CSC 665: Artificial Intelligence

Reinforcement Learning II

Instructor: Pooyan Fazli
San Francisco State University

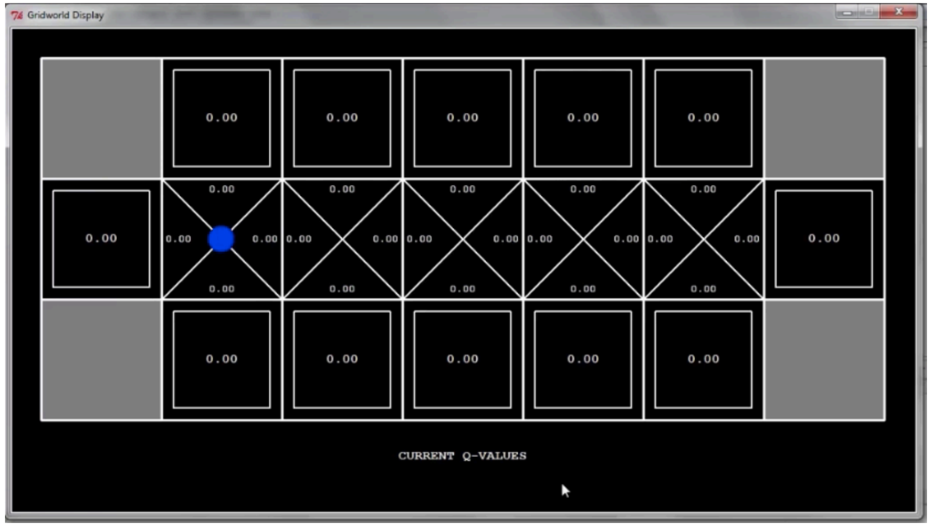
Exploitation vs. Exploration

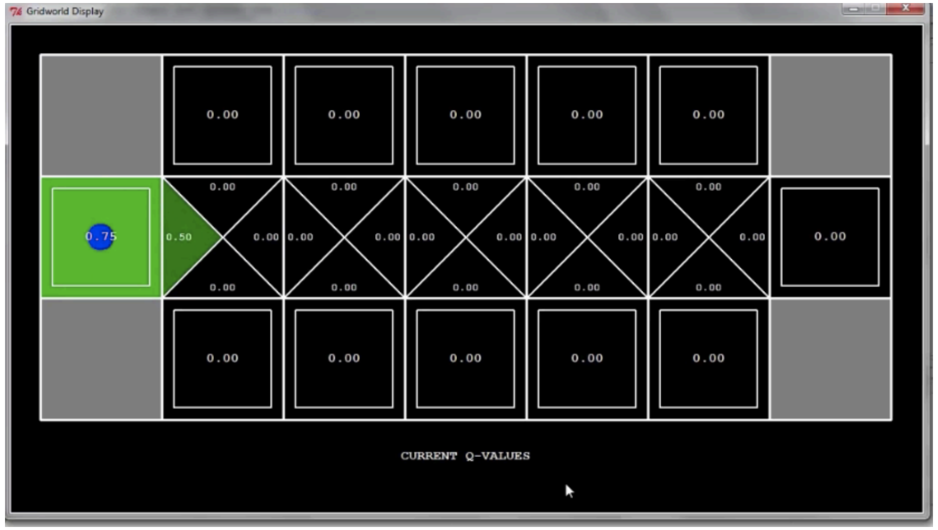
Exploitation vs. Exploration

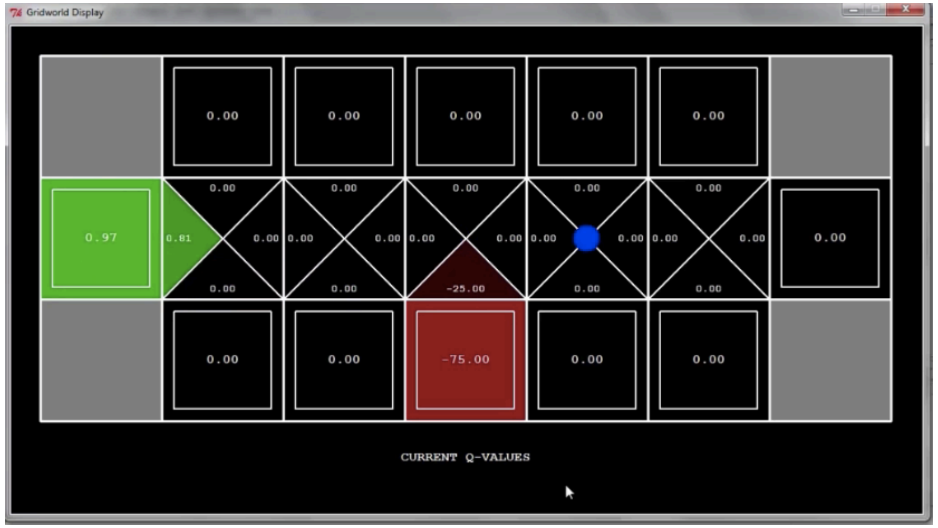
