This problem set is due *by 11:59pm* on **Tuesday, May 5**.

Problem 1 (The Weight-Perturbation Algorithm) *In this problem you will investigate one of the REINFORCE type algorithms, namely the Weight Perturbation Algorithm. A big advantage of policy gradient algorithms (vs. state space value methods) is that they scale much better to high dimensions. In addition, a compact representation of your policy can dramatically improve learning performance. For this problem, you will solve the cart-pole swing-up task using a compact policy representation, where the policy is parameterized by the energy-based swing-up controller (the same one you implemented for Pset 2). The goal of the algorithm is to learn the feedback gains K_e, K_p, K_d by minimizing the cost function*

$$J = \min_{\mathbf{x}_n}[\frac{1}{2}(\mathbf{x}_n - \mathbf{x}_{des})^T \mathbf{Q}(\mathbf{x}_n - \mathbf{x}_{des}), n = 0, \ldots, N] \tag{1}$$

where $\mathbf{x}_{des} = [0 \;\; \pi \;\; 0 \;\; 0]^T$, $\mathbf{Q} = diag([100 \;\; 100 \;\; 100 \;\; 100])$, and N denotes the number of time steps. Download the file `cartpole_wp.m` *from the course website. This file contains a (nearly) complete implementation of the weight perturbation algorithm for the cart-pole swing-up task. Your goal is to finish the algorithm implementation by completing the following:*

a) Add code which evaluates the above cost after each trial as well as the weight-perturbation update itself. Also set the gains in the `control` *function appropriately. Submit your code, a plot of the learning curve, and the final trajectory.*

b) In addition to the controller gains, suppose the parameters (m_c, m_p, l) are also unknown. Describe how you would modify the learning procedure to address this issue.

Problem 2 (Temporal Difference Learning) *Temporal Difference Learning (TD-Learning) is a model-free value method that can efficiently learn the value function of a system executing a given policy. In this problem we will ask you to use TD(λ) to learn a value function on a markov chain derived from the torque limited pendulum swing-up problem. Download the file* `markov_td.m`*, which has a transition matrix for the markov chain and some code to help you get started. Use a discount factor of $\gamma = .95$ and the cost vector $\mathbf{g}$ given in the code.*

With the transition matrix you can solve for the exact value function easily. Compare this value function to the value function given by TD, and examine the error characteristics over time (e.g., squared error between true value and learned value vs. update number, but if you think other measures would be interesting or relevant, look at those as well).

How does the learning depend upon the value of the parameter λ? What sort of improvement can you obtain when you make λ depend on time? Does it make sense to reset the system after 40 transitions? How does this effect performance?

6.832 Underactuated Robotics
Spring 2009