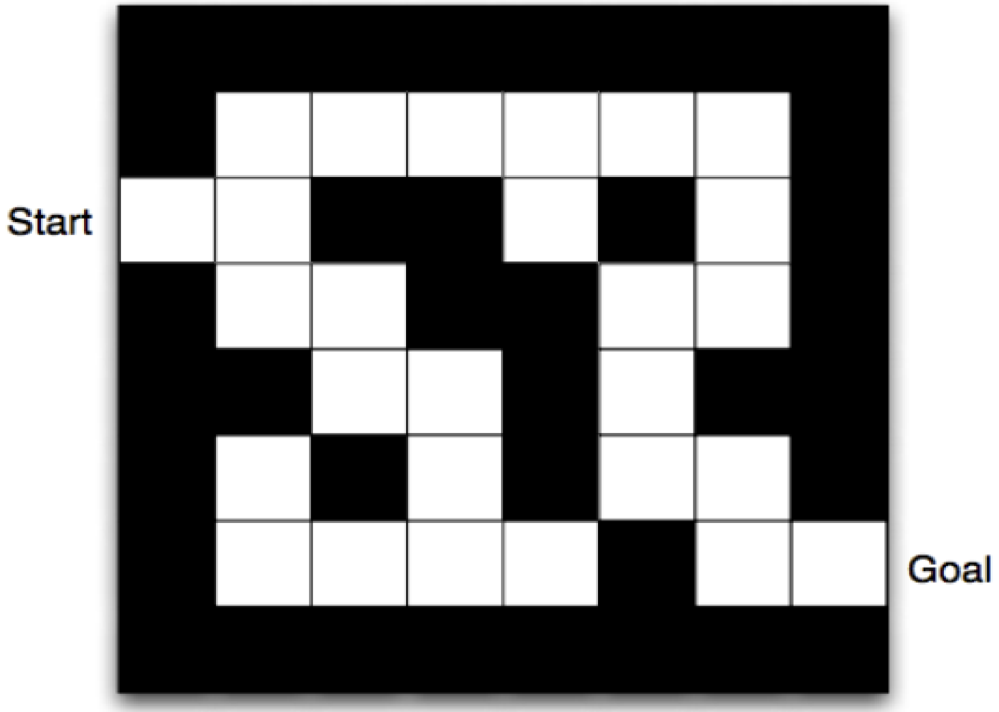
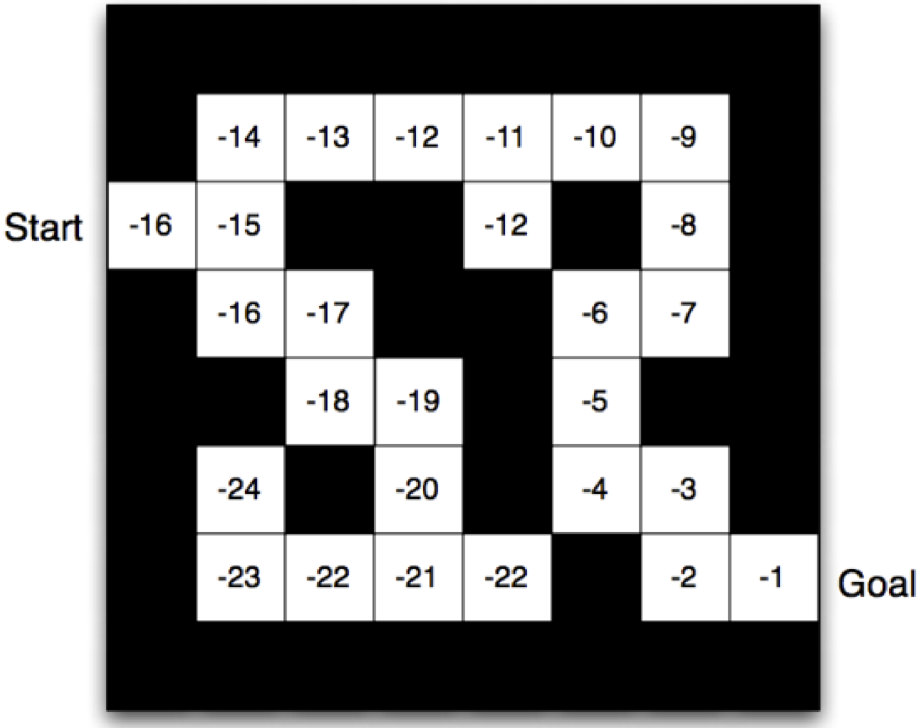
Markov Decision Process Examples

