becomes increasingly time-consuming as the size of the tree grows.

Minimum-time linear quadratic regulators (LQR) are a class of optimal controllers for linear systems with quadratic cost functions of state and/or action for which the global exact closed-form cost-to-go to the specified goal state in the dynamic system's phase space can be found efficiently by numerical matrix integration. The solution applies to all nodes in the tree, for each x_{rand}. We adapt this class of controller and its associated cost-to-go function to our purposes. A more detailed explanation and derivation is included in [23]. Consider a smoothly differentiable, possibly nonlinear system:

$$\dot{x} = f(x, u), x \in \Re^n, \quad u \in \Re^m \tag{2}$$

and x_{rand}, a random sample in the state space produced by the RRT algorithm that is building a tree on this system. We linearize $f(x, u)$ at x_{rand} and a nominal input, u_f, which is always set to a zero vector in practice. We chose x_{rand} as the linearization point so that the global optimal cost-to-go can be found once rather than finding the global optimal cost-to-go to x_{rand} using the dynamics from linearizing about RRT node i and then repeating that process for each node in the tree.

Most randomly sampled points in state space are not stabilizable (e.g., any point that has a non-zero velocity). For these points, infinite-horizon LQR is not well defined, and the linearized system will in fact have affine dynamics. The derivation that follows mirrors that of an open-loop finite-horizon LQR, but has been generalized to affine systems, and for that reason, we refer to the resulting controller as an affine quadratic regulator (AQR).

For notational simplicity, we will define a new coordinate system centered about x_{rand}:

$$\bar{x} = x - x_{rand}. \tag{3}$$

The following series of equalities and approximations shows the derivation of the linearized (affine) system from the

derivative of the state:

$$\dot{\bar{x}} = \frac{d}{dt}\left(x(t) - x_{rand}\right) = \dot{x}(t) \tag{4}$$

$$\approx f(x_{rand}, u_f) + \frac{\delta f}{\delta x}\left(x(t) - x_{rand}\right) + \frac{\delta f}{\delta u}\left(u - u_f\right) \tag{5}$$

$$= c + A\bar{x} + B\bar{u}, \quad A \in \Re^{nxn}, \quad B \in \Re^{nxm} \tag{6}$$

We define the following cost function:

$$J(\bar{x}, t_0, t_f) = \int_{t_0}^{t_f} \left[1 + \frac{1}{2}\bar{u}^T(t) R \bar{u}(t)\right] dt, R = R^T > 0, \tag{7}$$

$$s.t.\ \bar{x}(t_f) = \vec{0} \tag{8}$$

$$\bar{x}(t_0) = \bar{x}_0 \tag{9}$$

$$\dot{\bar{x}} = A\bar{x} + B\bar{u} + c. \tag{10}$$

The linearized dynamics are autonomous, so the cost function can be re-parameterized with respect to time, without loss of generality, by T, the total trajectory length. T can have a significant impact on the cost-to-go from a state to x_{rand}. Consider a situation where, just by the passive linearized dynamics alone, the linearized system will arrive at x_{rand} from some state x_i in t_i seconds. A controller that drives the system from x_i to x_{rand} in some other amount of time may require a large increase in control effort.

The optimal control solution for this constrained system can be found using Pontryagin's minimum principle, which is a necessary condition for optimality. To apply this principle, we must define the Hamiltonian [11].[1]

$$H(t) = 1 + \frac{1}{2}\bar{u}^T(t) R \bar{u}(t) + \lambda(t)\left[A\bar{x} + B\bar{u} + c\right] \tag{11}$$

For the optimal solution, the Hamiltonian must be at a minimum or stationary point with respect to changes in the control function $\bar{u}(t)$ [11]. Defining L as the integrand of J, this translates to:

$$0 = \frac{\delta H}{\delta \bar{u}} = \frac{\delta L}{\delta \bar{u}} + \frac{\delta f^T}{\delta \bar{u}}\lambda(t) = R\bar{u} + B^T\lambda(t) \tag{12}$$

