Reinforcement Learning

- Basic idea:
  - Receive feedback in the form of rewards
  - Agent's utility is defined by the reward function
  - Must (learn to) act so as to maximize expected rewards
  - All learning is based on observed samples of outcomes!

Example: Learning to Walk

- Initial
- A Learning Trial
- After Learning (1K Trials)

Example: Toddler Robot

(Coehlo, Zhang and Saaraj, 2005)

Active Reinforcement Learning

(Videos: TODDLER – 40s)
# Active Reinforcement Learning

- **Full** reinforcement learning: optimal policies (like value iteration)
  - You don’t know the transitions $T(s,a,s')$
  - You don’t know the rewards $R(s,a,s')$
  - You choose the actions now
  - **Goal:** learn the optimal policy / values

- In this case:
  - Learner makes choices!
  - Fundamental tradeoff: exploration vs. exploitation
  - This is NOT offline planning! You actually take actions in the world and find out what happens...

## Value Iteration

- **Value iteration:** find successive (depth-limited) values
  - Start with $V_0(s) = 0$, which we know is right
  - Given $V_k$, calculate the depth $k+1$ values for all states:
    $$V_{k+1}(s) \leftarrow \max_{a} \left( R(s,a,s') + \gamma V_k(s') \right)$$

- But $Q$-values are more useful, so compute them instead
  - **Start with $Q_0(s,a) = 0$**, which we know is right
  - Given $Q_k$, calculate the depth $k+1 q$-values for all $q$-states:
    $$Q_{k+1}(s,a) \leftarrow \sum_{a'} T(s,a,s') \left[ R(s,a,s') + \gamma \max_{a'} Q_k(s',a') \right]$$

---

# Q-Learning

- **Q-Learning:** sample-based Q-value iteration
  $$Q_{k+1}(s,a) = \sum_{a'} T(s,a,s') \left[ R(s,a,s') + \gamma \max_{a'} Q_k(s',a') \right]$$

- **Learn** $Q(s,a)$ values as you go
  - Receive a sample $(s,a,s',r)$
  - Consider your old estimate: $Q(s,a)$
  - Consider your new sample estimate:
    $$\text{sample} = R(s,a,s') + \gamma \max_{a'} Q_k(s',a')$$
  - Incorporate the new estimate into a running average:
    $$Q(s,a) \leftarrow (1 - \alpha)Q(s,a) + \alpha \text{[sample]}$$

---

# Video of Demo Q-Learning -- Crawler

- **Amazing result:** Q-learning converges to optimal policy -- even if you’re acting suboptimally!
  - **This is called off-policy learning**

- **Caveats:**
  - You have to explore enough
  - You have to eventually make the learning rate small enough
  - ... but not decrease it too quickly
  - Basically, in the limit, it doesn’t matter how you select actions!