Example I: Maze Problem as MDP



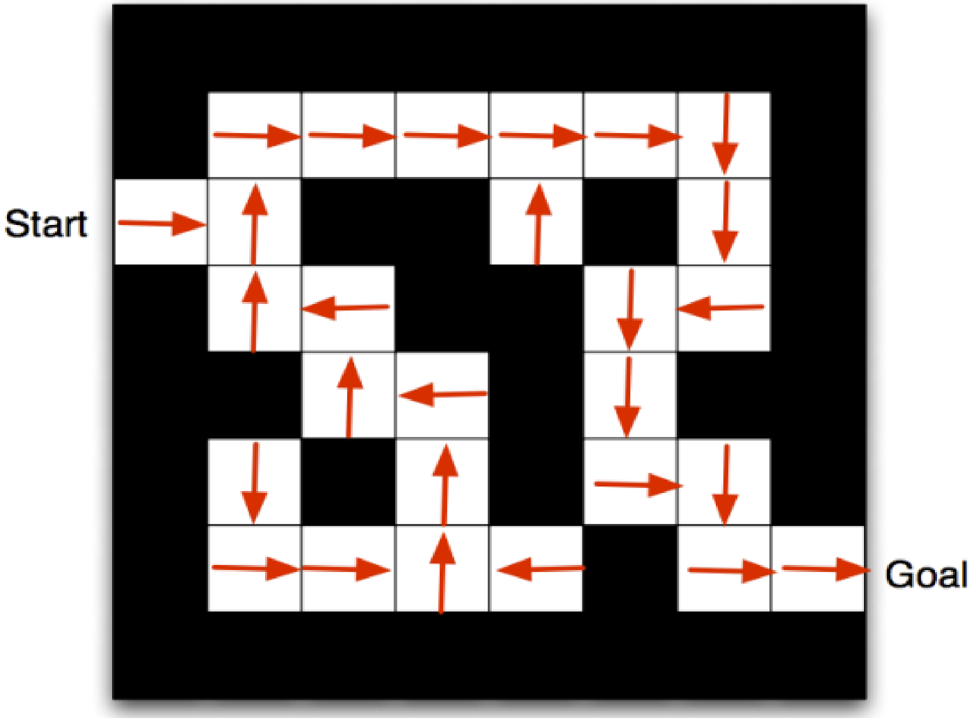
- Rewards: -1 per time-step
- Actions: N, E, S, W
- States: Agent's location

Maze Problem as MDP



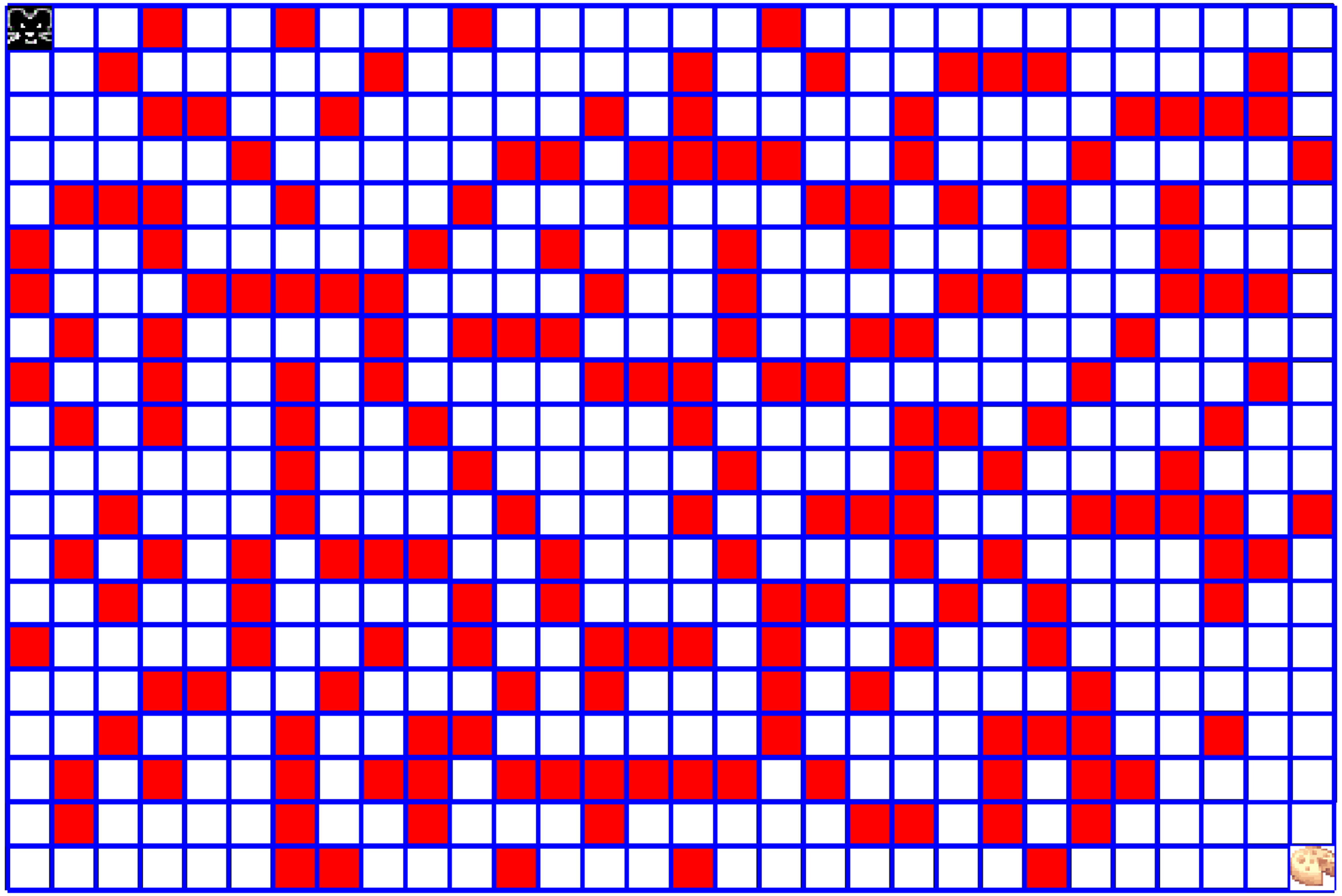
- Numbers represent value $V^{\pi}(s)$ of each state s

