Learning II
15-491 CMRoboBits

Manuela Veloso
And
Brett Browning

Fall 2007

Outline

- Project advice
- Recap on learning
- Experts-based learning approaches

Project Advice

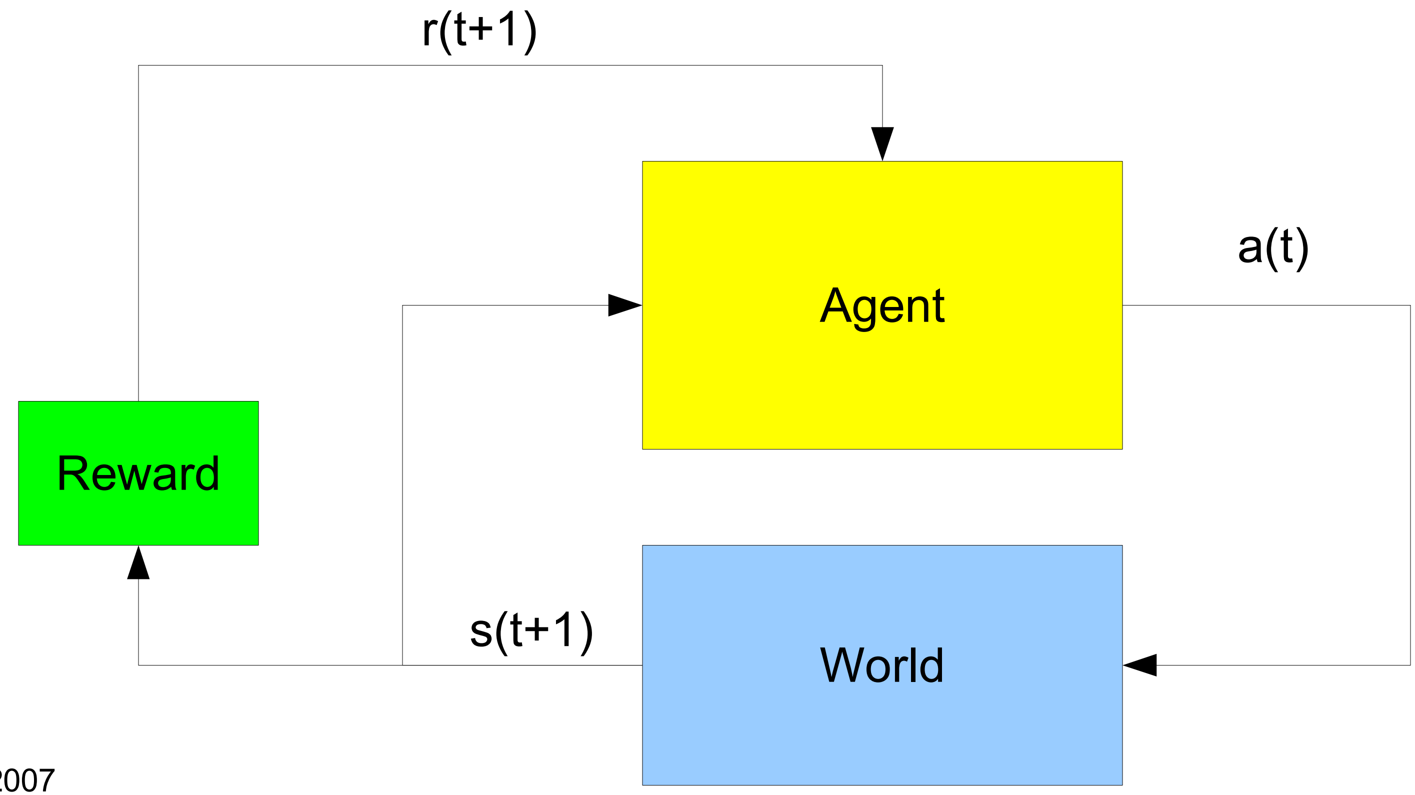
- Start early, work consistently
- Set short-term goals
  - Phases are there to help you
  - Evaluate and reassess at each milestone
- Take a cyclic approach: test often!
- Backup your work
- Write as you go

Learning Recap

- Supervised learning
  - Given sample set of (x,y), estimate y=f(x)
  - Calibration, prediction, learning by demonstration
  - Classification, ...
- Unsupervised learning
  - Given sample set of x, learn projection y=f(x)
  - Dimensionality reduction (PCA), clustering
- Semi-supervised learning
  - In-between, e.g. Reinforcement learning

Reinforcement Learning

- Agent acts in the world, and receives rewards
- Credit assignment problem



RL and Robotics

- Good for learning robotics control problems
- Algorithms
  - Q-learning, SARSA-learning, TD-lambda
  - Policy iteration, value iteration
- Sounds great, ...but... algorithms don't scale
  - High dimensionality
  - Continuous state/action spaces
  - Non-deterministic execution
  - Even worse for multiple agents

RL Examples

- Not all bad news
- Some examples
  - Restricted policy RL* and Pegasus
  - Gra-Wolf
- Still a very active research area!!!