This "stationarity condition" and the positive definiteness of R allow us to solve for the optimal control function in terms of the Lagrange multiplier, λ:

$$\bar{u}^* = -R^{-1}B^T\lambda(t) \tag{13}$$

The time-varying Lagrange multiplier's dynamics will satisfy

$$-\dot{\lambda} = \frac{\delta H}{\delta x} = A^T\lambda(t), \quad 0 \le t \le T \tag{14}$$

so the closed-form solution for λ in terms of its final value is

$$\lambda(t) = e^{A^T(T-t)}\lambda(T) \tag{15}$$

We do not know the final value of λ. However we can define boundary conditions in terms of the system's state, in the form of a strict final boundary value condition:

$$x(T) = x_{rand} \quad (\bar{x}(T) = 0) \tag{16}$$

[1]The change in state, $\dot{\bar{x}}$, as a function of state and action is not time-varying in this case, but the definition of the Hamiltonian is general enough to handle such a problem specification [28].

The relationship between $\bar{x}$ and λ can be found by substituting the optimal control solution $\bar{u}^*$, which is in terms of λ, for $\bar{u}$ in the equation governing the dynamics of $\bar{x}$:

$$\dot{\bar{x}}(t) = A\bar{x}(t) - BR^{-1}B^T e^{A^T(T-t)}\lambda(T) + c \tag{17}$$

The relationship between the state and the Lagrange multiplier (Equ. 17) and the dynamics of the Lagrange multiplier (Equ. 14) together form what is referred to as the *Hamiltonian system*. With initial and final boundary values for the state, x_0 and x_{rand}, we can solve the Hamiltonian system for $\lambda(t)$.

We integrate Equ. 17 to get a relationship between the state, not the change in state, and the Lagrange multiplier, evaluate the relationship at T, impose the state final boundary value constraint, and substitute in the expression for λ in terms of its final value (shown earlier in Equ. 15):

$$\bar{x}(T) = e^{AT}\bar{x}_0 - \int_0^T S(T,\tau)\lambda(T) + e^{A(T-\tau)}cd\tau = 0, \tag{18}$$

$$S(t,\tau) \equiv \int_0^t e^{A(t-\tau)}BR^{-1}B^T e^{A^T(t-\tau)} \tag{19}$$

S is referred to as the *continuous reachability gramian*. The symmetry of S allows us to invert it when solving for the final value of λ. Recall that with this final value of λ we know $\bar{u}^*(t)$ and $\bar{x}(t)$ for the entire optimal trajectory. The dynamics of $P = S^{-1}$ are

$$\dot{P}(t) = AP(t) + P(t)A^T - BR^{-1}B^T. \tag{20}$$

$S(t)$ for all $t < T$ can be found by integrating the dynamics of P backwards in time from $P(T) = 0$. $S(T)$ is infinite, because $P(T) = 0$, as a result of the final state constraint that mandates an infinite cost for all trajectories that do not reach $\bar{x} = 0$ at T.

By plugging $\bar{u}^*$ into the cost function, we get the following:

$$J^*(\bar{x}, T) = T + \frac{1}{2}d^T(\bar{x}_0, T)S(T)d(\bar{x}_0, T), \tag{21}$$

$$d(\bar{x}_0, T) = e^{AT}\bar{x}_0 + \int_0^T e^{A(T-\tau)}cd\tau \tag{22}$$

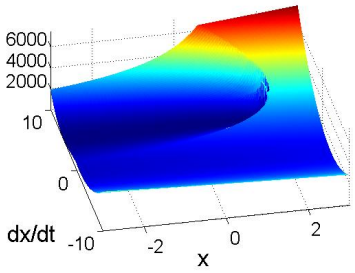
Finally, since the cost of traveling along the optimal trajectory from $\bar{x}_0$ to the origin of our state coordinate system in T units of time is highly dependent on T, we search for the horizon time with the least cost

$$T^* = \text{argmin}_T J^*(\bar{x}_0, T), \quad 0 \le T \le T_{max} \tag{23}$$

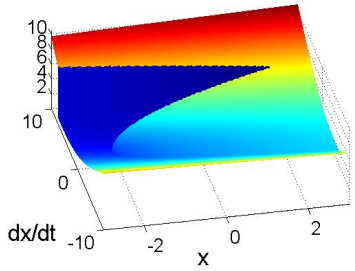
$J^*(\bar{x}_0, T^*)$ is the distance when traveling from x_0 to x_{rand} according to the AQR-based pseudometric, which has been visualized for a bounded-input continuous-time double integrator in Fig. 4. The AQR-based pseudometric captures many of the features of the exact solution that the Euclidean distance cannot.