Maze Problem as MDP



- Arrows represent policy $\pi(s)$ for each state s

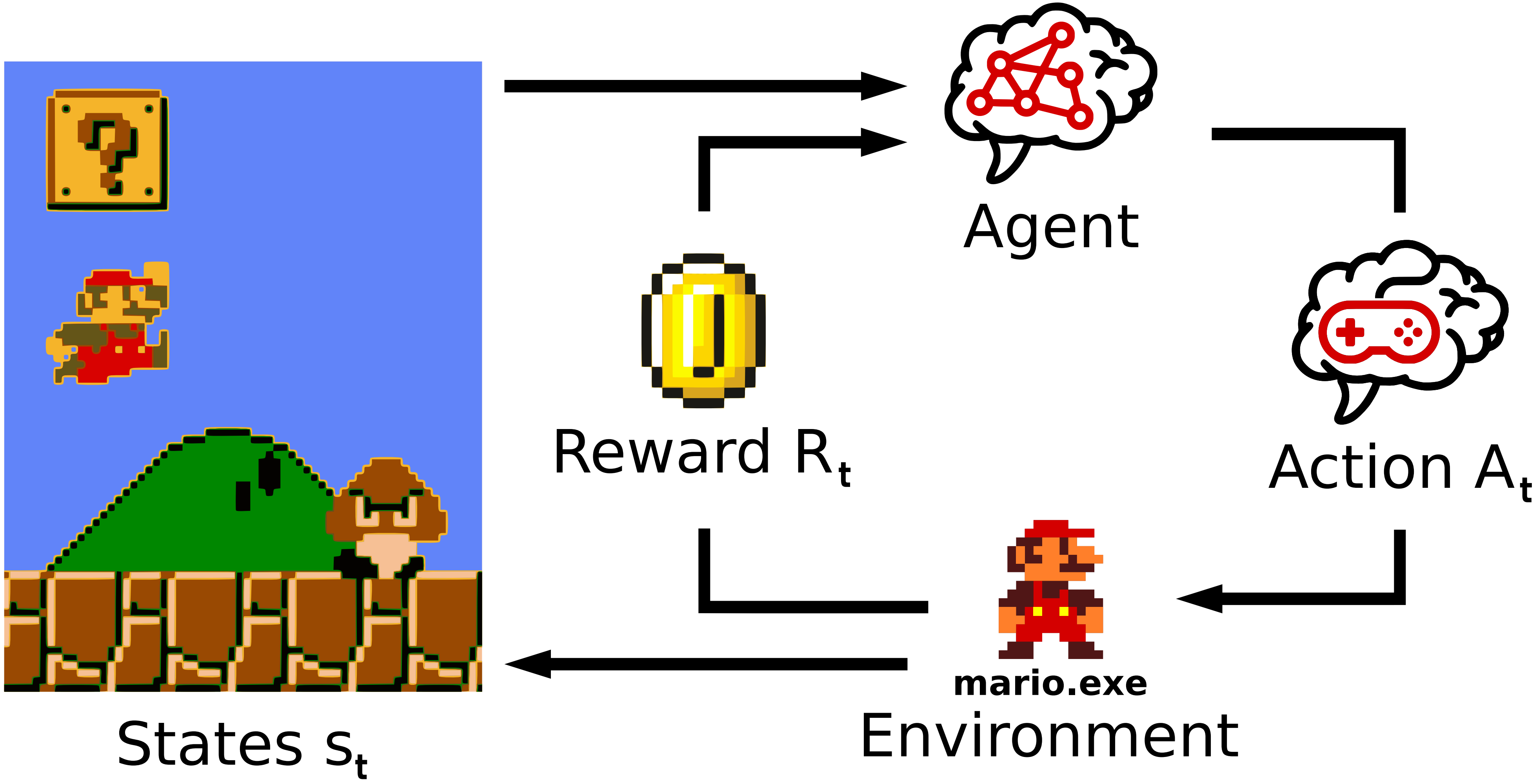
Maze Problem as MDP



https://www.samyzaf.com/ML/rl/qmaze.html

Example II: Super Mario Bros. Game

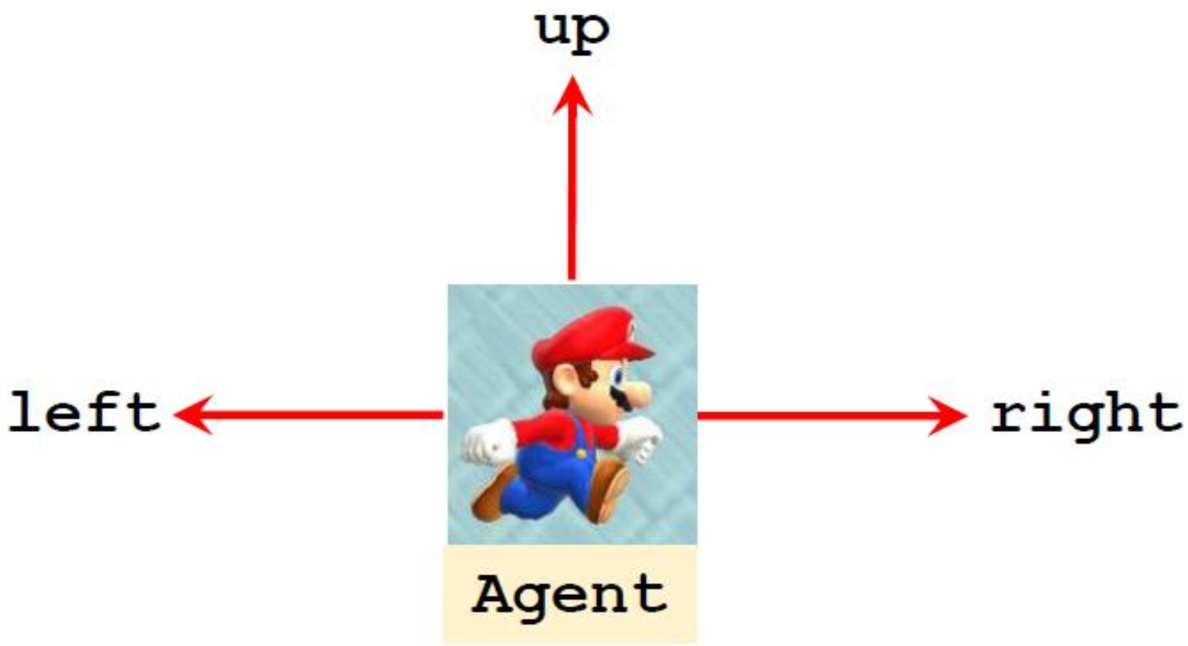
Super Mario Bros. Game as MDP



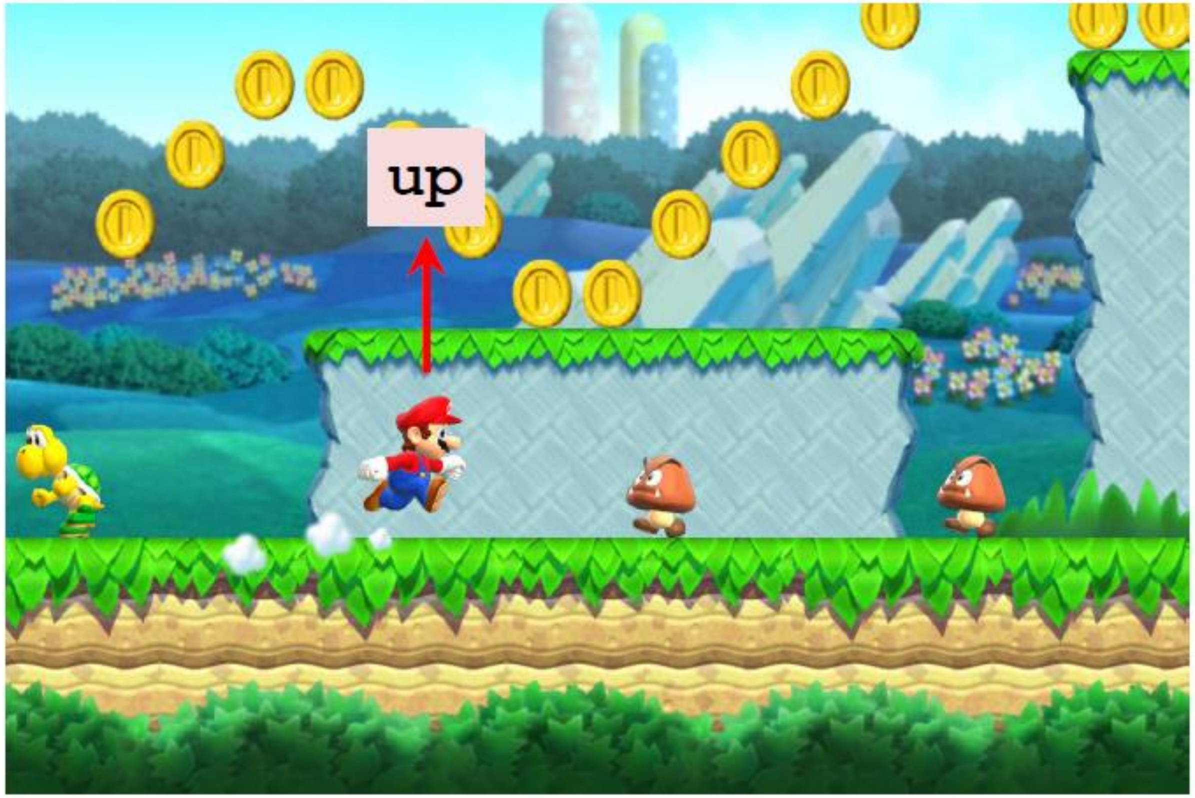