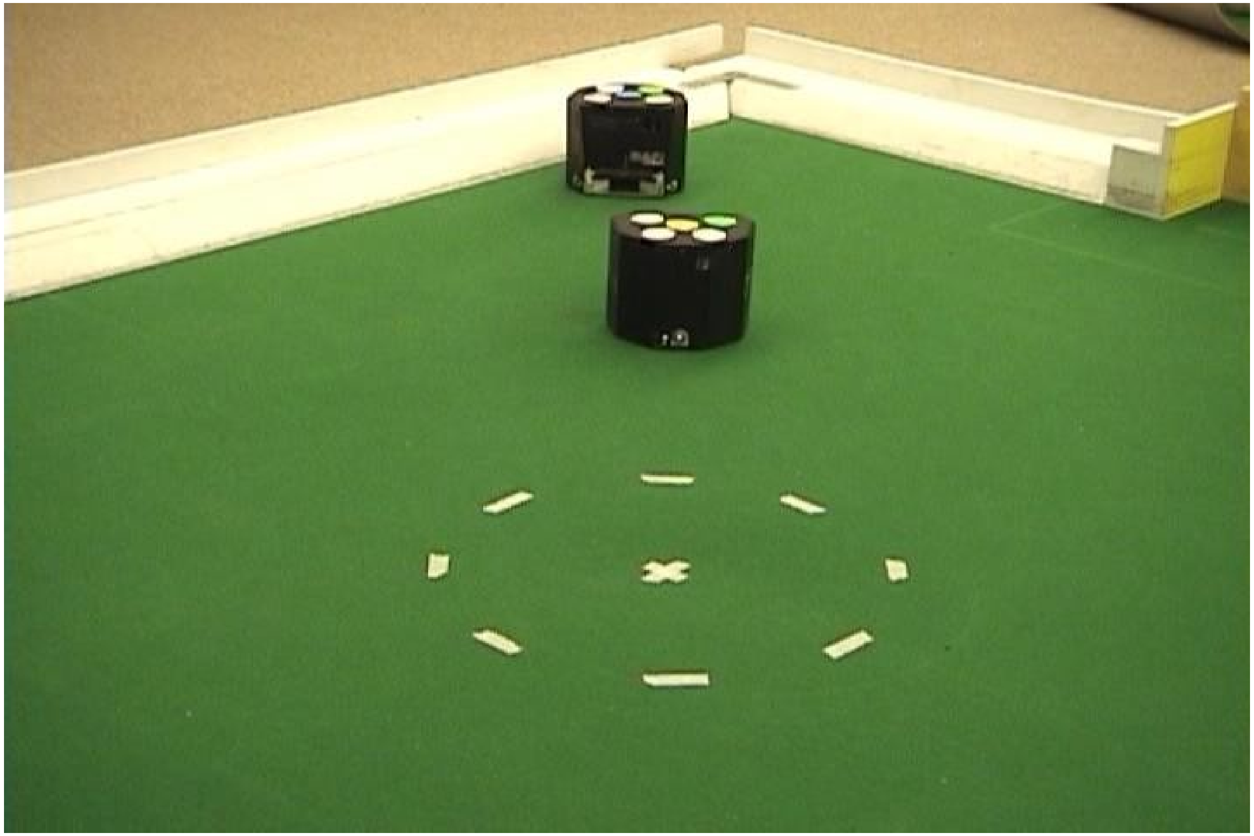
* - My terminology

Learning Helicopter Control

- Ng, Bagnell, et al.

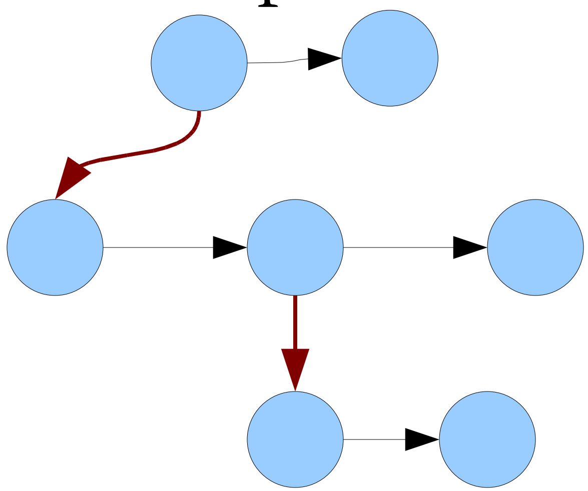
Multi-Robot Learning

- Bowling et al.



Learning in State Machines

- State machines provide task decomposition



- Three key learning problems
  - Control policy learning in a state: RL, LbD
  - Learning transition policy (and sub-skills)
  - Learning the hierarchy: (e.g. Options, hierarchical RL)

Learning Transitions

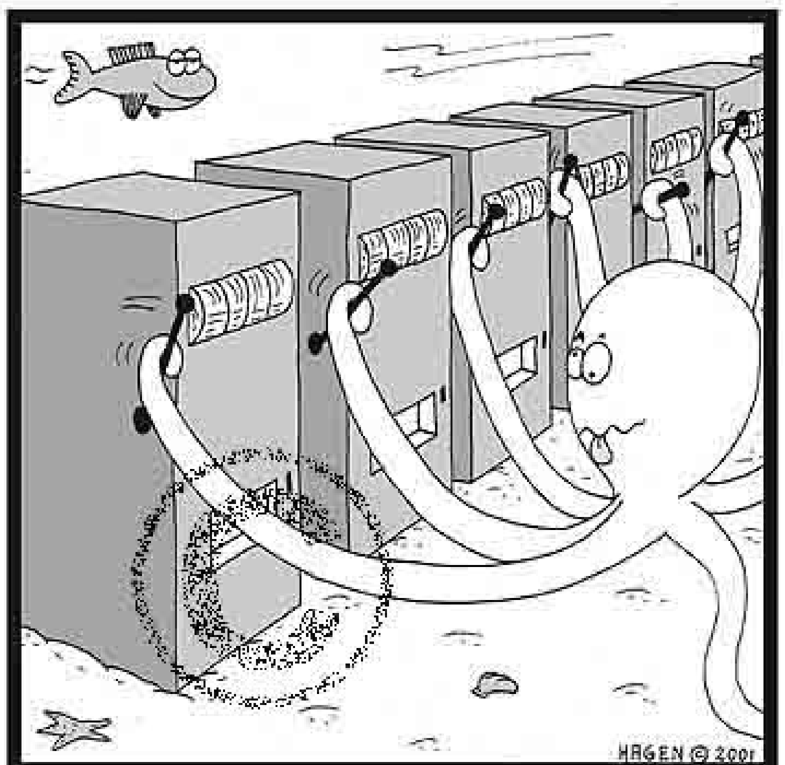
- Can be a hard problem
  - Continuous state/action space
  - Partial observability
  - Data collection can be hard
- Let's look at a simpler problem: Selection

State Machine Selection

- Given a set of state machines (complex actions)
  - A1, A2, ..., Ak
  - k should be small (< 10)
- Learn to select which one to use
  - Note, we are avoiding representing state, and are explicitly handling discrete actions
- How can we do this?

Experts Learning

- Let's go to the casino
- K-armed bandits problem



Compulsive gambling

Choose a bandit at time t
Observe reward(s) r
Want to maximize our reward or minimize loss (regret)

More Formally…

- Each expert is a “one-armed-bandit”
- At time t
  - Select expert to use: Ai
  - Observe payoffs: r
  - Based on payoffs, update how we select experts in the future

Observing Payoffs

- Suppose you can observe all of the payoffs
  - Full observability
- Want to minimize regret

$$r_A - max_i r_i, \quad r_i = \sum_t r_i^t$$

- rA is algorithm reward
- Note, different formulations are possible

Simple Algorithm

- Stochastic policy, choose actions randomly
- Initialize weights w (e.g. w=0)
- Choose action according to

$$p_i = \frac{e^{w_i}}{\sum_i e^{w_i}}$$

- Update based on payoffs

$$w'_i = w_i + r_i$$

Partial Observability

- Generally can't see all the payoffs
- Exp3 Approach [Fruend & Schapire]
  - Select single action as before

