# Lyapunov Design for Safe Reinforcement Learning Control

**Theodore J. Perkins and Andrew G. Barto**
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003, USA
{perkins,barto}@cs.umass.edu

**Abstract**

We propose a general approach to safe reinforcement learning control based on Lyapunov design methods. In our approach, a Lyapunov function—a special form of domain knowledge—is used to formulate the action choices available to a reinforcement learning agent. A learning agent choosing among these actions provably enjoys performance guarantees, and satisfies safety constraints of various kinds. We demonstrate the general approach by applying it to several illustrative pendulum control problems.

## Introduction

Recently, practitioners of artificial intelligence have paid increasing attention to issues of safety, particularly for learning systems which may come to behave in ways not expected by the system designer (Singh *et al.* 1994; Weld and Etzioni 1994; Schneider 1997; Neuneier and Mihatsch 1999; Gordon 2000; Perkins and Barto 2001b; Barto and Perkins 2001). In problems where safety is not a concern or is easily achieved, reinforcement learning (RL) techniques have generated impressive solutions to difficult, large-scale control problems. Examples include such diverse problems as playing backgammon (Tesauro 1994), elevator dispatching (Crites and Barto 1998), option pricing (Tsitsiklis and Van Roy 1999), and job-shop scheduling (Zhang and Dietterich 1996). However, there are obstacles to applying RL to problems where it is important to ensure reasonable system performance and/or respect safety constraints at all times. Most RL algorithms offer no guarantees on the quality of control during learning, which can be costly if learning is done on-line. Further, although one does not expect a strictly optimal solution from an RL algorithm, a solution produced by an RL algorithm may not even share basic, qualitative properties of an optimal solution. For example, an optimal solution to a stochastic shortest path problem brings any initial system state to a final, goal state. However, an intermediate or even final solution produced by an RL algorithm may *not* bring every initial state to a goal state.

Safe control design has always been a concern in control engineering, where one of the fundamental theoretical tools employed is Lyapunov analysis (Vincent and Grantham 1997). Lyapunov analysis alone does not generally solve an optimal control problem. However, Lyapunov methods are used quite successfully for verifying qualitative properties of control designs, such as stability or limiting behavior. In this paper, we investigate the use of Lyapunov-based methods to ensure that an RL agent's behavior satisfies qualitative properties relating to goal-achievement and safety. The precise properties we seek to establish are described in the next section.

We assume no knowledge of the internals of the RL agent. It is modeled as a black box that non-deterministically selects control actions in a manner unknown to us. The advantage of this assumption is generality. Our approach is consistent with virtually any choice of RL algorithm and applies equally well during and after learning. Since we avoid assumptions regarding the internals of the agent, we must establish safety guarantees using "external" factors independent of the agent's behavior. The general approach we propose is to use domain knowledge in the form of a Lyapunov function to design the action choices available to the agent. An appropriately designed set of actions restricts the agent's behavior so that, regardless of precisely which actions it chooses, desirable performance and safety objectives are guaranteed to be satisfied.

## Markov Decision Processes and Qualitative Properties

We model the agent's environment as a Markov decision process (MDP), evolving on a state set $S$. The environment starts in state $s_0$, which is determined according to a start state distribution $S_0$. At the times $t = 0, 1, 2, \ldots$ the agent chooses an action, $a_t$, from a set of allowed actions, $A(s_t)$. The immediate cost incurred, $c_t$, and the next state of the environment, $s_{t+1}$, are determined stochastically according to a joint distribution depending on the state-action pair $(s_t, a_t)$. Some MDPs have goal states which correspond to a termination of the control problem. If a problem has goal states, we assume they are modeled as absorbing states—i.e., only one action is available, which incurs no cost and leaves the state of the MDP unchanged. The set of goal states is denoted $G$. If there are no goal states, then $G = \emptyset$.

A (stochastic) policy $\pi$ maps each state $s \in S$ to a distribution over the available actions, $A(s)$. The expected discounted return of policy $\pi$ is defined as:

$$V^\pi = \lim_{\tau\to\infty} E\left\{\sum_{t=0}^{\tau} \gamma^t c_t\right\},$$

where the expectation is with respect to the start state distribution, the stochastic transitions of the environment, and the stochastic actions selections of the policy. The factor $\gamma$ is a discount rate in the range [0,1]. The RL agent's task is to find a policy that minimizes expected discounted return.

Now, suppose some controller acts on an MDP, resulting in a sequence of states $\{s_0, s_1, s_2, \ldots\}$. Such a sequence is called a trajectory, and there are many qualitative questions we may ask concerning the trajectories that may occur. Let $T \subset S$ be a set of states. We define a number of properties that may be of interest:

**Property 1 (Reach)** *The agent brings the environment to set $T$ (for certain/with probability 1). That is, there exists $t \geq 0$ such that $s_t \in T$ (for certain/with probability 1).*

Sometimes we want to know whether the agent keeps the MDP in a subset of state space if it starts there.

**Property 2 (Remain)** *For all $t \in \{0, 1, 2, \ldots\}$, $s_t \in T$ (for certain/with probability 1).*

Combining the two previous properties is the idea that the agent is guaranteed to bring the MDP to a part of state space and keep it there.

**Property 3 (Reach and remain)** *There exists $\tau \in \{0, 1, 2, \ldots\}$ such that for all $t \geq \tau$, $s_t \in T$ (for certain/with probability 1).*

For some MDPs it is impossible to maintain the state in a given set $T$. A lesser requirement is that the MDP spend an infinite amount of time in $T$—always returning to $T$ if it leaves. We only have need for the probability 1 version of this property.