The user must choose some maximum T by considering the possibility of the lowest $J^*(\bar{x}, T)$ for a given $\bar{x}$ occurring

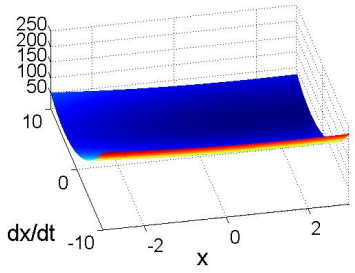
at a longer T and the additional computation time of finding $J^*(\bar{x}, T)$ for longer T. The additional computation of considering longer T is made more efficient by observing that both S and d can be solved recursively by integrating backwards from the final conditions. See [23] for implementation details.



(a) AQR-Based Distance



(b) Time of Exact Minimum-Time Trajectory



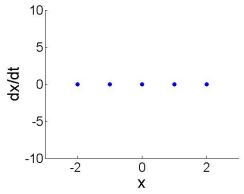
(c) Euclidean Distance

Fig. 4. Maps of distance to a sample point, [2,5], in the state space of the bounded-input continuous-time double integrator. The distance magnitudes in each subfigure are different, but the relevant feature here is how well the shape of the distance landscape of one pseudometric compares to another. The AQR-based pseudometric approximates the exact solution, which is known for this system.

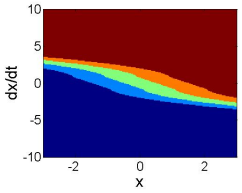
A. Voronoi Diagrams

Every new sampled point (x_{rand}) is mapped back to the nearest node to it in the RRT (x_{near}) so that x_{near} can be expanded towards x_{rand}. Since the sampled space is sampled uniformly with respect to the Euclidean distance, the probability of expanding node i in the RRT is proportional to the Euclidean area of the Voronoi region of node i, where the Voronoi region contains all points closer to node i than any other RRT node, in terms of the distance pseudometric used by the NEAREST_NEIGHBOR function. This is referred to as the Voronoi bias, the bias RRTs have towards exploring places not yet visited. Regions on the frontier of the tree and regions where little exploration has occurred contain the fewest nodes; the farther apart the nodes are in a region, the larger their Voronoi regions and the more likely they are to be nearest of all RRT nodes to x_{rand} and expanded.

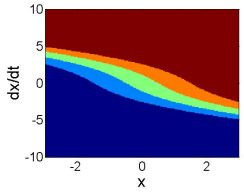
By determining the Voronoi regions, the distance pseudometric also determines what regions of the sampled space are least explored, and which children of an expanded x_{near} have brought the RRT closer to that less explored space, which directly affects how well RRT's exploration of configuration space on holonomic systems can be replicated within the state space of dynamic (nonholonomic) systems. Fig. 5 shows a side-by-side comparison of the Voronoi regions as a function of the various distance pseudometrics known for the bounded-input continuous-time double integrator. The regions based on the AQR-based pseudometric approximate those of the exact solution well.



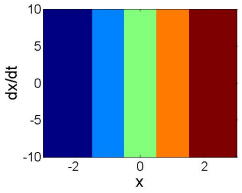
(a) The Five-Node Tree



(b) AQR-Based Distance



(c) Exact Min-Time



(d) Euclidean Distance

Fig. 5. Voronoi diagrams for a 5-node toy RRT on the bounded-input continuous-time double integrator.

V. EXPERIMENTS

Since we developed our pseudometric with a specific, measurable performance index in mind, i.e., state space coverage, the design of experiments was straightforward. Note that [20] also used this technique to quantitatively compare planning algorithms' effectiveness. For each dynamic system, we build two RRTs. The parameters of the system and the RRT-building algorithm are held constant, except for the distance pseudometric, so that any resulting differences in state space coverage between the trees can be attributed solely to the choice of pseudometric. The two trees do not receive identical random x_{rand} to grow towards, and due to this inherent randomness, the reported results are the averages of repeated trials. While RRTs can be biased to grow towards a particular point, these RRTs have no goal-bias, since we are interested in creating trees whose branches reach out into all regions of the sampled state space.