$$p_i = \frac{e^{w_i}}{\sum_i e^{w_i}}$$

  - Observe reward r for action j, update

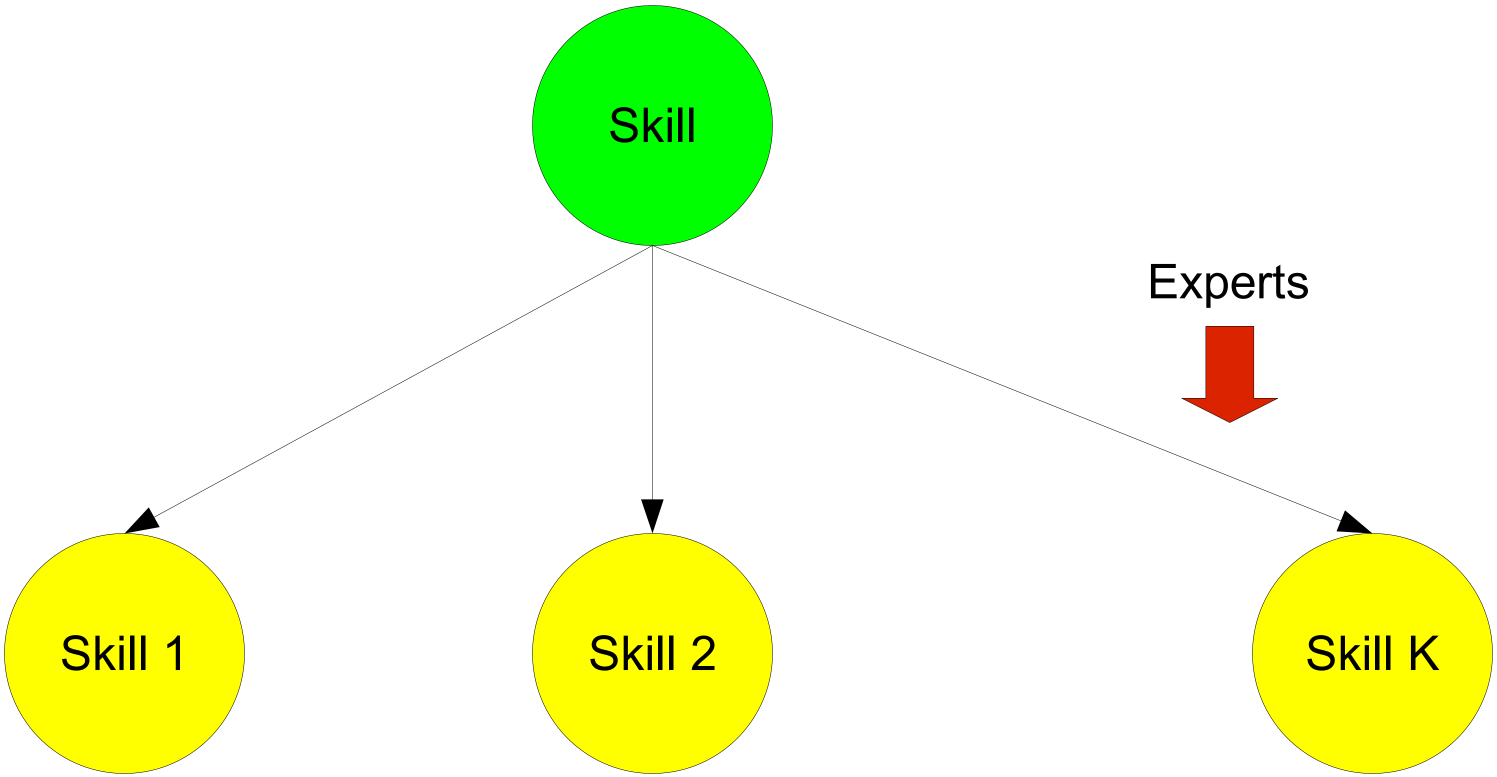
$$x_i = \frac{r_i}{p_i}, \quad r_i = 0, if\ i \neq j$$

$$w'_i = w_i e^{x_i \alpha}$$

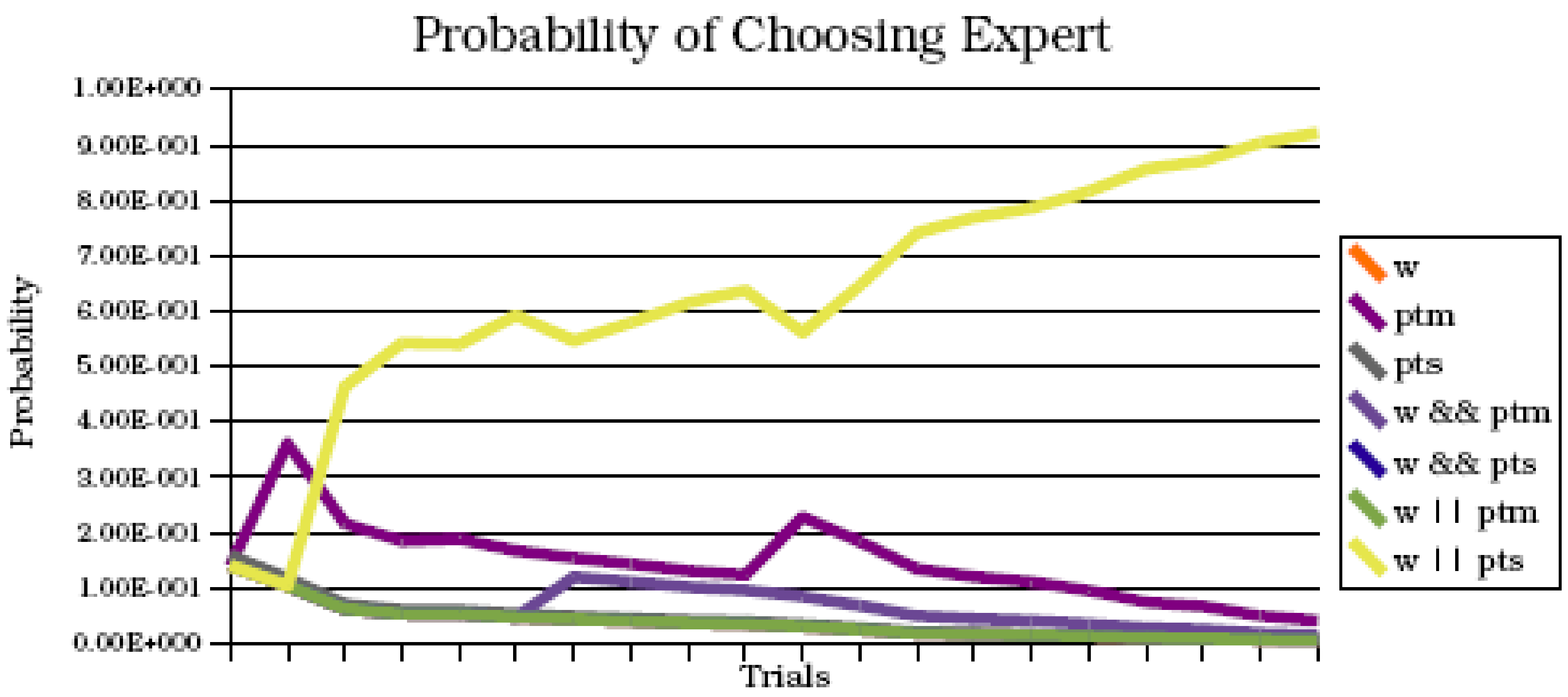
Applying it to Robots

- At skill level
  - Have multiple state machines to do a task
  - Use Exp3 to select which state machine to execute
  - Rewards from success/failure
- At play level
  - Have multiple plays to do a task
  - Use Exp3 to select which play to execute
  - Rewards from success/failure