- Exploitation: Execute the current best (maybe optimal?) policy to get high payoff

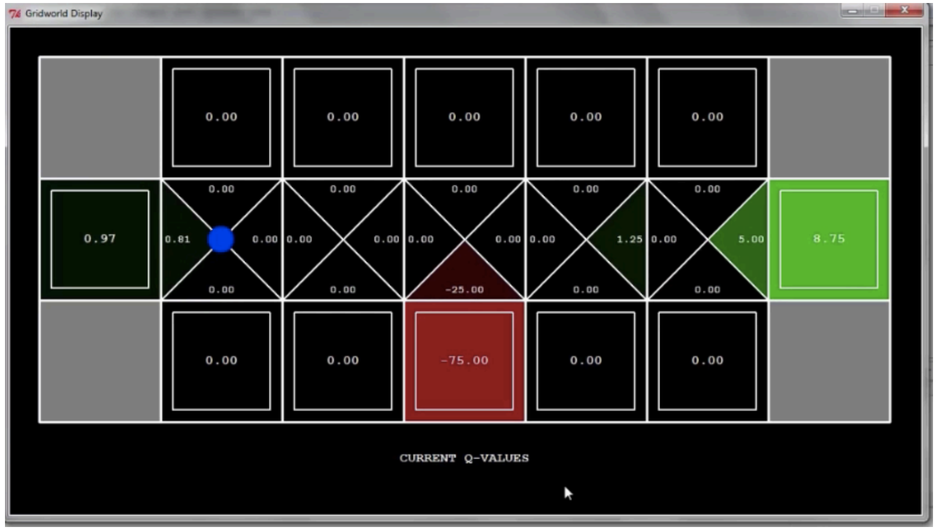
- Exploration: Try new sequences of (possibly random) actions to improve the agent’s knowledge of the environment even though current model doesn’t believe they have high payoff