The RRT algorithm attempts to grow the tree towards random samples (x_{rand}) which are taken from some finite-volume subset of the infinitely large phase space. This

sampled space was divided into bins. The percentage of populated bins is our measure of coverage. For two-dimensional state spaces, a 10x10 grid of identically sized bins was used. For four-dimensional state spaces, a 6x6x6x6 grid of identically sized bins was used. (That amounts to 1296 bins in total to cover the space.)

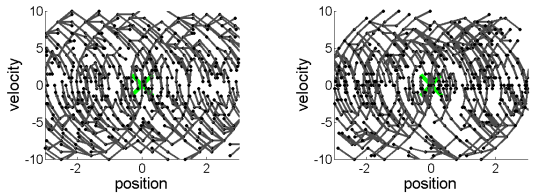
RRTs were grown on four different dynamic systems: the double integrator, pendulum, cart-pole, and Acrobot. All four systems have bounds on the force/torque that can be applied. The double integrator is equivalent to a brick that moves along a single dimension without friction. The pendulum is a point-mass on a massless rod attached to an actuated pivot point, and is also undamped. The cart-pole is the same classic system that control textbooks address. It is a pendulum attached to a brick, with no actuation at the pivot point of the pendulum, driven entirely by forces applied to the brick. The Acrobot is a double (two-link) pendulum with actuation only at the joint between the two links. All trees were grown from a root node where the system is at its stable equilibrium. The green "X" indicates the location of the tree root (x_{init}), and the axes of the plot are set such that only the sampled region of state space (the region from which x_{rand} is uniformly, randomly drawn) is visible.

The RRT algorithm's, dynamic systems' and the pseudometrics' settings may all have a significant effect on the results. In the Euclidean distance metric, the squared differences between two states along each axis are equally weighted. The AQR-based pseudometric has two parameters: the maximum considered horizon for the finite-time AQR optimal control problem and R, the penalty factor for applying force/torque in the AQR cost function. For these experiments, the maximum considered finite horizon length was arbitrarily set to 5 seconds. However, R, the penalty factor for applying force/torque, was varied in order to find the value which produced the greatest coverage.

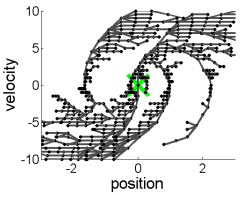
A. Double Integrator

The double integrator system is unique among the four dynamic systems considered, and for two reasons. First, ignoring the bounds on the force that can be applied, its dynamics are linear. There is no need for linearization in order to apply the AQR-based pseudometric, which eliminates one source of error. Second, the true minimum-time trajectory between any two points is known; the corresponding minimum-time pseudometric returns the length, in time, of the node that can reach a given state in the least amount of time. The AQR-based pseudometric is intended to approximate the true minimum-time pseudometric. We have already compared maps of the distances they assign to a mesh of points around a given goal. In this section, we can see how well the AQR-based pseudometric approximates the true minimum-time pseudometric in terms of state space coverage.

The axes of the figures in Fig. 6 are set such that only sampled space from which x_{rand} is randomly uniformly drawn is visible, and it is perhaps most readily apparent that the exact minimum-time pseudometric leads to a very uniform coverage of the entire sampled space. We can



(a) Using the Exact Minimum-Time Pseudometric (b) Using AQR-Based Distance



(c) Using Euclidean Distance

Fig. 6. Examples of 1000-node RRTs grown on a double integrator with bounds on the input applied to the system.

also see that coverage of the RRT using the AQR-based approximation of the minimum-time pseudometric reaches almost as much of the sampled space. Finally, we can see that the Euclidean-based RRT is not able to explore the upper left and lower right quadrants of state space as well as the RRTs using the other pseudometrics. This can be explained by the fact that many of the x_{rand} in the upper left and lower right quadrants were closest in Euclidean distance to the branches representing states where the system is constrained by its dynamics to continue moving away, not toward, that x_{rand}.