Learning Skill Selection

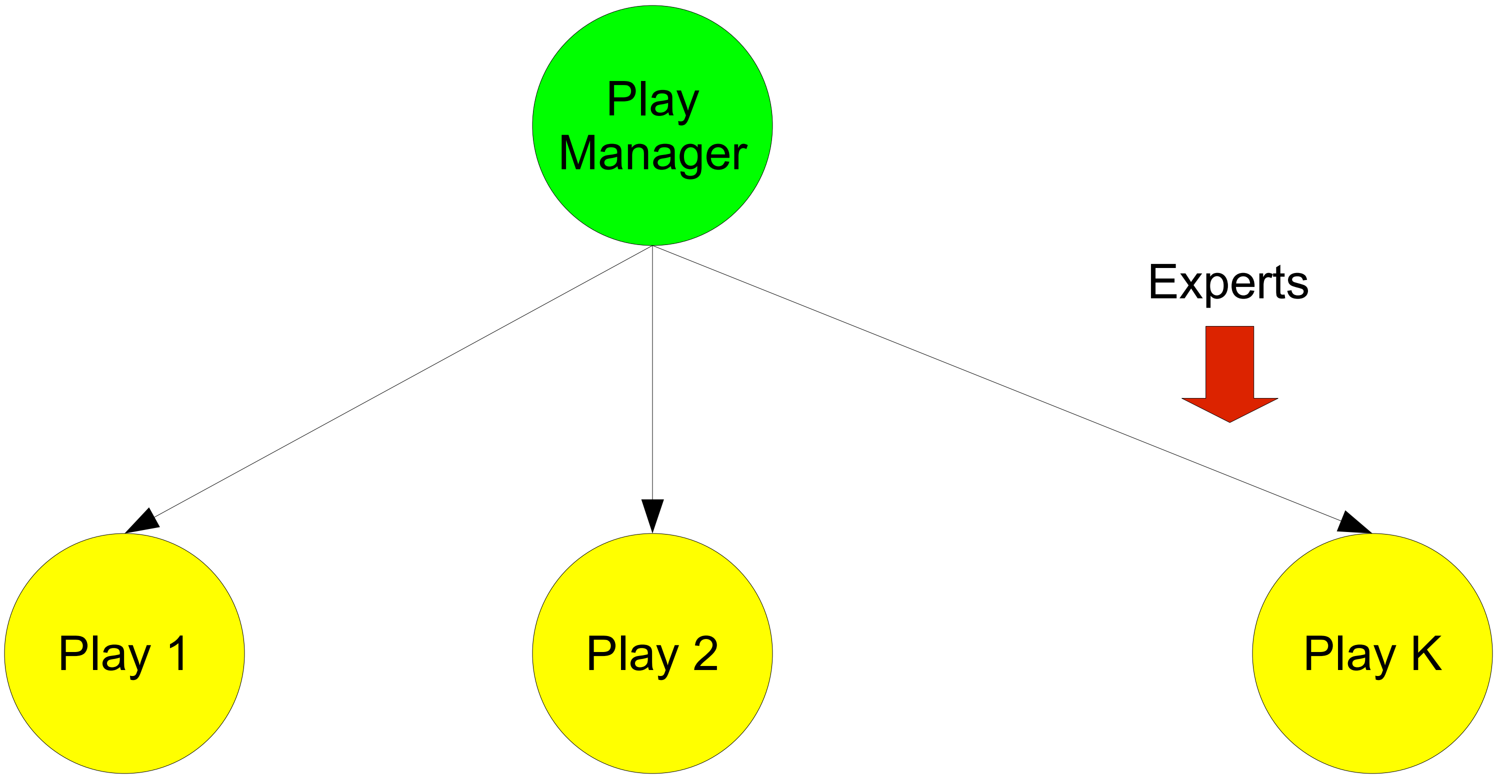


Skill Selection Learning



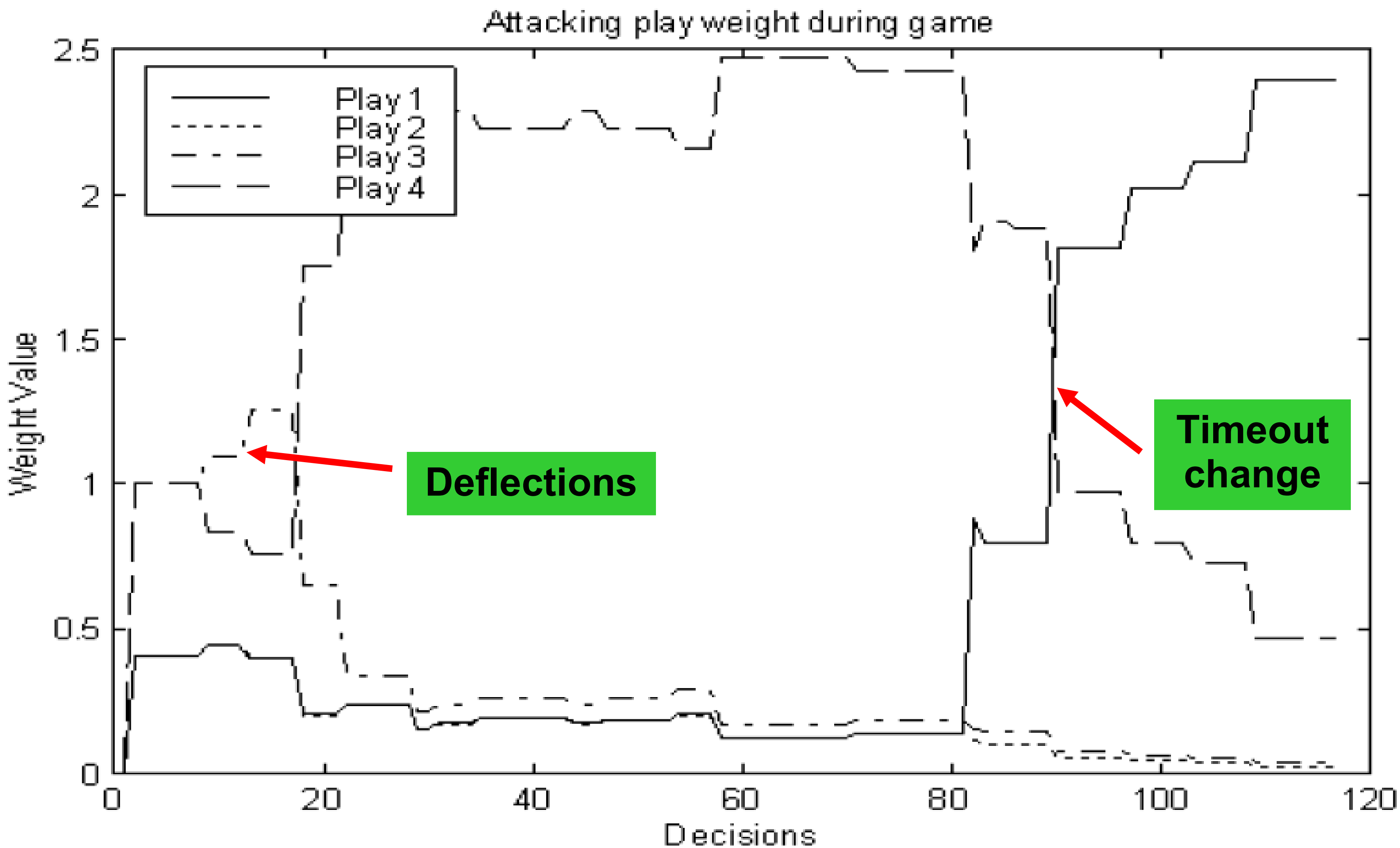
Argall et al.