**Property 4 (Infinite time spent)** *With probability 1, for infinitely many $t \in \{0, 1, 2, \ldots\}$, $s_t \in T$.*

Questions such as these are often addressed in the model checking literature. Indeed, Gordon (2000) has demonstrated the relevance of model checking techniques for verifying learning systems. Whether model checking techniques, which are most successful for finite-state systems, are relevant for the infinite-/continuous-state problems we are working on is uncertain.

The final property we describe is convergence, or stability. For this we define a distance-to-$T$ function $\delta_T : S \to \Re^+$. At the least, this function should satisfy $\delta_T(s) = 0$ for $s \in T$ and $\delta_T(s) > 0$ for $s \notin T$.

**Property 5 (Asymptotic approach)** *(For certain/with probability 1), $\lim_{t\to\infty} \delta_T(s_t) = 0$.*

This last property is often studied in control theory, typically taking $T$ to contain a single state. Our situation differs from the usual control theory setting because of the presence of the learning agent acting on the system. Nevertheless, we are able to use control-theoretic techniques to establish the asymptotic approach and other properties.

## Lyapunov Theory

Lyapunov methods originated in the study of the stability of systems of differential equations. The central task is to identify a *Lyapunov function* for the system—often thought of as a generalized energy function. Showing that the system continuously dissipates this energy (i.e., that the time derivative of the Lyapunov function is negative) until the system reaches a point of minimal energy establishes that the system is stable. Sometimes a Lyapunov function may correspond to some real, physical notion of energy. For example, by analyzing how the mechanical energy of a frictionally-damped pendulum system changes over time, one can establish that the pendulum will always asymptotically approach a stationary, downward-hanging position (Vincent and Grantham 1997). Sometimes the Lyapunov function is more abstract, as in, for example, the position error functions used to analyze point-to-point or trajectory-tracking control of robot arms (see, e.g., Craig 1989).

Lyapunov methods are used in control theory to validate a given controller and to guide the design process. The vast majority of the applications are still for systems of (control) differential equations. However, the basic idea of Lyapunov has been extended to quite general settings, including environments with general state sets evolving in discrete time (Meyn and Tweedie 1993). We must further extend these approaches because of the RL agent in the control loop, which non-deterministically drives the environment. Let $T \subset S$ and let $L : S \to \Re$ be a function that is positive on $T^c = S - T$. We use $s'$ to denote a state resulting when the system is in state $s$ and an action $a \in A(s)$ is taken. Let $\Delta > 0$ and $p > 0$ be fixed real numbers.

**Theorem 1** *If for all states $s \notin T$ and all actions $a \in A(s)$, $s' \in T$ or $L(s) - L(s') \geq \Delta$ holds (for certain/with probability 1) then the environment enters $T$ (for certain/with probability 1). For start state $s_0$, this happens at time no later than $t = \lceil L(s_0)/\Delta \rceil$ (for certain/with probability 1).*

To paraphrase, if on every time step the environment either enters $T$ or ends up in a state at least $\Delta$ lower on the Lyapunov function $L$, regardless of the RL agent's choice of action, then the environment is bound to enter $T$.

**Proof sketch:** (for the "for certain" case) Suppose the environment starts in state $s_0$ and the agent takes a sequence of $k = \lceil L(s_0)/\Delta \rceil$ actions without the environment entering $T$. Then at each successive time step, the environment must enter a state at least $\Delta$ lower on $L$ than at the previous time step. After $k$ time steps, we must have $L(s_k) \leq L(s_0) - k\Delta = L(s_0) - \lceil L(s_0)/\Delta \rceil \Delta \leq L(s_0) - L(s_0) = 0$. But $L$ is positive for all states not in $T$, so $s_k \in T$, contradicting the assumption that the environment does not enter $T$. $\Box$

When descent cannot be guaranteed, having some chance of descent is sufficient as long as $L$ can be bounded above:

**Theorem 2** *If $\sup_{s \notin T} L(s) = U \in \Re$, and if for all states $s \notin T$ and actions $a \in A(s)$, with probability at least $p$, $s' \in T$ or $L(s) - L(s') \geq \Delta$, then with probability 1 the environment enters $T$.*

**Proof sketch:** Let $k = \lceil U/\Delta \rceil$. Since $0 < L(s) \leq U$ for all $s \notin T$, there is at least probability $p^k$ that the environment enters $T$ during any block of $k$ consecutive time steps,

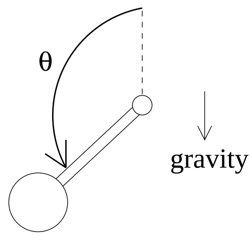


Figure 1: A single link pendulum.

regardless of the state of the environment at the start of that time. This is because there is at least probability $p$ of a descent of $\Delta$ or more on each time step, and $k$ such descents ensure entering $T$ from $s \notin T$. The probability that the environment does not enter $T$ during the first $k$ steps is no more than $1 - p^k$. The probability that the environment does not enter $T$ during the first $2k$ time steps is no more than $(1 - p^k)^2$, and so on. The probability that the environment never enters $T$ is no more than $\lim_{j\to\infty}(1 - p^k)^j = 0$. □

The basic approach we propose is to identify a candidate Lyapunov function for a given control problem and then *design* or *restrict* the action choices of the RL agent so that one of the theorems above applies. In turn, these theorems allow us to establish the properties described in the previous section. In the remainder of the paper we study three pendulum control problems and demonstrate the use of Lyapunov functions in designing actions and establishing these properties with respect to relevant subsets of the state space.

## Deterministic Pendulum Swing-Up and Balance

Pendulum control problems have been a mainstay of control research for many years, in part because pendulum dynamics are simply stated yet highly non-linear. Many researchers have discussed the role of mechanical energy in controlling pendulum-type systems (see, e.g., Spong 1995; Boone 1997a, 1997b; DeJong 2000; and Perkins and Barto 2001a, 2001b). The standard tasks are either to swing the pendulum's end point above some height (swing-up) or to swing the pendulum up and balance it in a vertical position (swing-up and balance). In either task, the goal states have greater total mechanical energy than the start state, which is typically the hanging-down, rest position. Thus, any controller that solves one of these tasks must somehow impart a certain amount of energy to the system. We first demonstrate our approach on a deterministic pendulum swing-up and balance task.

