ESE 650 Final Project - Raibert Hopper

Tarik Tosun

April 29, 2014

Raibert Controller

Flight: Adjust foot placement for next touchdown to control velocity

$$x_{foot,d} = \frac{\dot{x} T_{st}}{2} + K_{foot} (\dot{x} - \dot{x}_d)$$

Stance: Torque the body to adjust attitude

$$\tau(t) = p_{att} * \phi + d_{att} * \dot{\phi}$$

Hop height is maintained by applying a constrant thrust impulse on each hop.

Demo

TD Gain Learning

States: $\dot{x}$, $\dot{y}$, ϕ, θ, ϕ_{hip}, θ_{hip}

Actions: K_{foot}, p_{att}, d_{att}

Reward Function:

$$r(t) = \begin{cases} 1 - \gamma\left(\theta^2 + \phi^2 + \dot{x} + \dot{y}\right) & \text{if standing} \\ 0 & \text{if fallen over} \end{cases}$$

γ is a reward shaping parameter, and will be small ($\gamma = 0.05$)

Q-learning with Radial Basis Functions

- Distribute N gaussians over the state-action space
- Each has some Q-value Q_i

$$Q(s,a) = \sum_{i}^{N} k_i(s,a) Q_i$$

$$Q_i \leftarrow (1-\alpha_i) Q_i + \alpha_i \left\{ r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right\}$$

$$\alpha_i = \beta(t) k_i(s,a)$$

Training

- Generate many controllers and initial conditions
- Batch-learn the Q-function

Testing

- Run learned controller
- Select gains based on state

Future Robustness

- sarsa update rule

sarsa

$$Q(s_t, a_t) \leftarrow (1-\alpha)Q(s_t, a_t)+\alpha \{r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\}$$

- Uses the Q-value of the t+1 state actually chosen by the policy
- Accounts for stochasticity of ϵ-greedy control