Q-learning – Manual Exploration – Bridge Grid







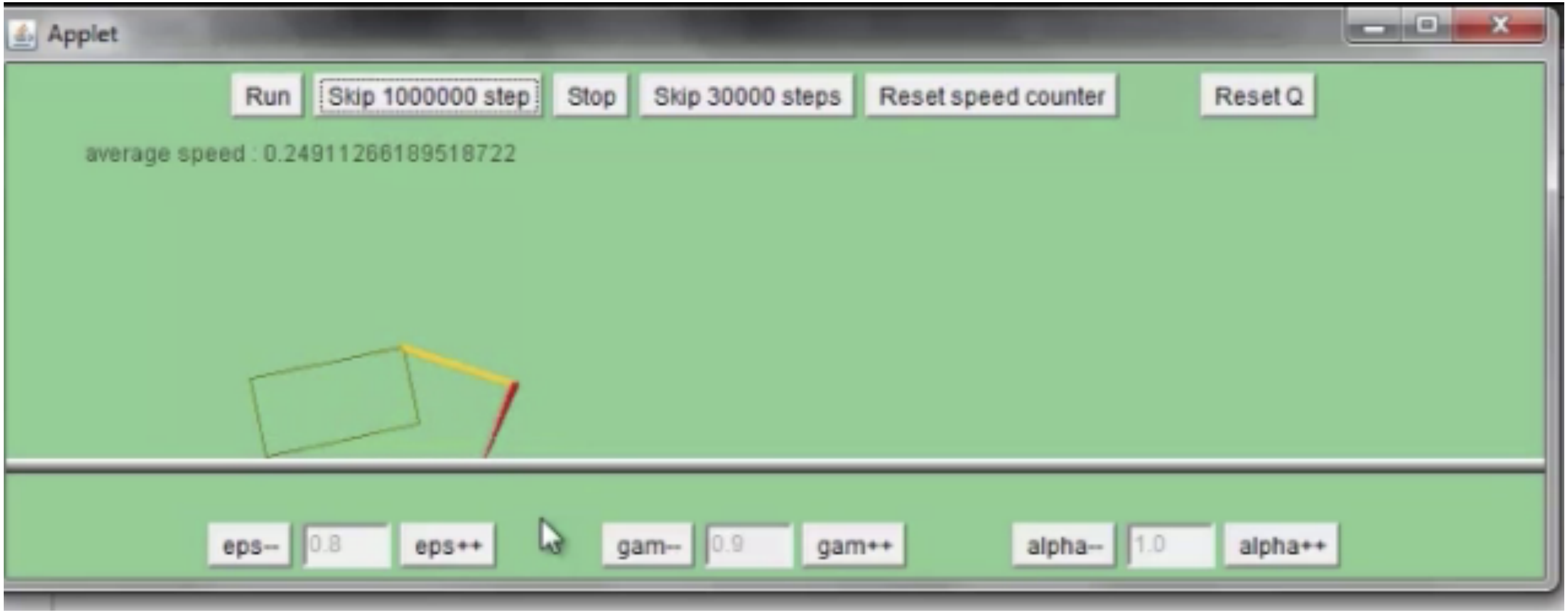


How to Explore?

- Simplest: random actions (ε-greedy)
  - Every time step, flip a coin
  - With (small) probability ε, act randomly
  - With (large) probability 1-ε, act on current best policy

- Properties of ε-greedy exploration
  - Every s,a pair is tried infinitely often
  - Does a lot of stupid things
    - Jumping off a cliff lots of times to make sure it hurts
  - Keeps doing stupid things for ever
    - Decay ε towards 0

Q-learning – Epsilon-Greedy – Crawler



Exploration Functions

- What to explore?
  - Actions towards unexplored regions are encouraged (much faster than ε-greedy!)

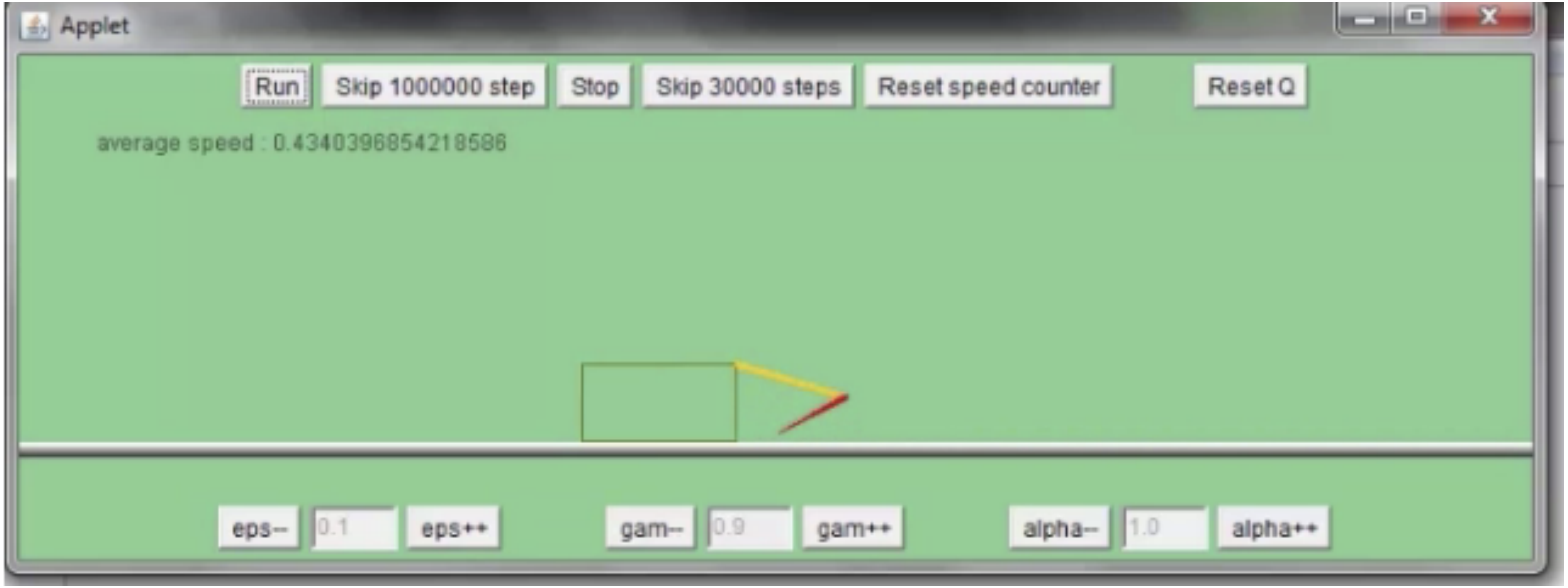
- Exploration function
  - Takes a value estimate u, a visit count n, and a constant k returns:

$$f(u,n) = u + k/n$$

Regular Q-update: $Q(s,a) \leftarrow (1-\alpha)\, Q(s,a) + \alpha\, [R(s,a,s') + \gamma \max_{a'} Q(s',a')]$

Modified Q-update: $Q(s,a) \leftarrow (1-\alpha)\, Q(s,a) + \alpha\, [R(s,a,s') + \gamma \max_{a'} f(Q(s',a'), n(s',a'))]$

Q-learning – Exploration Function – Crawler



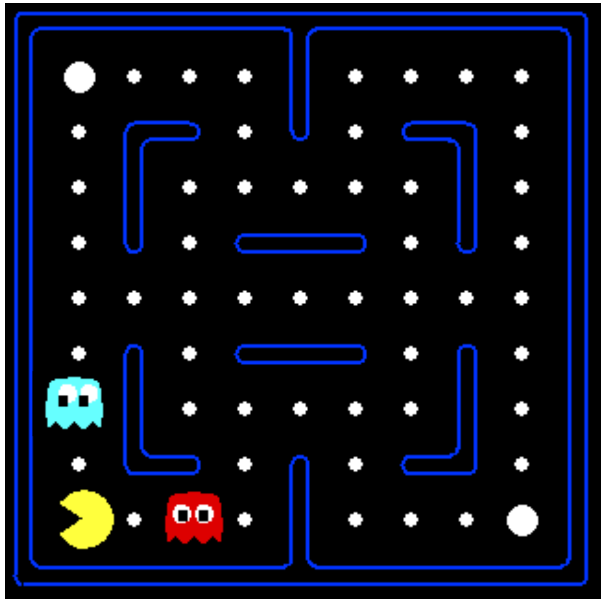
Approximate Q-Learning

Generalizing Across States

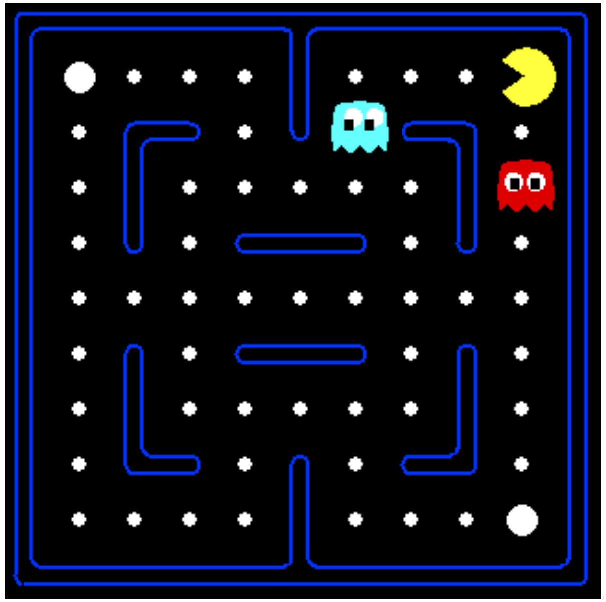
- Basic Q-Learning keeps a table of all Q-values
- In realistic situations, we cannot possibly learn about every single state!
  - Too many states to visit them all in training
  - Too many states to hold the Q-tables in memory
- Instead, we want to generalize:
  - Learn about some small number of states from experience
  - Generalize that experience to new, similar situations