### Dynamics and Controllers

Figure 1 depicts the pendulum. The state of the pendulum is specified by an angular position, $\theta$, and an angular velocity, $\dot\theta$. The angular acceleration of the pendulum is given by

$$\ddot\theta = \sin\theta + u \ ,$$

where the sine term is due to gravity and $u$ is a control torque. This equation assumes the pendulum is of length one, mass one, and gravity is of unit strength. Taking advantage of symmetry, we chose to normalize $\theta$ to the range $[-\pi, \pi]$. We also artificially restricted $\dot\theta$ to the range $[-6, 6]$. This was important only for the worst-behaved, naive RL experiments, in which agents sometimes developed control policies that drove the pendulum's velocity to $\pm\infty$. We assume the control torque is bounded in absolute value by $u_{max} = 0.2250$. With this torque limit, the pendulum cannot be directly driven to the balanced position. Instead, the solution involves swinging the pendulum back and forth repeatedly, pumping energy into it until there is enough energy to swing upright. Note that this coordinate system puts the upright, zero velocity state at the origin.

Most RL algorithms are designed for discrete time control. To bridge the gap between the RL algorithms and the continuous time dynamics of the pendulum, we define a number of continuous time control laws for the pendulum. Each control law chooses a control torque based on the state of the pendulum. The discrete time actions of the RL agent in our experiments correspond to committing control of the pendulum to a specific control law for a specific period of time. Discrete time control of continuous systems is often done in this manner (for other examples, see, e.g., Huber and Grupen 1998; Branicky et al. 2000). We use two constant-torque control laws:

**CL1**$(\theta, \dot\theta) = +u_{max}$,

**CL2**$(\theta, \dot\theta) = -u_{max}$.

A third control law is a saturating linear quadratic regulator (LQR) design for a linear approximation of the pendulum dynamics centered at the origin. See Vincent and Grantham (1997) for details. For a region of state space surrounding the origin, this controller is sufficient to pull the pendulum in to the origin and hold it there.

$$\textbf{CL3}(\theta,\dot\theta) = \max(-u_{max}, \min(u_{max}, -2.4142\,\theta - 2.1974\,\dot\theta))$$

In other words, CL3 is the function $-2.4142\,\theta - 2.1972\,\dot\theta$ clipped to remain in the range $[-u_{max}, +u_{max}]$. The constants 2.4142 and 2.1972 come from numerical solution of a matrix equation that is part of the LQR design process; they have no special meaning by themselves.

None of these control laws rely on Lyapunov analysis, and none alone can bring the pendulum to an upright, balanced state (the origin) from all initial states. We now develop two controllers based on a Lyapunov analysis of the pendulum.

The mechanical energy of the pendulum is $\mathrm{ME}(\theta, \dot\theta) = 1 + \cos(\theta) + \frac{1}{2}\dot\theta^2$. At the origin, the mechanical energy of the pendulum is precisely 2. For any other state with $\mathrm{ME} = 2$, if $u$ is taken to be zero, the pendulum will naturally swing up and asymptotically approach the origin. So, the pendulum swing-up and balance problem can be reduced to the problem of achieving any state with a mechanical energy of 2.

The time derivative of the pendulum's mechanical energy is $\dot{\mathrm{ME}}(\theta, \dot\theta) = -\sin(\theta)\,\dot\theta + \dot\theta\,\ddot\theta = \dot\theta\,u$. So a natural choice of controller to increase the pendulum's energy to 2 from some initial state would be to choose $u$ of magnitude $u_{max}$

and with sign matching $\dot{\theta}$. It turns out that this choice has potential difficulties where the control torque of $\pm u_{max}$ can be at equilibrium with the effects of gravity (Perkins 2002). However, a modification of this rule can avoid the equilibrium point problem. We call this strategy MEA for "modified energy ascent." Parameterized by $w$, this rule supplies a control torque of magnitude $w$ in the direction of $\dot{\theta}$ "most of the time." If that choice would be dangerously close to equilibrium with gravity, though, it switches to lower torque of magnitude $w/2$.

$$\text{MEA}(w,\theta,\dot{\theta}) = \begin{cases} \text{sgn}(\dot{\theta})w & \text{if } |\dot{\theta}| > \dot{\epsilon} \\ & \text{or } (0 < \dot{\theta} < \dot{\epsilon} \text{ and } \theta \notin P1) \\ & \text{or } (-\dot{\epsilon} < \dot{\theta} < 0 \text{ and } \theta \notin P2) \\ \frac{1}{2}\text{sgn}(\dot{\theta})w & \text{if } (0 < \dot{\theta} < \dot{\epsilon} \text{ and } \theta \in P1) \\ & \text{or } (-\dot{\epsilon} < \dot{\theta} < 0 \text{ and } \theta \in P2) \\ \text{sgn}(\theta)w & \text{if } \dot{\theta} = 0. \end{cases}$$

where $\text{sgn}(x) = \{+1 \text{ if } x \geq 0 \text{ and } -1 \text{ if } x < 0\}$, $\dot{\epsilon} > 0$ is a constant, and $P1$ and $P2$ are sets of states surrounding equilibrium points of $w$ with gravity. In particular, letting $\theta_w = \arcsin(w)$, then $P1 = [-\pi + \theta_w - \epsilon, -\pi + \theta_w + \epsilon) \cup [-\theta_w - \epsilon, -\theta_w + \epsilon)$ and $P2 = (\theta_w - \epsilon, \theta_w + \epsilon] \cup (\pi - \theta_w - \epsilon, \pi - \theta_w + \epsilon]$. We take $\dot{\epsilon} = 0.1$ and $\epsilon = \frac{1}{2}(\arcsin(w) - \arcsin(\frac{1}{2}w))$. We use MEA to define two final control laws:

**CL4**$(\theta,\dot{\theta}) = \text{MEA}(u_{max},\theta,\dot{\theta})$,

**CL5**$(\theta,\dot{\theta}) = \text{MEA}(\frac{1}{2}u_{max},\theta,\dot{\theta})$.

**Theorem 3** *(Perkins 2002) If the pendulum is controlled continuously by CL4 or CL5 from any initial state, then the mechanical energy of the pendulum increases monotonically and without bound. Further, for any initial state, if the pendulum is controlled by either CL4 or CL5 for a period of time $\delta$, the mechanical energy of the pendulum will increase by some minimal amount $\Delta$. The amount $\Delta$ differs for CL4 and CL5 and depends on $\delta$, but it does not depend on the initial state of the pendulum.*

The pendulum can be brought to the upright balanced position by using either CL4 or CL5 to increase the pendulum's energy to 2, and then letting $u = 0$ as the pendulum swings up and asymptotically approaches the origin. But is this the optimal strategy? In the next section we formulate a minimum-time swing-up and balance task, and we find that neither CL4 nor CL5 alone produce time-optimal control. An RL controller that learns to switch between the two can produce faster swing-up. Allowing switching among other controllers enables even faster swing-up.

## Experiments

We performed four learning experiments in the deterministic pendulum domain. In each experiment, the RL agent had different actions to choose from—i.e., different sets of control laws with which it could control the pendulum. In all cases, the task was to get the pendulum to a small set of goal states near the origin, $G1 = \{(\theta,\dot{\theta}) : \|(\theta,\dot{\theta})\|_2 \leq 0.01\}$, in minimum time. However, in the first three experiments, we used the fact that from any state with mechanical energy 2, taking $u = 0$ uniformly results in a trajectory asymptotically approaching the origin (and thus entering $G1$). In the first three experiments we therefore defined a surrogate goal set $G2 = \{(\theta,\dot{\theta}) : \text{ME}(\theta,\dot{\theta}) = 2\}$.

In all experiments we used the Sarsa($\lambda$) algorithm with $\lambda = 0.8$ to learn control policies. The action value functions were represented using separate CMAC function approximators for each action. Each CMAC covered the range of states $-\pi \leq \theta \leq \pi$ and $-6 \leq \dot{\theta} \leq 6$. Each CMAC had 10 layers, and each layer divided both dimensions into 24 bins, for a total of 576 tiles per layer. Offsets were random. The learning rate for the $k^{th}$ update of a tile's weight was $1/\sqrt{k}$. See Sutton and Barto (1998) for details and references on Sarsa and CMACs. We performed 25 independent runs of 20,000 learning trials in which the RL agents chose actions randomly with probability 0.1 and otherwise chose an action with maximal estimated action value; ties were broken randomly. After each learning trial we performed a test trial, in which there was no learning and no exploration, to evaluate the current policy. All trials started from state $(\theta,\dot{\theta}) = (\pi, 0)$. Trials were terminated if they exceeded 900 time steps, which is approximately 50 times the duration required to achieve $G1$ in any of the formulations.

In the first experiment, the agent had two actions to choose from, corresponding to control laws CL4 and CL5. When an agent chooses an action, it means that the pendulum is controlled according to the corresponding control law for 1 second or until the pendulum enters the goal set, whichever happens first. For actions that do not directly result in a goal state, the cost is 1. For actions that cause the pendulum to enter $G2$ the cost is the time from the start of the action until the pendulum reaches $G2$ plus a "terminal" cost equal to the time it takes the pendulum to free swing (under $u = 0$) to the set $G1$. Thus, an optimal policy reflects minimum time control to the set $G1$. By Theorem 3, both of the actions for the agent in experiment 1 either cause the pendulum to enter $G2$ or increase the mechanical energy by at least some amount $\Delta > 0$. Defining $L(\theta,\dot{\theta}) = 2 - \text{ME}(\theta,\dot{\theta})$, we see that under this action set the conditions of Theorem 1 are satisfied. This means that the agent is guaranteed to reach $G2$, and by extension, $G1$ on every trial (Property 1 with $T = G1$). We also know that the pendulum stays in $G1$ forever as it asymptotically approaches the origin, satisfying Property 3 with $T = G1$ and Property 5 with $T = \{(0,0)\}$. Further, the pendulum is guaranteed to remain in the set $T = \{(\theta,\dot{\theta}) : \text{ME}(\theta,\dot{\theta}) \leq 2\}$, which can be viewed as a sort of safety constraint. The agent cannot pump an arbitrarily large amount of energy into the pendulum, which in reality might result in damage to the equipment or dangerous fast motions. We see that many reassuring qualitative properties can be established for the RL agent by virtue of the actions we have designed for it.