Learning Play Selection



Not all plays are available at each time step

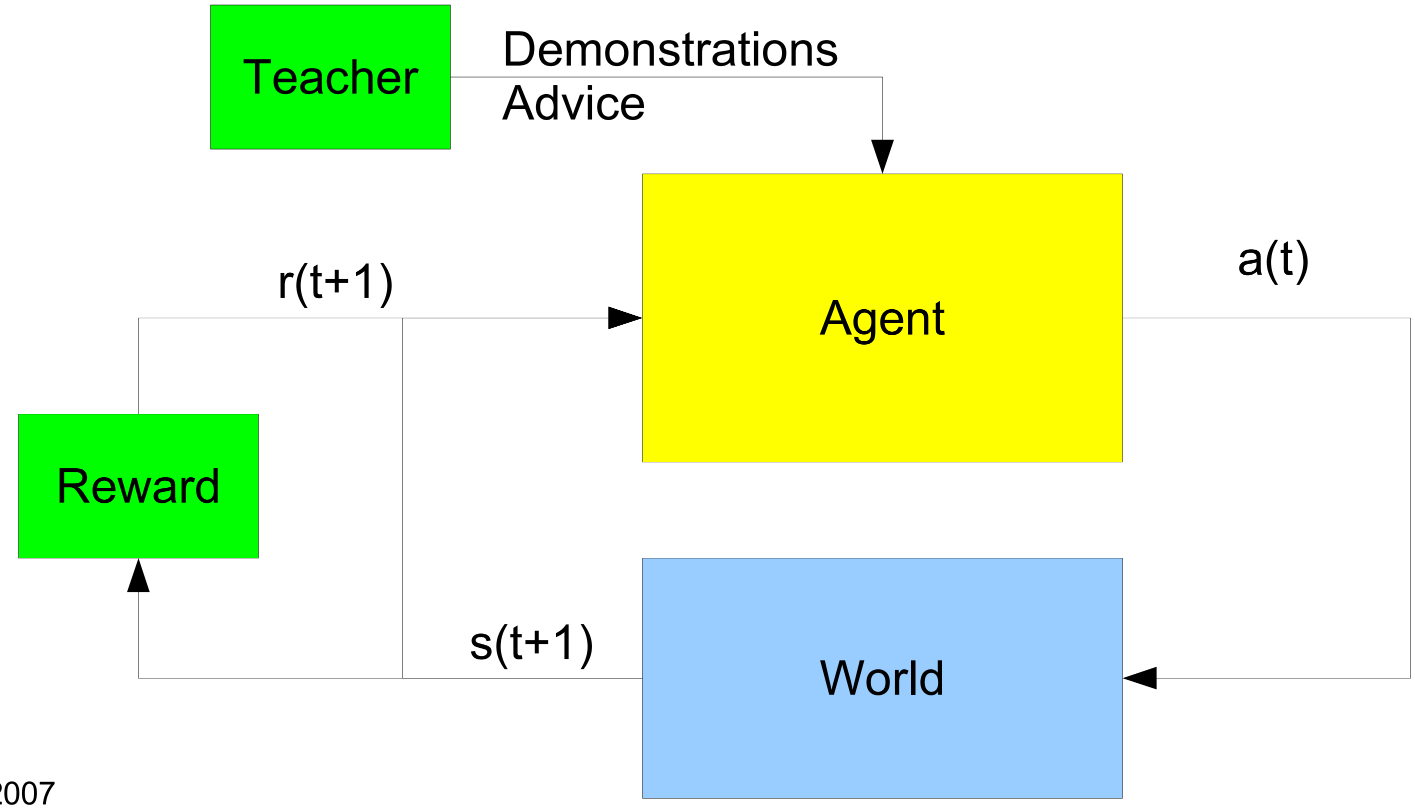
Play Selection



Bowling et al.

Learning by Demonstration

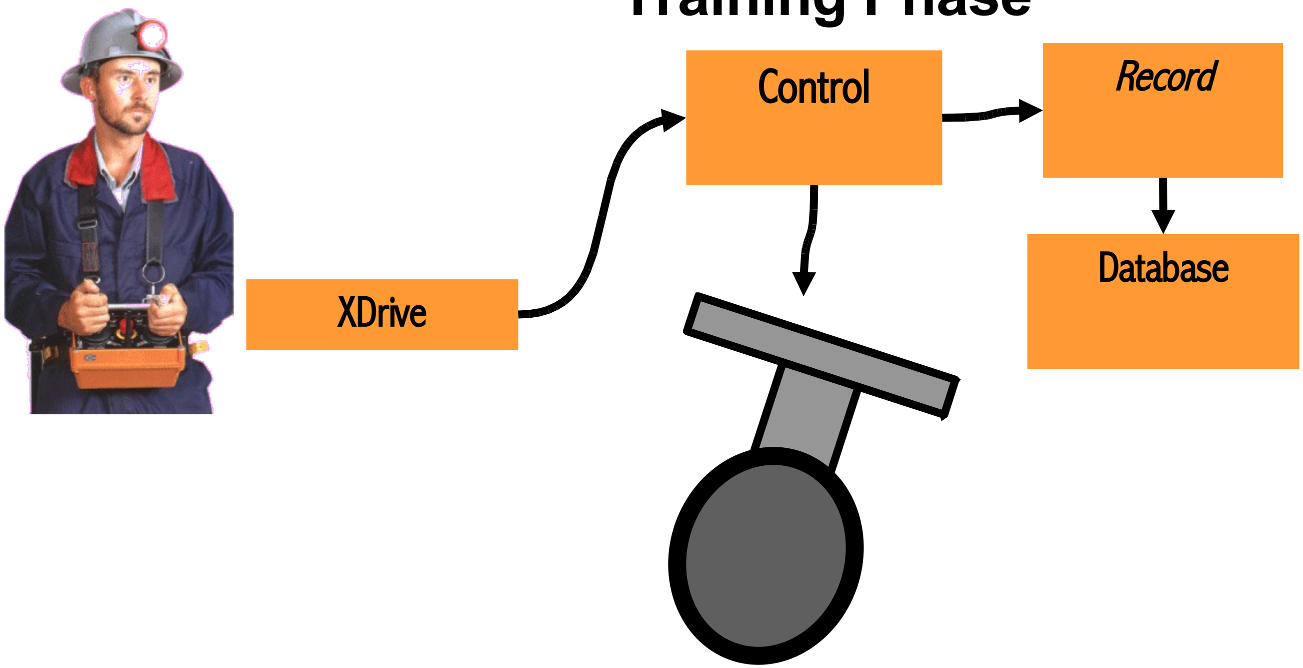
- Agent acts in the world, and receives rewards
- Teacher provides advice/demonstrations to agent