Example: Pacman

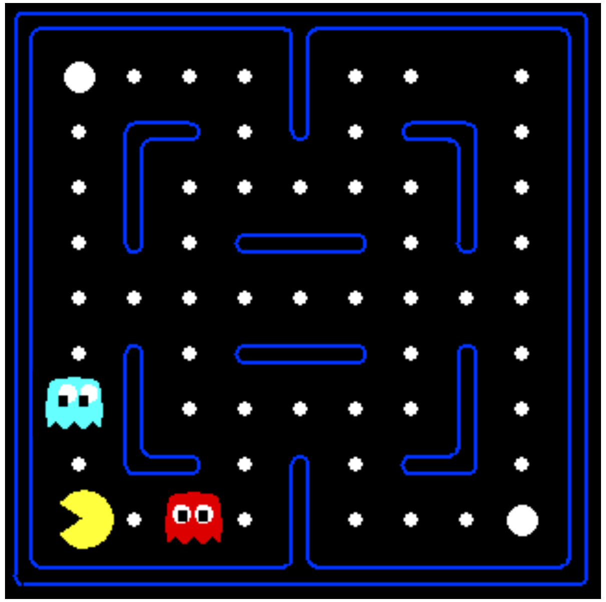
Let's say we discover through experience that this state is bad:



In naïve q-learning, we know nothing about this state:



Or even this one!

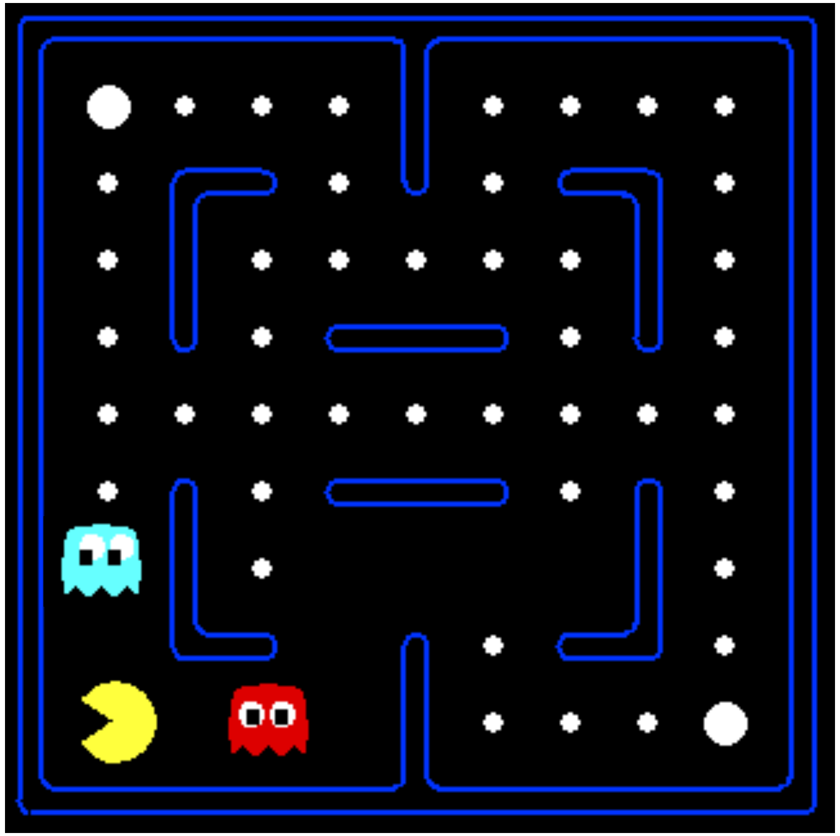


Feature-Based Representations

- Solution: describe a state using a vector of features (similar to the evaluation function in Assignment 2)

- Features are functions, f(s), from states to real numbers that capture important properties of the state
  - Example features:
    - Distance to closest ghost
    - 1 / (distance to closest ghost) f_{GST}
    - Number of ghosts
    - Distance to closest dot
    - 1 / (dist to closest dot) f_{DOT}
    - Is Pacman in a tunnel? (0/1)
    - …… etc.