Terminology: state and action

state s (this frame)

Action $a \in \{\text{left}, \text{right}, \text{up}\}$



Terminology: state transition



state transition

old state $\xrightarrow{\text{action}}$ new state

- E.g., "up" action leads to a new state.

- State transition can be random.
- Randomness is from the environment.

Terminology: state transition

state transition

old state $\xrightarrow{\text{action}}$ new state

- E.g., "up" action leads to a new state.

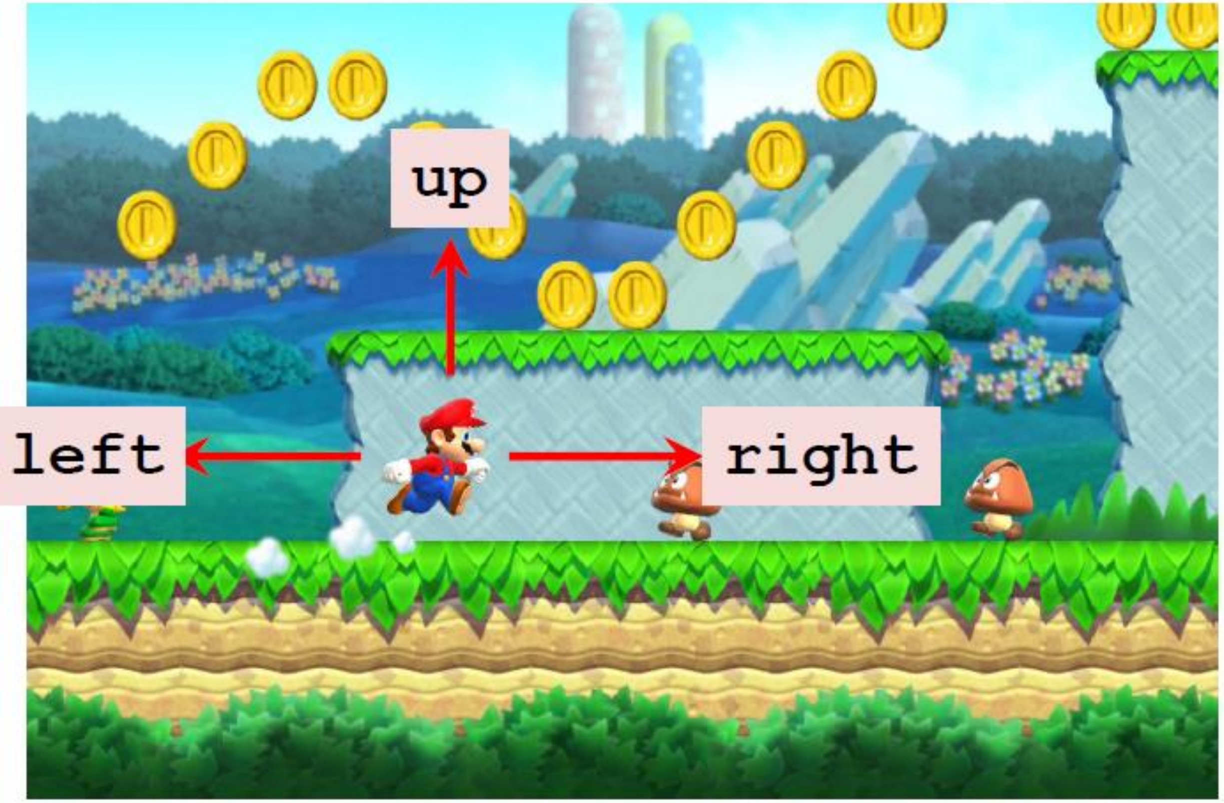
- State transition can be random.
- Randomness is from the environment.
- $p(s'|s,a) = \mathbb{P}(S' = s'|S = s, A = a)$.