In experiment 2, the RL agent chose from four actions, corresponding to control laws CL1, CL2, CL4, and CL5. In experiment 3, there were just two actions, corresponding to control laws CL1 and CL2. For both of these experiments the goal set was $G2$ and the costs for actions the same as in experiment 1. In these two experiments, the pendulum is

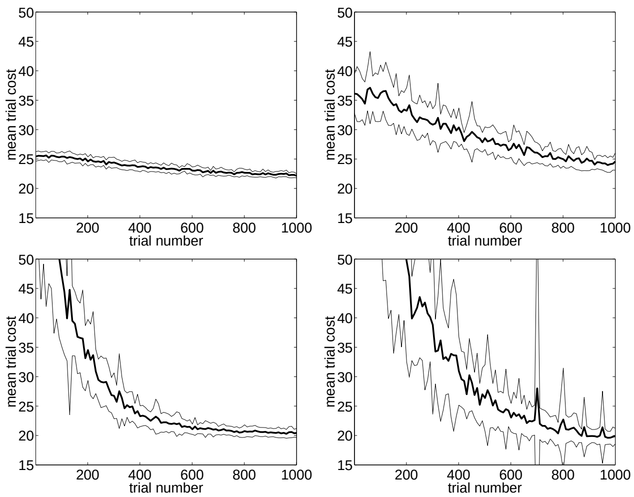


Figure 2: Mean time to $G1$ during test trials with $\pm$ 1 standard deviation lines. Top row: experiments 1 and 2. Bottom row: experiments 3 and 4.

| | | Exp1 | Exp2 | Exp3 | Exp4 |
|---|---|---|---|---|---|
| % of first 10 | Learn | 100 | 100 | 100 | 88.4 |
| reaching $G1$ | Test | 100 | 96.0 | 82.4 | 50.4 |
| Time to $G1$ | Learn | 25.4 $\pm$1 | 35.8 $\pm$4 | 62.1 $\pm$12 | 240 $\pm$68 |
| first 10 | Test | 25.4 $\pm$1 | 36.1 $\pm$4 | 77.7 $\pm$31 | 272 $\pm$88 |
| Time to $G1$ | Learn | 21.2 $\pm$.05 | 21.0 $\pm$.08 | 21.4 $\pm$.08 | 22.4 $\pm$.1 |
| last 1000 | Test | 21.0 $\pm$.06 | 19.9 $\pm$.1 | 19.5 $\pm$.1 | 18.7 $\pm$.2 |
| Longest | Learn | 34.6 | 80.9 | 171 | timeout |
| trial | Test | 35.2 | timeout | timeout | timeout |

Figure 3: Summary statistics for experiments 1–4.

guaranteed to remain in the set $T = \{(\theta, \dot\theta) : \mathrm{ME}(\theta, \dot\theta) \leq 2\}$ (Property 2) by virtue of the definition of $G2$. However, none of the other properties enjoyed by the agent in experiment 1 hold. In experiment 4, the agent had three actions to choose from, corresponding to control laws CL1, CL2, and CL3. In this case, we used $G1$ as the goal set. The cost for taking an action was the negative of the time spent outside $G1$ (e.g., simply $-1$ if the action does not take the pendulum into $G1$). There was no terminal cost in this case.

We have described the experiments in this order because this order represents decreasing dependence on Lyapunov domain knowledge. In experiment 1, the agent is constrained to make the pendulum ascend on mechanical energy and uses the $G2$ goal set. In experiment 2, the agent has actions based on the Lyapunov-designed controllers CL4 and CL5, and uses goal set $G2$. But the agent is not constrained to ascend on mechanical energy. In experiment 3, we drop the actions based on the Lyapunov-designed control laws, and in experiment 4 we drop the $G2$ goal set. (We also add CL3 because it is virtually impossible to hit the small goal set, $G1$, by switching between CL1 and CL2 in discrete time.) Experiment 4 might be considered the standard formulation of the control problem for RL.

## Results and Discussion

Figure 2 displays the mean total cost (i.e., time to $G1$) of test trials in the four experiments, along with lines depicting $\pm 1$ standard deviation. Each data point corresponds to a block of ten trial trials averaged together; trials that time out are removed from these averages. The curves are plotted on the same scale to facilitate qualitative comparison. The horizontal axis shows the trial number, going up to 1,000 of the 20,000 total trials. Most striking is the low-variability, good initial performance, and rapid learning of the "safe" RL agent in experiment 1. This contrasts sharply with the results in experiment 4, which show a more typical profile for RL systems solving dynamical control tasks. In experiments 2 and 3, we see intermediate behavior. Figure 3 gives more detailed statistics summarizing the experiments. The table reports the percentage of the first ten trials which make it to $G1$ before timing out at 900 time steps, the mean total trial cost for the first ten trials (omitting trials that timed out), the same for the last 1000 trials, and the worst trials seen in the different experimental configurations. The means are accompanied by 95% confidence intervals. The results show that the greater the reliance on Lyapunov domain knowledge, the better the initial and worst-case performance of the agent. Importantly, only in experiment 1, for which theoretical guarantees on reaching the goal were established, did the RL agent reach to goal on every learning and test trial.

The performance figures for the final 1000 trials reveal significant improvements from initial performance in all experiments. In the end, all the RL agents are also outperforming the simple policy of using CL4 or CL5 to pump energy into the pendulum until it reaches $G2$ and then letting the pendulum swing upright. Using CL4 is the better of the two, but still incurs a total cost of 24.83. The numbers also reveal a down-side to using Lyapunov domain knowledge. The constraint to descend on the Lyapunov function limits the ability of the agent in experiment 1 to minimize the time to $G1$. The agents in experiments 2 and 3 do better because they are not constrained in this way. Recall that the time derivative of the pendulum's mechanical energy is $\dot{\mathrm{ME}}(\theta, \dot\theta) = \dot\theta\, u$. In experiment 1, the agent always chooses a $u$ that matches $\dot\theta$ in sign, thus constantly increasing mechanical energy. However note that when $\dot\theta$ is close to zero, mechanical energy necessarily increases only slowly. Every time the pendulum changes direction, $\dot\theta$ must pass through zero, and at these times the agent in experiment 1 does not make much progress towards reaching $G2$. It turns out that a better policy in the long run is to accelerate against $\dot\theta$ and turn the pendulum around rapidly. This way, the pendulum spends more time in states where $\dot\theta$ is far from zero, meaning that energy can be added to the system more quickly. The agent in experiment 1 is able to achieve this to some degree, by switching from CL4 to the lower-torque CL5, allowing more rapid turn-around. But this does not match the freedom afforded in experiments 2 and 3. The agent in experiment 4 gains yet another advantage. It is not constrained to choosing $u = 0$ once the pendulum has reached $G2$ and before it has reached $G1$. The agent is able to continue accelerating the pendulum towards upright and then decelerate it as it approaches, resulting in even faster trajectories to $G1$.