Demonstration

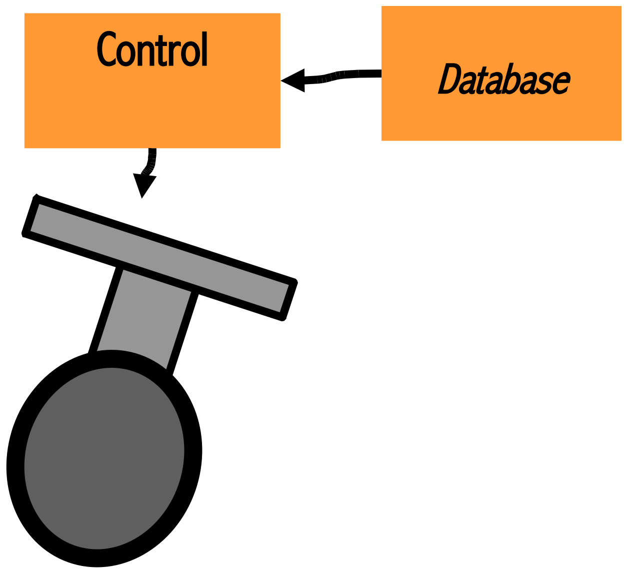
- Example motions provided by a human operator via tele-operation (or observation)

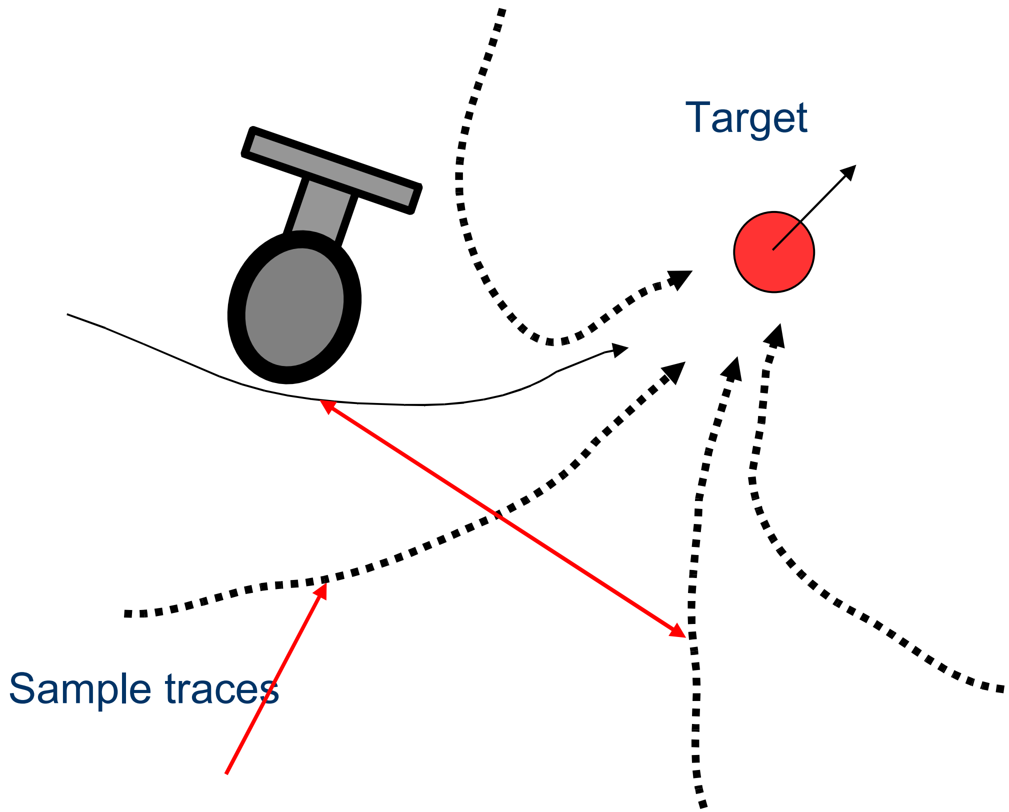
Training Phase



Execution

- Generalize from example traces to estimate teacher's control policy





As a Regression Problem

- Teacher provides examples via demonstration
  - (x0,y0), ...(xN, yN)
- Mapping problem
  - Translate teacher's representation into robot's perception and action space:
  - z=H(x), a=G(y)
- Regression problem
  - Learn a model for a=f(z)
  - f(.) is the control policy

Providing Advice

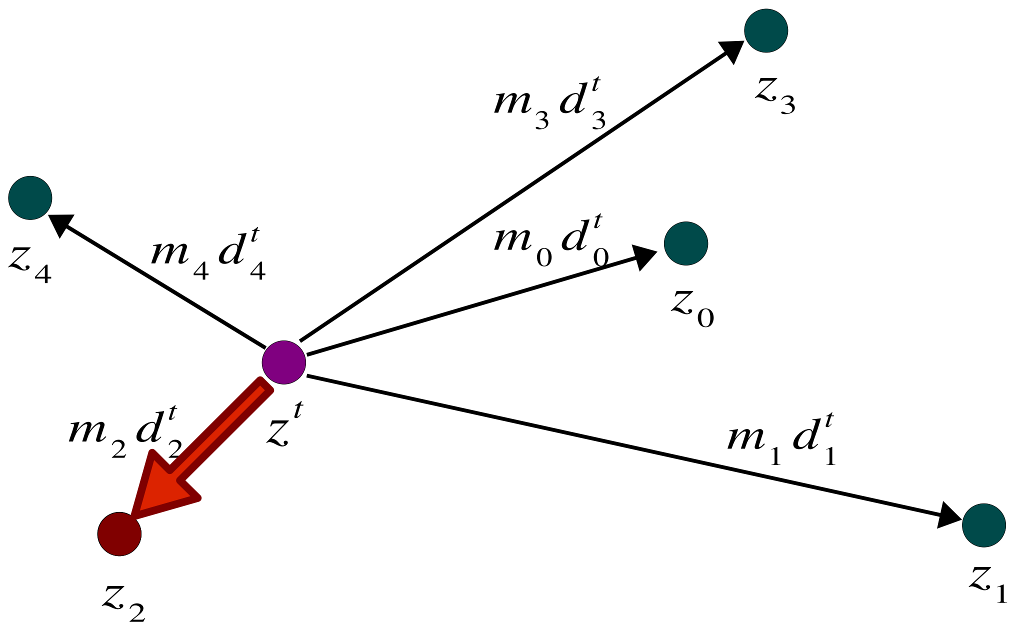
- Performance driven by data properties, representation and regression algorithm
- Can provide post-demonstration advice to *refine* control policy
- Advice method is an open research problem
- One example
  - Binary critique: label execution portions as good or bad