Terminology: reward

reward R

- Collect a coin: $R = +1$
- Win the game: $R = +10000$
- Touch a Goomba: $R = -10000$ (game over).
- Nothing happens: $R = 0$

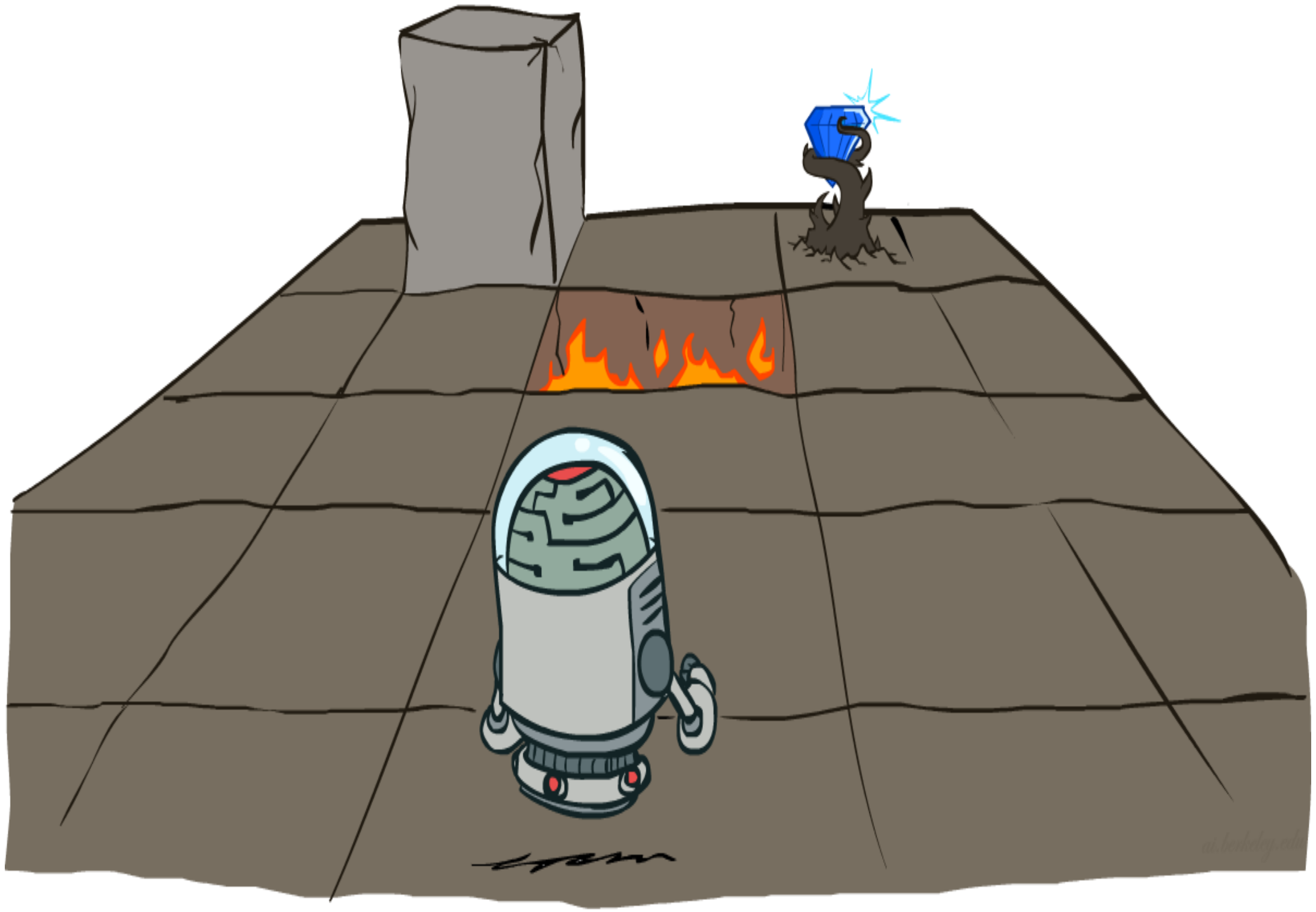
Terminology: policy

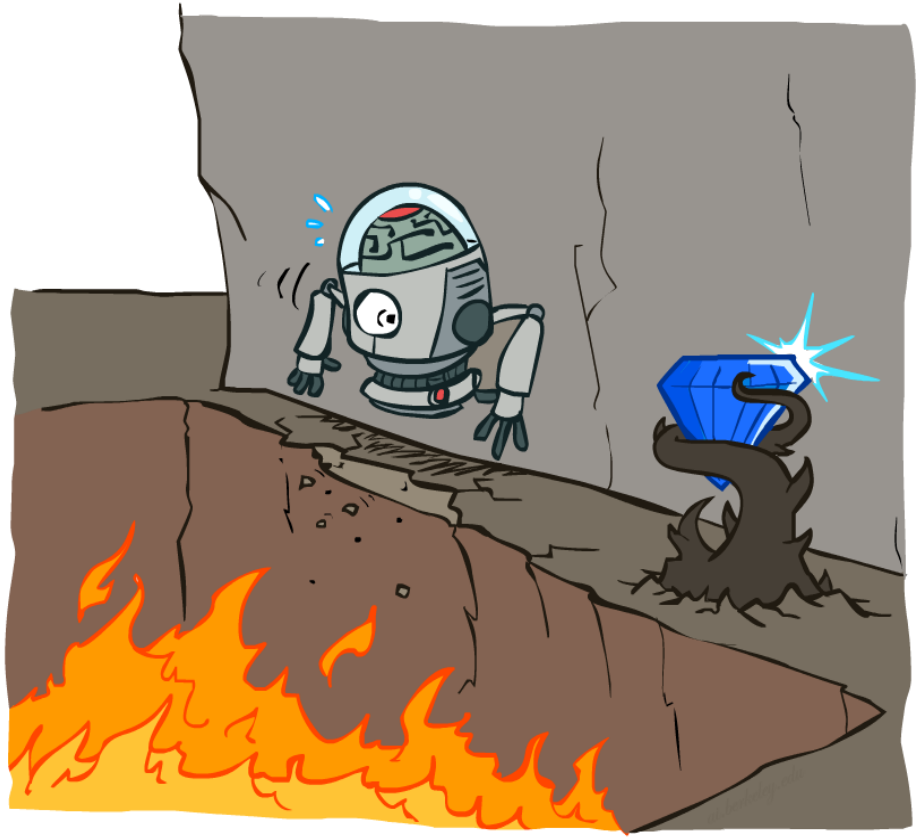


policy π

- $\pi(a \mid s)$ is the probability of taking action $A = a$ given state s, e.g.,
  - $\pi(\text{left} \mid s) = 0.2,$
  - $\pi(\text{right} \mid s) = 0.1,$
  - $\pi(\text{up} \mid s) = 0.7.$
- Upon observing state $S = s$, the agent's action A can be random.