## Stochastic Swing-Up and Balance 1

We now examine applying the ideas of the previous section to a stochastic control problem. Stochastic control problems are more challenging than their deterministic counterparts. Absolute guarantees on system behavior are generally not possible, and the randomness in the system behavior generally necessitates greater computational effort and/or longer learning times. In this section we assume a pendulum whose dynamics are modeled by the stochastic control differential equation

$$d\dot{\theta} = sin(\theta)dt + u(1 + 0.1dW) \ ,$$

where the $W$ denotes a standard Wiener process and the other variables are as before. Intuitively, this means that the intended control input $u$ is continuously, multiplicatively disturbed by Gaussian noise with mean 1 and standard deviation 0.1. This could model, for example, a pendulum system in which the power supply to the motor varies randomly over time.

### Experiments

In experiments 5 through 8 we used the same action definitions, goal sets, costs, learning algorithms, etc. as in experiments 1 through 4. In short, the only difference in experiments 5 through 8 is the stochastic dynamics of the pendulum. In this model, no controller can guarantee getting to the $G1$ or $G2$ goal sets. It is always possible that the random disturbances would "conspire" against the controller and thwart any progress. Note, however, that when $u$ is uniformly zero the noise term vanishes. The dynamics then match the deterministic case. This means that in any state with mechanical energy of 2, zero control input allows the pendulum to naturally swing up to the origin. Further, although controlling the pendulum by CL4 and CL5 does not guarantee increasing the pendulum's energy, it is possible to show that with some probability $p > 0$, at least $\Delta$ energy is imparted, regardless of the starting state. Intuitively, this is because a $\Delta$ increase is guaranteed under the deterministic dynamics, and the noise is smooth and matches the deterministic dynamics in expected value. So, there is some probability of an outcome that is close to the deterministic outcome. Recalling the definition $L(\theta, \dot{\theta}) = 2 - \mathrm{ME}(\theta, \dot{\theta})$, we observe the actions defined for experiment 5 (corresponding to CL4 and CL5) satisfy Theorem 2, ensuring that the RL agent reaches $G2$ with probability one on every trial. In turn, this ensures that the trajectories afterward reach $G1$ and asymptotically reach the origin. Thus the RL agent in experiment 5 enjoys the same Properties 1, 3, and 5 as the agent in experiment 1, but in the probability 1 sense, rather than for certain. The agents in experiments 5, 6, and 7, which use goal set $G2$, are guaranteed to remain in the set of states with mechanical energy no more than 2 (Property 2 with $T = \{ME \leq 2\}$).

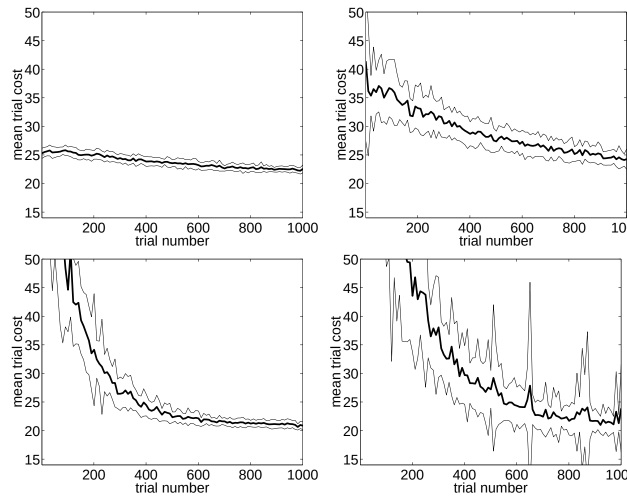


Figure 4: Mean time to $G1$ during test trials with $\pm$ 1 standard deviation lines. Top row: experiments 5 and 6. Bottom row: experiments 7 and 8.

### Results and Discussion

Figure 4 shows the mean costs of the first 1,000 test trials in experiments 5 through 8, and Figure 5 gives summary statistics. A control disturbance with 0.1 standard deviation seems fairly large. About 32% of the time one would expect a deviation of more than 10% in the intended control input, and about 5% of the time, a deviation of more than 20%. However, the results of these experiments are quite similar to those of experiments 1 through 4. One difference is that, in the present set of experiments, fewer test trials time out. For deterministic control problems, it is not uncommon to observe an RL agent temporarily learning a policy that causes the system to cycle endlessly along a closed loop in state space. This results in test trials timing out. When the system is stochastic, however, it is harder for such cycles to occur. Random disturbances tend to push a trajectory into a different part of state space where learning has been successful. (Similarly, random exploratory actions are part of the reason timeouts are less frequent in learning trials.) This may explain the reduced number of test trial timeouts early in learning, compared with the previous set of experiments. Another difference is that, for the stochastic pendulum, the policy of using CL4 to get to $G2$ performs better than it does on the deterministic pendulum. On the stochastic pendulum, it incurs an average total cost of 21.2. The agent in experiment 5 improved on this performance minimally, if at all. It may be that, within the constraints of ascending on mechanical energy and taking $u = 0$ after reaching $G2$, that CL4 performs near optimally. The agents in experiments 6, 7, and 8 were able to achieve significantly faster swing-up compared to CL4.