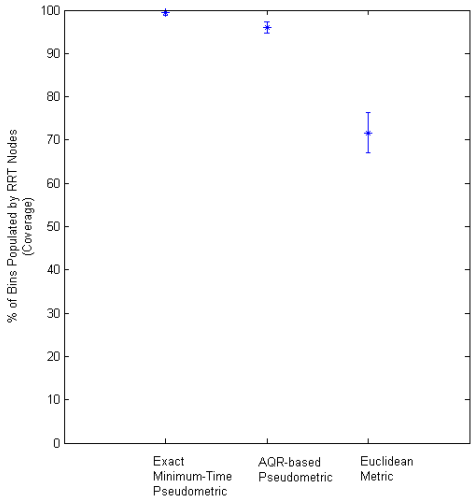
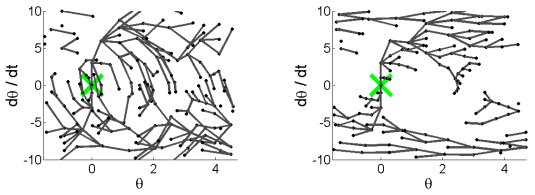


Fig. 7. Comparative coverage of 1000-node RRTs' exploration of the double integrator's state space. Fifty RRTs were grown for each distance pseudometric, and the mean and standard deviation of those RRTs' coverage is shown.

B. Pendulum

In Fig. 8, it is clear that the RRT with the AQR-based pseudometric is able to reach a greater percentage of the state space than the RRT with the Euclidean metric. This



(a) Using AQR-Based Distance (b) Using Euclidean Distance

Fig. 8. 200-node RRTs grown on a torque-limited pendulum.

comparison holds true over repeated trials, as shown in Fig. 9.

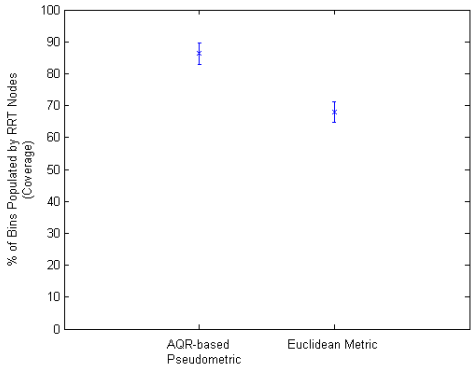


Fig. 9. Comparative coverage of 200-node RRTs' exploration of the pendulum's state space. Fifty RRTs were grown for each distance pseudometric, and the mean and standard deviation of those RRTs' coverage is shown.

C. Cart-pole

In Fig. 10, there is no clear difference between the coverage of the RRTs using the two different distance pseudometrics. Fig. 11 confirms this lack of differentiation.

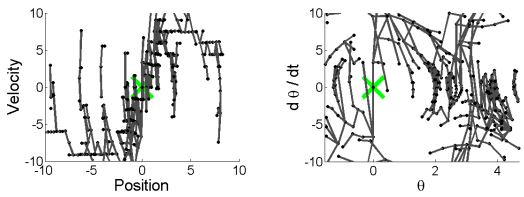
Tripling the size of the trees does not change the relative performance of the RRTs using these two pseudometrics.

D. Acrobot

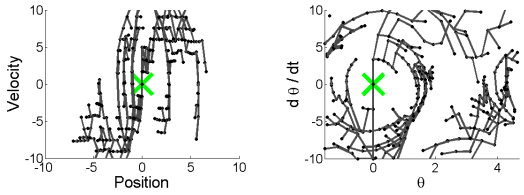
In Fig. 12, there appears to be some advantage to using the AQR-based distance pseudometric, in terms of coverage. However, it is not possible to know this from the figure because information is lost when the four-dimensional space is projected onto two two-dimensional graphs. Fig. 13 shows that on average, there is no significant difference in coverage.

VI. DISCUSSION

It is impossible to make sweeping statements about the value of the AQR-based pseudometric based on just four dynamic systems, but trends can be discussed, and perhaps verified in future work. There appears to be a negative correlation between the complexity/nonlinearity of a system's dynamics and the benefit of using the AQR-based distance pseudometric over the Euclidean distance. This makes intuitive sense since, for systems with more complex, nonlinear dynamics, the accuracy of the cost-to-go (distance) estimates of the AQR-based pseudometric will degrade faster



(a) Using AQR-Based Distance



(c) Using Euclidean Distance

Fig. 10. 500-node RRTs grown on a force-limited cart-pole.