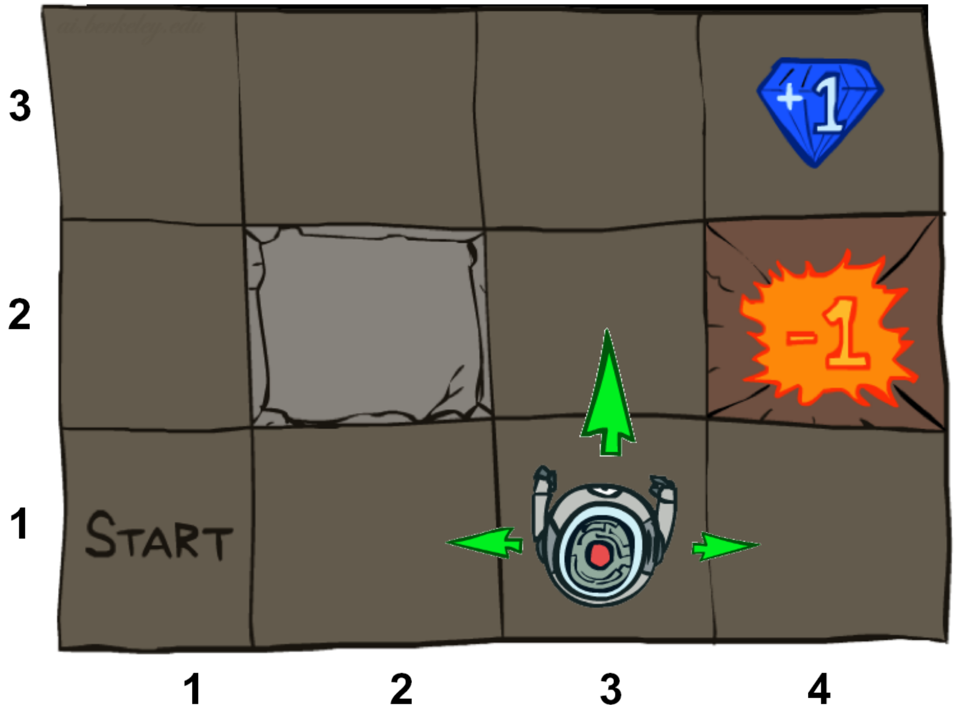
Example III: Robot in Grid World





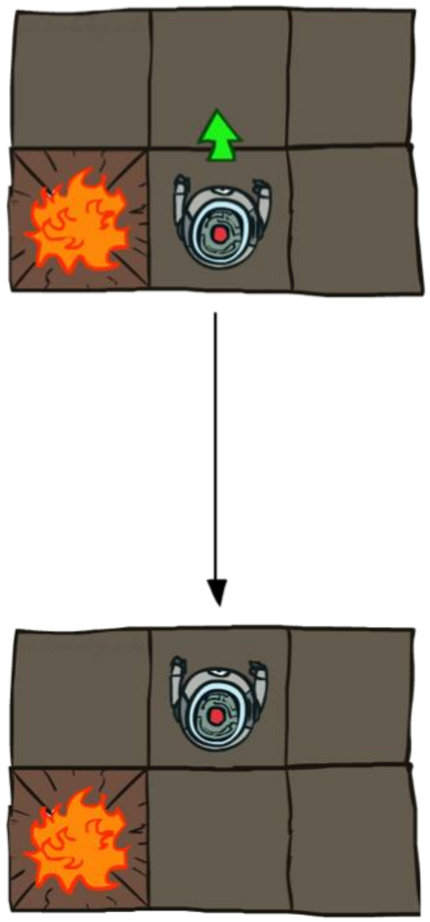
Example: Robot in Grid World

- A maze-like problem
  - The agent lives in a grid
  - Walls block the agent's path
- Noisy movement: actions do not always go as planned
  - 80% of the time, the action North takes the agent North (if there is no wall there)