## Stochastic Swing-Up and Balance 2

In this section we present our third set of pendulum experiments. We assume pendulum dynamics described by the

| | | Exp5 | Exp6 | Exp7 | Exp8 |
|---|---|---|---|---|---|
| % of first 10 reaching $G1$ | Learn | 100 | 100 | 100 | 88.0 |
| | Test | 100 | 100 | 96.8 | 71.6 |
| Time to $G1$ first 10 | Learn | 25.5 $\pm$1 | 35.5 $\pm$4 | 59.0 $\pm$10 | 236 $\pm$75 |
| | Test | 25.3 $\pm$1 | 41.4 $\pm$18 | 85.2 $\pm$42 | 269 $\pm$85 |
| Time to $G1$ last 1000 | Learn | 21.5 $\pm$.04 | 21.1 $\pm$.03 | 21.7 $\pm$.03 | 22.5 $\pm$.07 |
| | Test | 21.0 $\pm$.05 | 20.2 $\pm$.04 | 20.3 $\pm$.05 | 19.0 $\pm$.06 |
| Longest trial | Learn | 37.1 | 78.3 | 206 | timeout |
| | Test | 40.7 | 706 | timeout | timeout |

Figure 5: Summary statistics for experiments 5–8.

system of stochastic control differential equations:

$$d\theta = \dot{\theta}dt + 0.1dW \ ,$$

$$d\dot{\theta} = sin(\theta)dt + udt \ .$$

This describes a mean-zero standard deviation one disturbance to the position variable. For this problem we define no goal set. However, we let $T = \{(\theta, \dot{\theta}) : ||\theta|| \leq 0.5\}$, and declare that the agent incurs unit cost per time as long as the pendulum is not in $T$, and incurs no cost while the pendulum is in $T$. The set $T$ corresponds to a range of positions roughly 30 degrees or less from upright.

We define several new control laws for this problem. One is just the zero torque control law:

**CL6**$(\theta, \dot{\theta}) = 0$.

We define three more control laws that rely on CL4 (the stronger MEA controller) to increase the pendulum's energy to 2 when it is outside of $T$, but apply different, constant control torques inside $T$.

$$\textbf{CL7}(\theta, \dot{\theta}) = \begin{cases} \text{CL4}(\theta, \dot{\theta}) & \text{if } (\theta, \dot{\theta}) \notin T \text{ and } \text{ME}(\theta, \dot{\theta}) < 2 \\ 0 & \text{if } (\theta, \dot{\theta}) \notin T \text{ and } \text{ME}(\theta, \dot{\theta}) \geq 2 \\ -u_{max} & \text{if } (\theta, \dot{\theta}) \in T \end{cases}$$

$$\textbf{CL8}(\theta, \dot{\theta}) = \begin{cases} \text{CL4}(\theta, \dot{\theta}) & \text{if } (\theta, \dot{\theta}) \notin T \text{ and } \text{ME}(\theta, \dot{\theta}) < 2 \\ 0 & \text{otherwise} \end{cases}$$

$$\textbf{CL9}(\theta, \dot{\theta}) = \begin{cases} \text{CL4}(\theta, \dot{\theta}) & \text{if } (\theta, \dot{\theta}) \notin T \text{ and } \text{ME}(\theta, \dot{\theta}) < 2 \\ 0 & \text{if } (\theta, \dot{\theta}) \notin T \text{ and } \text{ME}(\theta, \dot{\theta}) \geq 2 \\ +u_{max} & \text{if } (\theta, \dot{\theta}) \in T \end{cases}$$

## Experiments

We performed two experiments. In experiment 9, the RL agent chose from actions corresponding to CL7, CL8, and CL9. Each of these controllers acts to bring the pendulum's energy up to 2 in any state outside $T$. Because a pendulum with that much energy naturally swings upright, the agent in experiment 9 can be guaranteed (with probability 1, because of the position disturbance) to reach $T$ from any initial state (Property 1). The position disturbance rules out, however, that any controller could keep the pendulum in $T$ indefinitely. So, the best one can say about the agent in experiment 9 is that it would return the pendulum to $T$ an infinite number of times (Property 4). In experiment 10, the agent chose from actions corresponding to CL1, CL2, and CL6—i.e., $-u_{max}$, 0, and $+u_{max}$ torques.

We performed 25 independent learning runs with 2,000 learning and 2,000 test trials in each run. Since there is no goal set, each trial was run for a full 900 time steps. We also instituted a discount rate of $\gamma = 0.95$. Agents chose a random action on a given time step with probability 0.02. We used the same learning algorithm, Sarsa($\lambda$) with $\lambda = 0.8$, and CMACs as in all previous experiments.

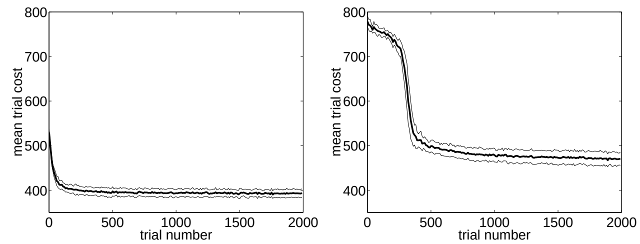


Figure 6: Mean trial costs with $\pm$1 standard deviation lines. Left: experiment 9. Right: experiment 10.

| | | Exp9 | Exp10 |
|---|---|---|---|
| Cost of first 10 trials | Learn | 531 $\pm$ 2 | 762 $\pm$ 1 |
| | Test | 529 $\pm$ 3 | 778 $\pm$ 5 |
| Cost of last 100 trials | Learn | 399 $\pm$ 3 | 479 $\pm$ 5 |
| | Test | 393 $\pm$ 3 | 470 $\pm$ 6 |
| Costliest Trials | Learn | 559.3 | 808.9 |
| | Test | 582.6 | 900.0 |

Figure 7: Summary statistics for experiments 9 and 10.