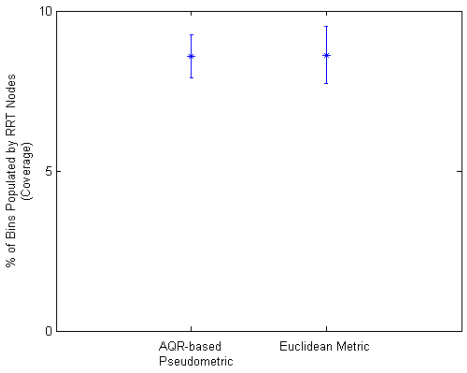
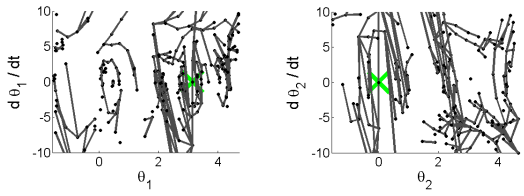
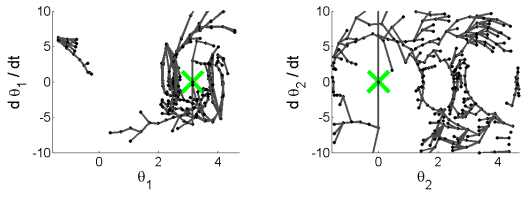


Fig. 11. Comparative coverage of 500-node RRTs' exploration of the cart-pole's state space. Ten RRTs were grown for each distance pseudometric, and the mean and standard deviation of those RRTs' coverage is shown.



(a) Using AQR-Based Distance



(c) Using Euclidean Distance

Fig. 12. 500-node RRTs grown on a torque-limited Acrobot.

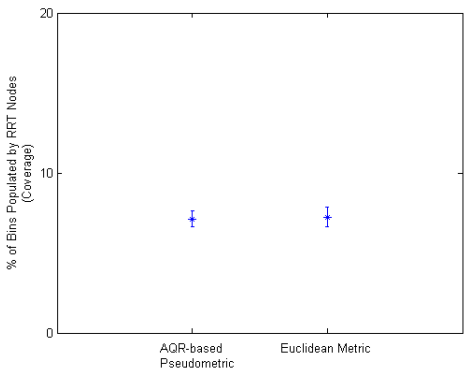


Fig. 13. Comparative coverage of 500-node RRTs' exploration of the Acrobot's state space. Ten RRTs were grown for each distance pseudometric, and the mean and standard deviation of those RRTs' coverage is shown.

as a function of distance to the linearization point. A more subtle possible trend is that the coverage of any given RRT grown using the AQR-based pseudometric is more consistent than when using the Euclidean metric. The standard deviation of coverage across repeated trials was the same or less.

Since the AQR-based pseudometric and the Euclidean metric are both equally ignorant of obstacles, tests which include obstacles are not planned. Since we observe a drop-off in benefit as system complexity and nonlinearity increase, future work could include exploring methods for approximating the exact minimum-time distance pseudometric which can reason about dynamics with higher-order terms.

The results presented in this work are focused solely on quantifying the impact of the AQR-based pseudometric on RRTs' coverage of state space. However, there may be other advantages to using this pseudometric as well. AQR's cost function allows the user to bias the RRT towards solutions which require low input energy. Growing an RRT with this bias may not lead to the greatest coverage, but may in of itself be of interest to the research community. The AQR-based pseudometric can also be used in other algorithms that depend on a notion of distance in state space, such as probabilistic roadmaps.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the contributions of David Wingate, Rick Cory, Alec Shkolnik, Ian Manchester, John Roberts, and Martin Glassman. We also appreciate the thoughtful feedback of Seth Teller and Abraham Bachrach.

REFERENCES

[1] J. H. Reif. Complexity of the movers problem and generalizations. In *Proceedings IEEE Symposium on Foundations of Computer Science*, pages 421-427, 1979.

[2] N. M. Amato and Y. Wu. A randomized roadmap method for path and manipulation planning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 113–120. IEEE, 1996.