- Can also describe a q-state (s, a) with features (e.g. action a in state s moves closer to food)



Linear Value Functions

- Using a feature representation, we can write the Q-value (or the V-value) for any state using a few weights:

$$V(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)$$

What you used in Assignment 2

$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \ldots + w_n f_n(s,a)$$

What you use in Approximate Q-learning

Approximate Q-Learning

$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \ldots + w_n f_n(s,a)$$

- Q-learning with linear functions:

transition $= (s, a, r, s')$

difference $= \left[r + \gamma \max_{a'} Q(s', a')\right] - Q(s,a)$

$$Q(s,a) \leftarrow (1-\alpha) Q(s,a) + (\alpha) \left[r + \gamma \max_{a'} Q(s', a')\right]$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha\,[\text{difference}]$$

Q-Learning

What you saw before

$$w_i \leftarrow w_i + \alpha\,[\text{difference}]\, f_i(s,a)$$

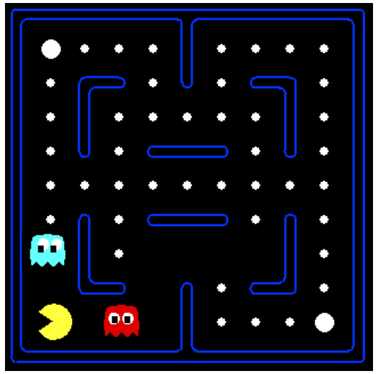
Approximate Q-Learning

- Intuitive interpretation:
  - Adjust weights of features

Example: Q-Pacman

$$Q(s,a) = 4.0 f_{DOT}(s,a) - 1.0 f_{GST}(s,a)$$

s

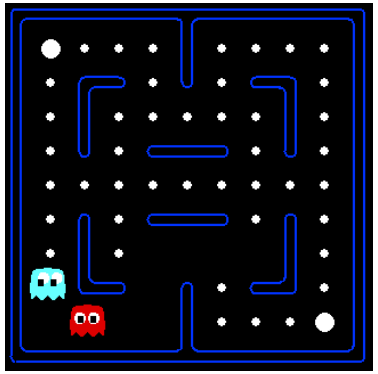


$f_{DOT}(s, \text{NORTH}) = 0.5$

$f_{GST}(s, \text{NORTH}) = 1.0$

$a = \text{NORTH}$
$r = -500$

s'



$Q(s, \text{NORTH}) = +1$

$r + \gamma \max_{a'} Q(s', a') = -500 + 0$

$Q(s', \cdot) = 0$

difference $= -501$

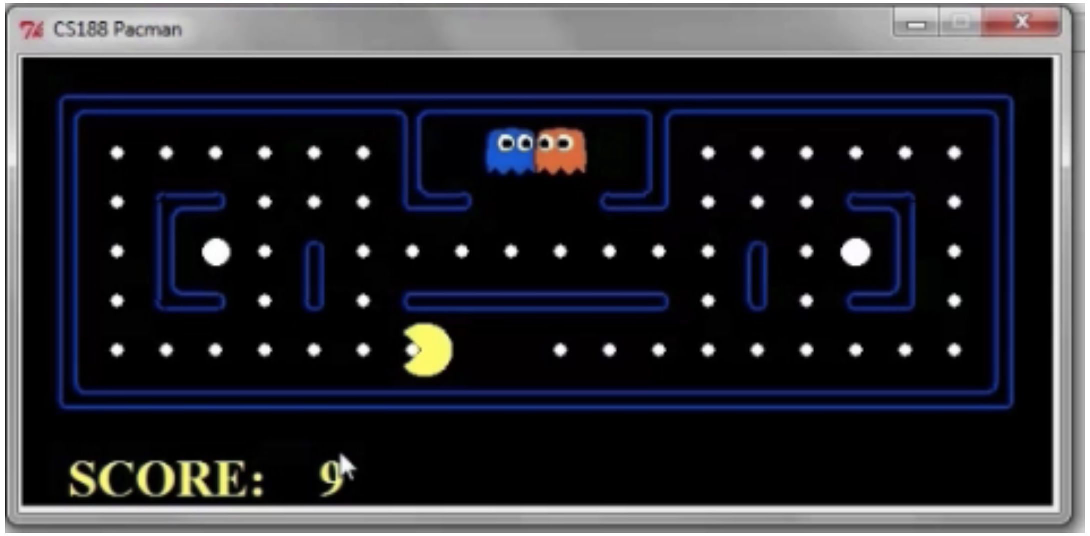
$w_{DOT} \leftarrow 4.0 + \alpha\,[-501]\,0.5$