## Results and Discussion

Figures 6 and 7 summarize the results of the experiments. The agent with the Lyapunov-based actions has better initial and final performance than the agent with the naive, constant-torque actions. It is possible that the naive agent would catch up eventually; the performance curve shows a slight downward trend. However, the learning seems quite slow, and even this performance results from several hours of hand-tuning the learning parameters. In both experiments the agents outperform all individual control laws. The best single control law for the task is CL8, which incurs an average cost of 514 per trial. In neither experiment is performance significantly variable across runs. However, we note again the good initial performance supplied by the Lyapunov constraints and the excellent worst-case performance. Surprisingly, in some of the test trials in experiment 10 the pendulum never entered $T$, resulting in a total cost of 900 for the trial. This was not a common occurrence, but it did happen a total of 18 times in 14 of the 25 independent runs. The latest occurrence was in the 174th trial of run 14—after $156,600 = 174 \times 900$ learning steps, so it was not simply a result of early, uninformed behavior.

## Conclusions

We have described a general approach to constructing safe RL agents based on Lyapunov analysis. Using Lyapunov methods, we can ensure an agent achieves goal states, or causes the environment to remain in, avoid, or asymptotically approach certain subsets of state space. Compared with the typical RL methodology, our approach puts more effort on the system designer. One must identify a candidate Lyapunov function and then design actions that descend appropriately while allowing costs to be optimized. In cases where safety is not so important, the extra design effort may not be worthwhile. However, there are many domains where performance guarantees are critical and Lyapunov and other analytic design methods are commonplace. We have used simple pendulum control problems to demonstrate our ideas. However, Lyapunov methods find many important applications in, for example, robotics, naval and aerospace control and navigation problems, industrial process control, and multi-agent coordination. Analytical methods are useful for the theoretical guarantees they provide, but do not by themselves usually result in optimal controllers. RL techniques are capable of optimizing the quality of control, and we believe there are many opportunities for control combining Lyapunov methods with RL.

## Acknowledgements

This work was supported in part by the National Science Foundation under Grant No. ECS-0070102. Theodore Perkins was also support by a graduate fellowship from the University of Massachusetts Amherst. We thank Daniel Bernstein and Doina Precup, who offered helpful comments on drafts of this document.

## References

A. G. Barto and T. J. Perkins. Toward on-line reinforcement learning. In *Proceedings of the Eleventh Yale Workshop on Adaptive and Learning Systems*, pages 1–8, 2001.

G. Boone. Efficient reinforcement learning: Model-based acrobot control. In *1997 International Conference on Robotics and Automation*, pages 229–234, 1997.

G. Boone. Minimum-time control of the acrobot. In *1997 International Conference on Robotics and Automation*, pages 3281–3287, 1997.

M. S. Branicky, T. A. Johansen, I. Peterson, and E. Frazzoli. On-line techniques for behavioral programming. In *Proceedings of the IEEE Conference on Decision and Control*, 2000.

J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Addison-Wesley, 1989.

R. H. Crites and A. G. Barto. Elevator group control using multiple reinforcement learning agents. *Machine Learning*, 33:235–262, 1998.

G. DeJong. Hidden strengths and limitations: An empirical investigation of reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 215–222. Morgan Kaufmann, 2000.

D. F. Gordon. Asimovian adaptive agents. *Journal of Artificial Intelligence Research*, 13:95–153, 2000.

M. Huber and R. A. Grupen. Learning robot control–using control policies as abstract actions. In *NIPS'98 Workshop: Abstraction and Hierarchy in Reinforcement Learning, Breckenridge, CO*, 1998.

S. Meyn and R. Tweedie. *Markov Chains and Stochastic Stability*. Springer-Verlag, New York, 1993.

R. Neuneier and O. Mihatsch. Risk sensitive reinforcement learning. In *Advances in Neural Information Processing Systems 11*, pages 1031–1037, Cambridge, MA, 1999. MIT Press.

T. J. Perkins and A. G. Barto. Heuristic search in infinite state spaces guided by lyapunov analysis. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 242–247, San Francisco, 2001. Morgan Kaufmann.

T. J. Perkins and A. G. Barto. Lyapunov-constrained action sets for reinforcement learning. In *Machine Learning: Proceedings of the Eighteenth International Conference*, pages 409–416, San Francisco, 2001. Morgan Kaufmann.

T. J. Perkins. *Qualitative Control Theoretic Approaches to Safe Approximate Optimal Control*. PhD thesis, University of Massachusetts Amherst, 2002. Forthcoming.

J. G. Schneider. Exploiting model uncertainty estimates for safe dynamic control learning. In *Advances in Neural Information Processing Systems 9*, pages 1047–1053, Cambridge, MA, 1997. MIT Press.

S. P. Singh, A. G. Barto, R. Grupen, and C. Connolly. Robust reinforcement learning in motion planning. In *Advances in Neural Information Processing Systems 6*, pages 655–662, Cambridge, MA, 1994. MIT Press.

M. W. Spong. The swing up control problem for the acrobot. *Control Systems Magazine*, 15(1):49–55, 1995.

R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press/Bradford Books, Cambridge, Massachusetts, 1998.

G. J. Tesauro. Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219, 1994.

J. N. Tsitsiklis and B. Van Roy. Optimal stopping of markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives. *IEEE Transactions on Automatic Control*, 44(10):1840–1851, 1999.

T. L. Vincent and W. J. Grantham. *Nonlinear and Optimal Control Systems*. John Wiley & Sons, Inc., New York, 1997.

D. Weld and O. Etzioni. The first law of robotics (a call to arms). In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 1042–1047, 1994.

W. Zhang and T. G. Dietterich. High performance job-shop scheduling with a time-delay td($\lambda$) network. In *Advances in Neural Information Processing Systems 8*, pages 1024–1030, Cambridge, MA, 1996. MIT Press.