[3] S. LaValle. Rapidly-exploring random trees: A new tool for path planning. Technical Report 98-11, Iowa State University, Dept. of Computer Science, 1998.

[4] S. M. LaValle and J. J. Kuffner, Jr. Randomized kinodynamic planning. *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1:473–479, 1999.

[5] S. LaValle and J. Kuffner. Rapidly-exploring random trees: Progress and prospects. In *Proceedings of the Workshop on the Algorithmic Foundations of Robotics*, 2000.

[6] Steven M. LaValle and James J. Kuffner, Jr. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(5):378–400, 2001.

[7] A. Yershova, L. Jaillet, T. Simeon, and S. M. LaValle. Dynamic-domain RRTs: Efficient exploration by controlling the sampling domain. *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 3856–3861, April 2005.

[8] Steven M. Lavalle. From dynamic programming to RRTs: Algorithmic design of feasible trajectories. In *Control Problems in Robotics*. Springer-Verlag, 2002.

[9] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 2nd edition, 2000.

[10] Peng Cheng and S. M. LaValle. Reducing metric sensitivity in randomized trajectory design. *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, 1:43–48, 2001.

[11] Frank L. Lewis. *Applied Optimal Control and Estimation*. Digital Signal Processing Series. Prentice Hall and Texas Instruments, 1992.

[12] John Canny, Ashutosh Rege, and John Reif. An exact algorithm for kinodynamic planning in the plane. In *SCG '90: Proceedings of the Sixth Annual Symposium on Computational Geometry*, pages 271–280. ACM, 1990.

[13] Maciej Kalisiak. *Toward More Efficient Motion Planning with Differential Constraints*. Ph.D. thesis, University of Toronto, 2008.

[14] J. P. Laumond, S. Sekhavat, and F. Lamiraux. Guidelines in nonholonomic motion planning for mobile robots. In J.-P. Laumond, editor, *Robot Motion Planning and Control*, pages 1-53. Springer-Verlag, Berlin, 1998.

[15] S. Sundar and Z. Shiller. Optimal obstacle avoidance based on the Hamilton-Jacobi-Bellman equation. *IEEE Trans. Robot. & Autom.*, 13(2):305-310, April 1997.

[16] T. Basar and G. J. Olsder. *Dynamic Noncooperative Game Theory*. Academic Press, London, 1982.

[17] S. M. LaValle. *A Game-Theoretic Framework for Robot Motion Planning*. Ph.D. thesis, University of Illinois, Urbana, IL, July 1995.

[18] Jongwoo Kim, Jim Keller, and R. Vijay Kumar. Design and verification of controllers for airships. In *Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, volume 1, pages 54–60, 2003.

[19] Emilio Frazzoli. *Robust Hybrid Control for Autonomous Vehicle Motion Planning*. Ph.D. thesis, Massachusetts Institute of Technology, June 2001.

[20] Jongwoo Kim, Joel M. Esposito, and Vijay Kumar. An RRT-based algorithm for testing and validating multi-robot controllers. In *Robotics: Science and Systems I*, June 2005.

[21] Peng Cheng. *Sampling-based motion planning with differential constraints*. Ph.D. thesis, 2005. Adviser-Steven M. Lavalle.

[22] Russ Tedrake. LQR-Trees: Feedback motion planning on sparse randomized trees. In *Proceedings of Robotics: Science and Systems (RSS)*, page 8, 2009.

[23] Elena Leah Glassman. A quadratic regulator-based heuristic for rapidly exploring state space. Master's thesis, Massachusetts Institute of Technology, February 2010.

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

[25] Alexander Shkolnik, Matthew Walter, and Russ Tedrake. Reachability-guided sampling for planning under differential constraints. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pages 2859–2865. IEEE/RAS, 2009.

[26] Donald E. Kirk. *Optimal Control Theory: an Introduction*. Dover Publications, 2004.

[27] Dimitri P. Bertsekas. *Dynamic Programming and Optimal Control*, volumes I and II. Athena Scientific, 3rd edition, May 2005.

[28] Russ Tedrake. Underactuated Robotics: Learning, Planning, and Control for Efficient and Agile Machines: Course Notes for MIT 6.832. Working draft edition, 2009.