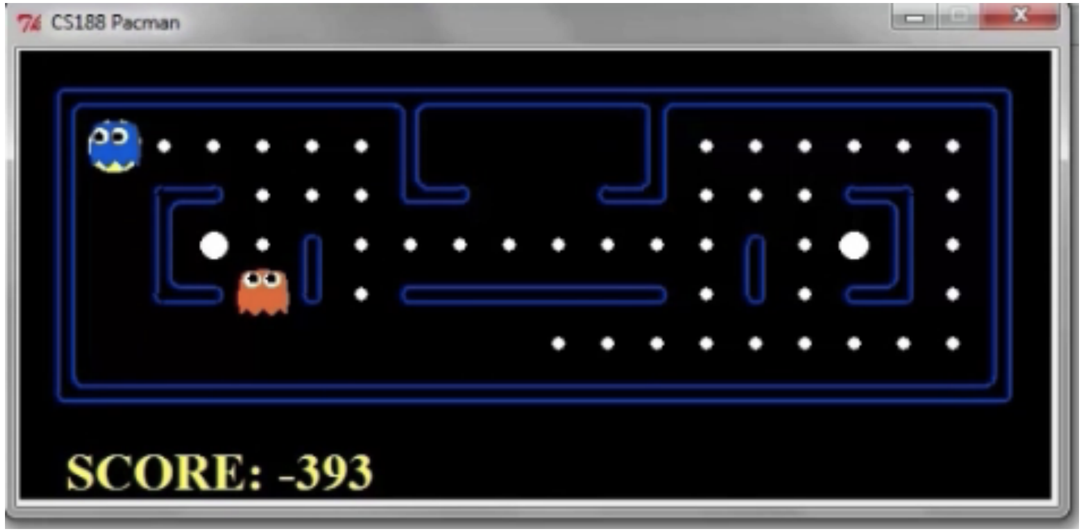
$w_{GST} \leftarrow -1.0 + \alpha\,[-501]\,1.0$

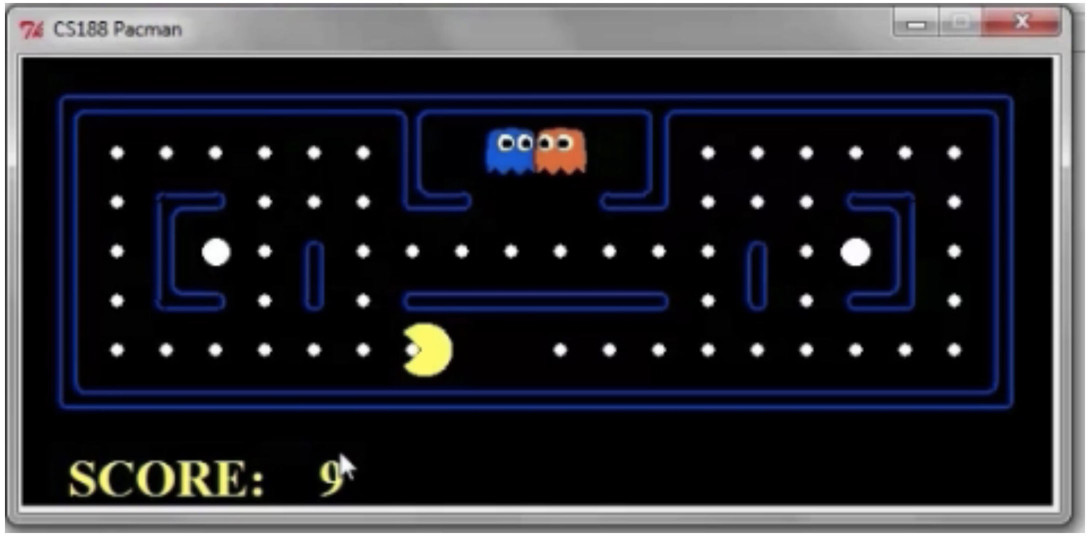
$$Q(s,a) = 3.0 f_{DOT}(s,a) - 3.0 f_{GST}(s,a)$$

alpha = 0.004

Approximate Q-Learning -- Pacman







Conclusion

- We’ve seen how AI methods can solve problems in:
  - Search
  - Constraint Satisfaction Problems
  - Games
  - Markov Decision Problems
  - Reinforcement Learning
- Next up: Uncertainty and Learning!

Reading

- Read Section 22.4 in the AIMA textbook