Advice with Binary Critiquing

Regression Technique : 1-NN $a^t = arg\,min_i\, D(z^t, z_i, m_i)$

Distance Metric :

$$D(z^t, z_i, m_i) = m_i (z^t - z_i)^T \Sigma^{-1} (z^t - z_i) = m_i d_i^t$$

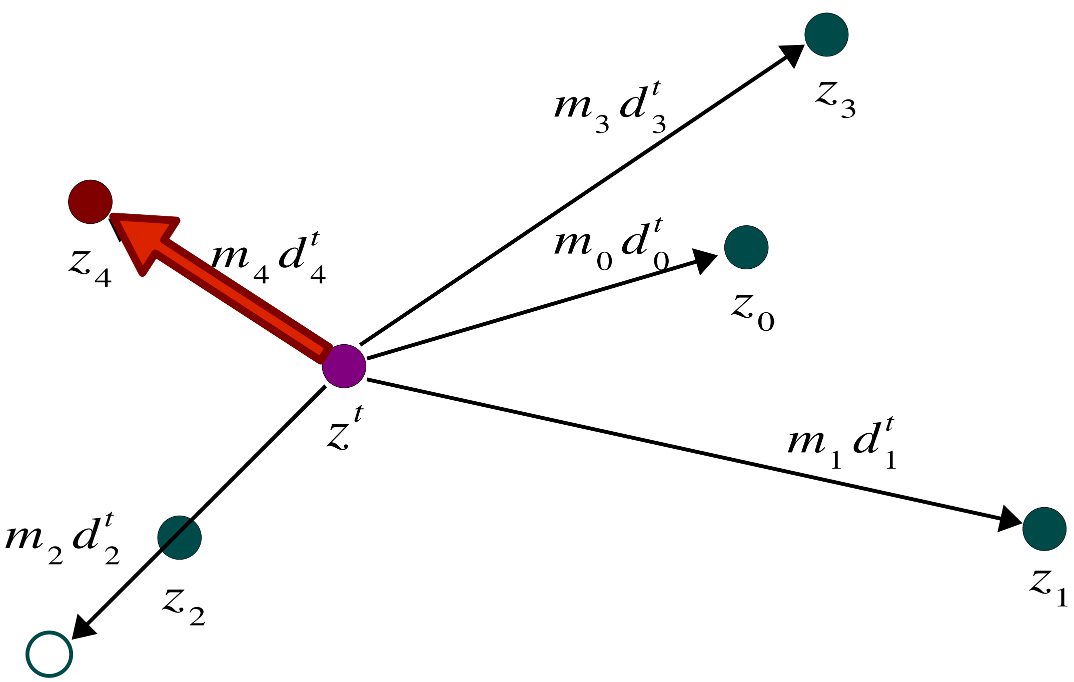


Select action $a^t = a_2$

Incorporating Advice

Scaling factor update :

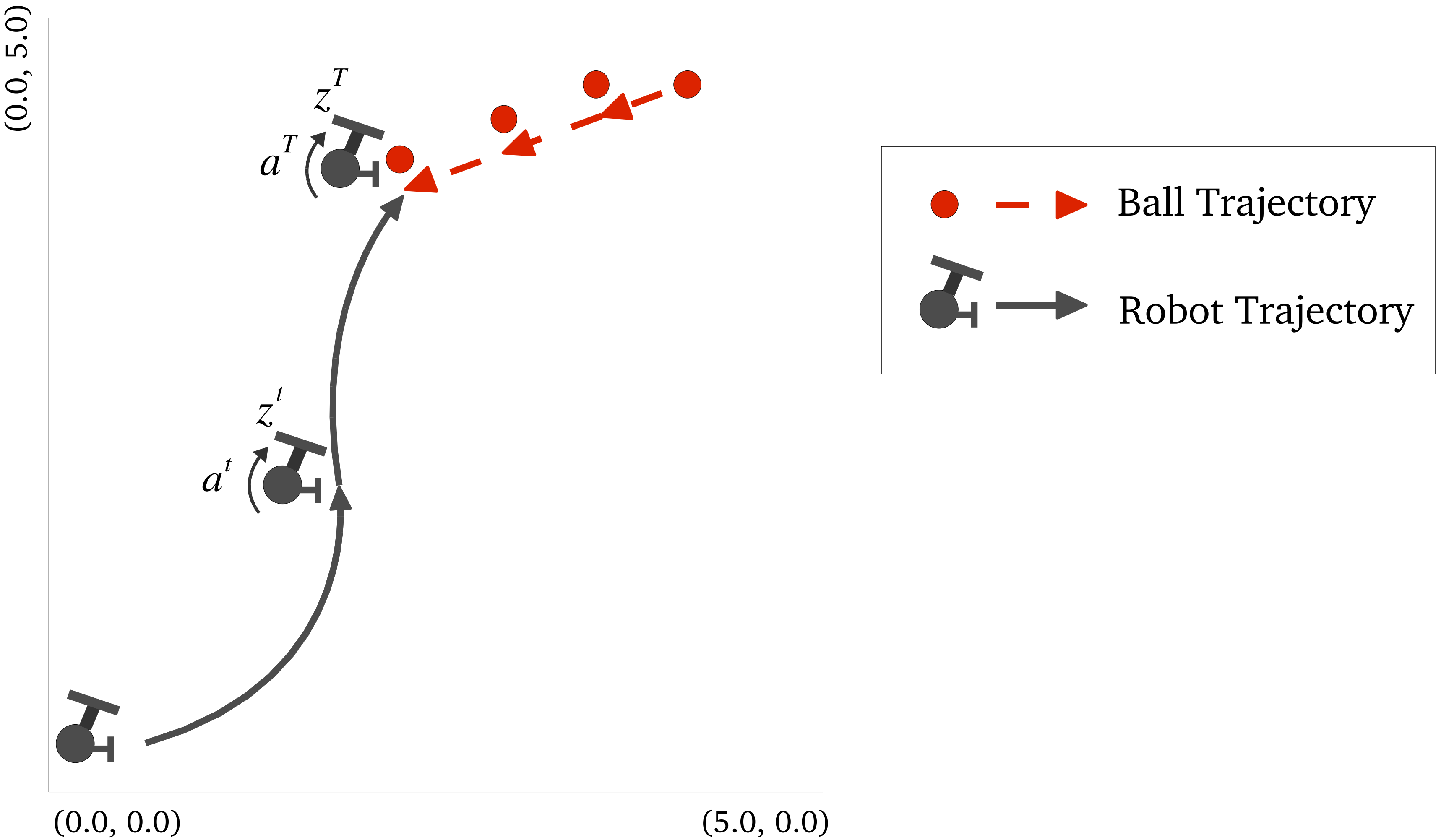
$$m_i \leftarrow \begin{cases} m_i + \kappa \cdot [D(z^t, z_i, m_i)]^{-1}, & \text{if } g = 0 \\ m_i, & \text{if } g = 1 \end{cases}$$



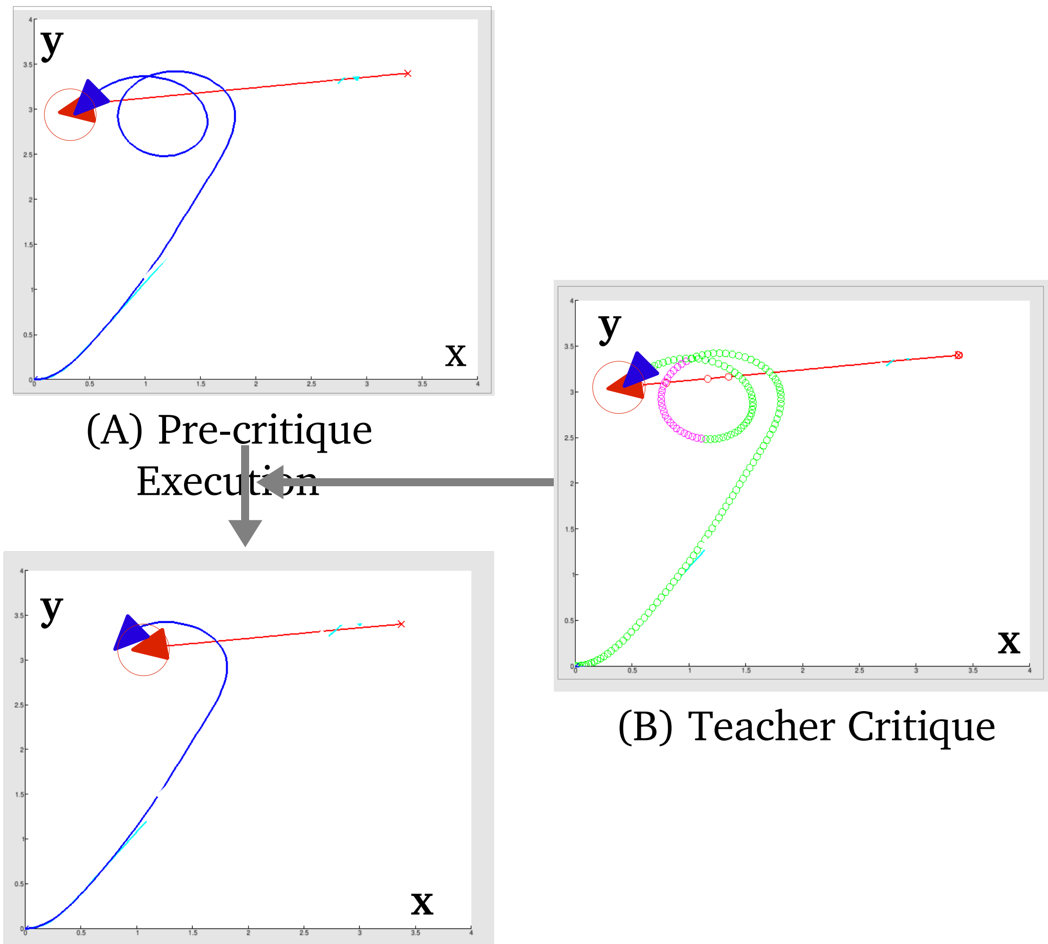
Suppose g=0, and so m_2 increases.

Select action $a^t = a_4$

Task Example



Advice Round



(A) Pre-critique Execution

(B) Teacher Critique

(C) Post-critique Execution

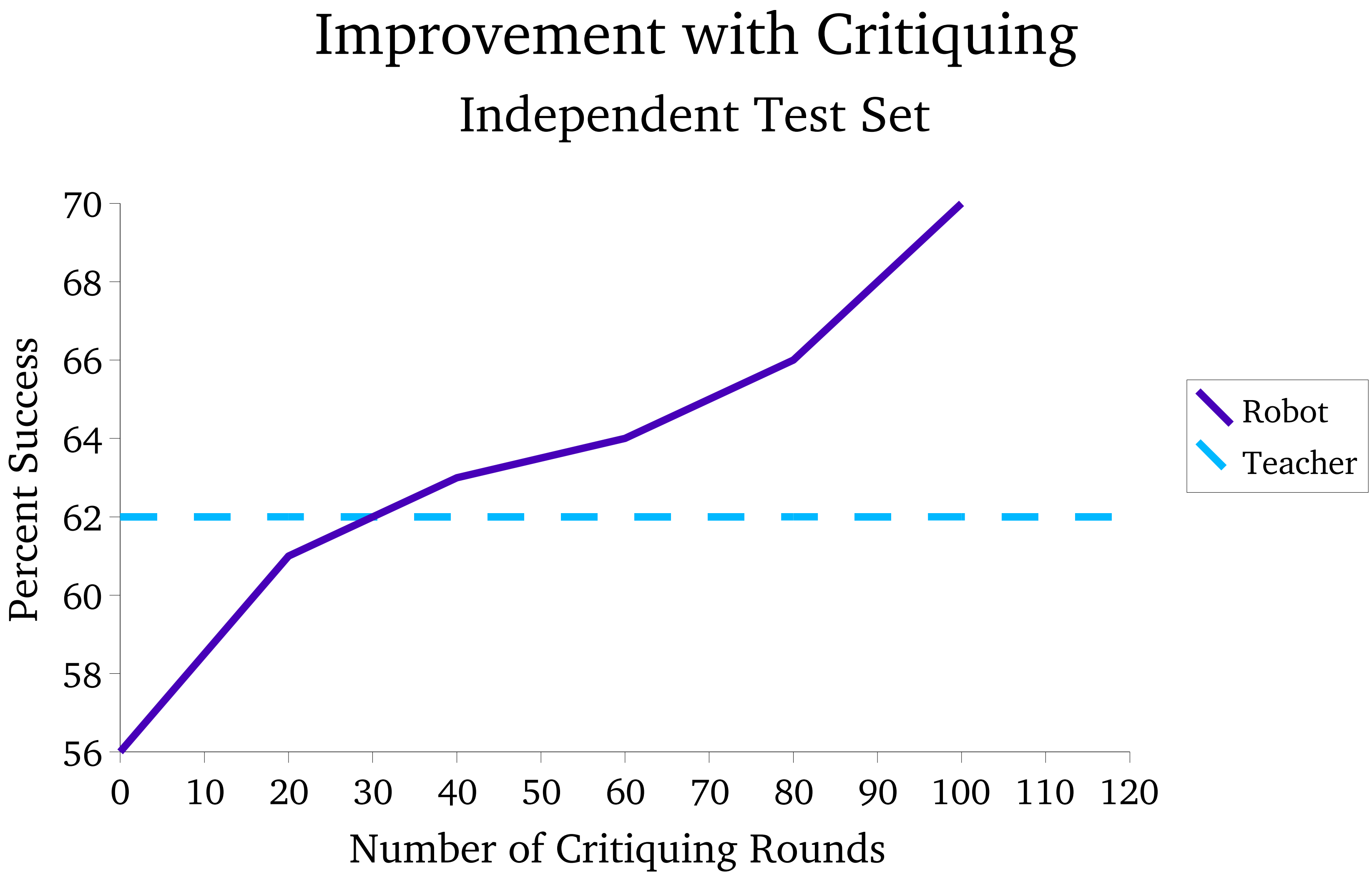
Robot Trajectory

Ball Trajectory

Poor Performance Flag (0)

Good Performance Flag (1)

Performance Results



Questions?