  - 10% of the time, North takes the agent West; 10% East
  - If there is a wall in the direction the agent would have been taken, the agent stays put
- The agent receives rewards each time step
  - Small "living" reward each step (can be negative)
  - Big rewards come at the end (good or bad)
- Goal: maximize sum of rewards

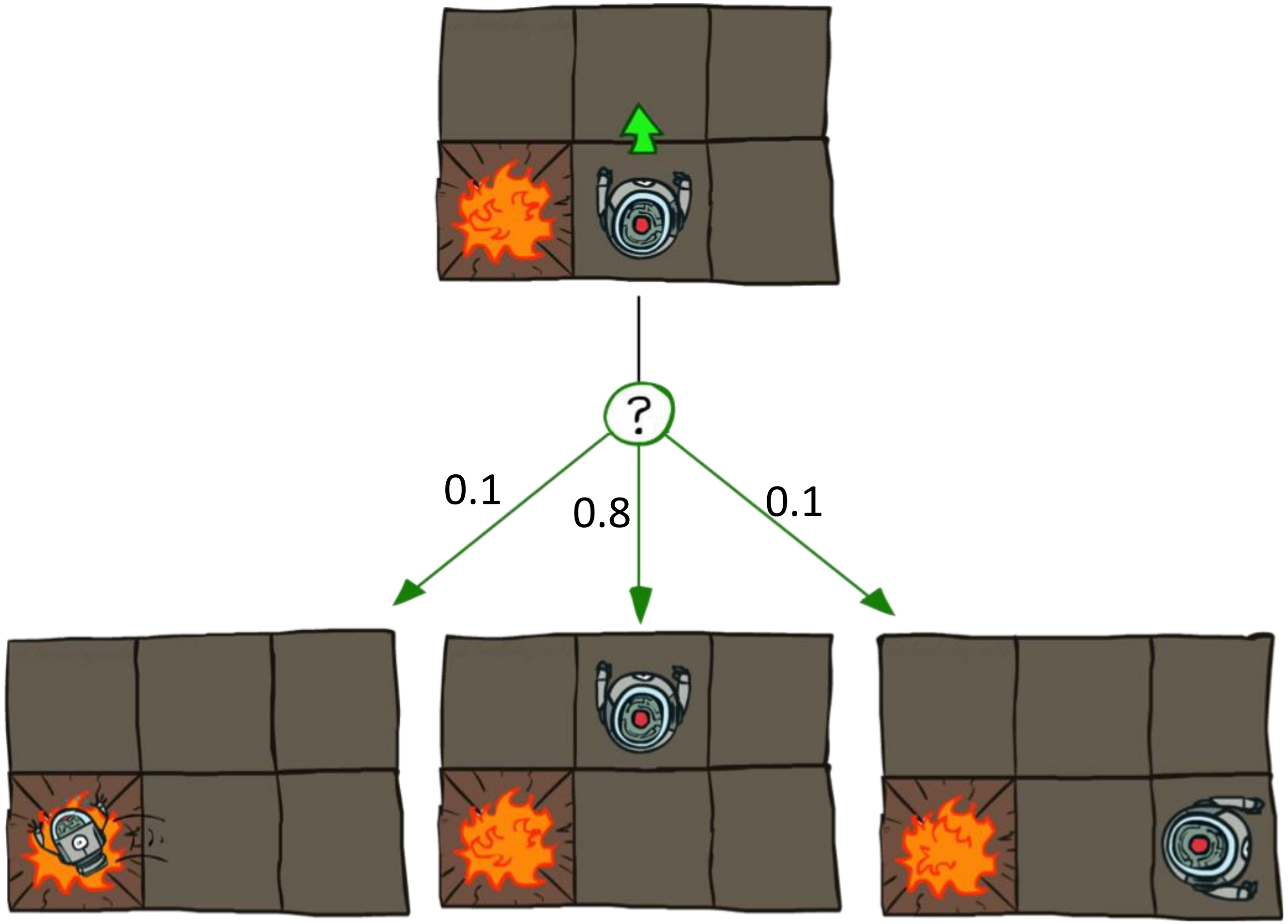


Grid World Actions

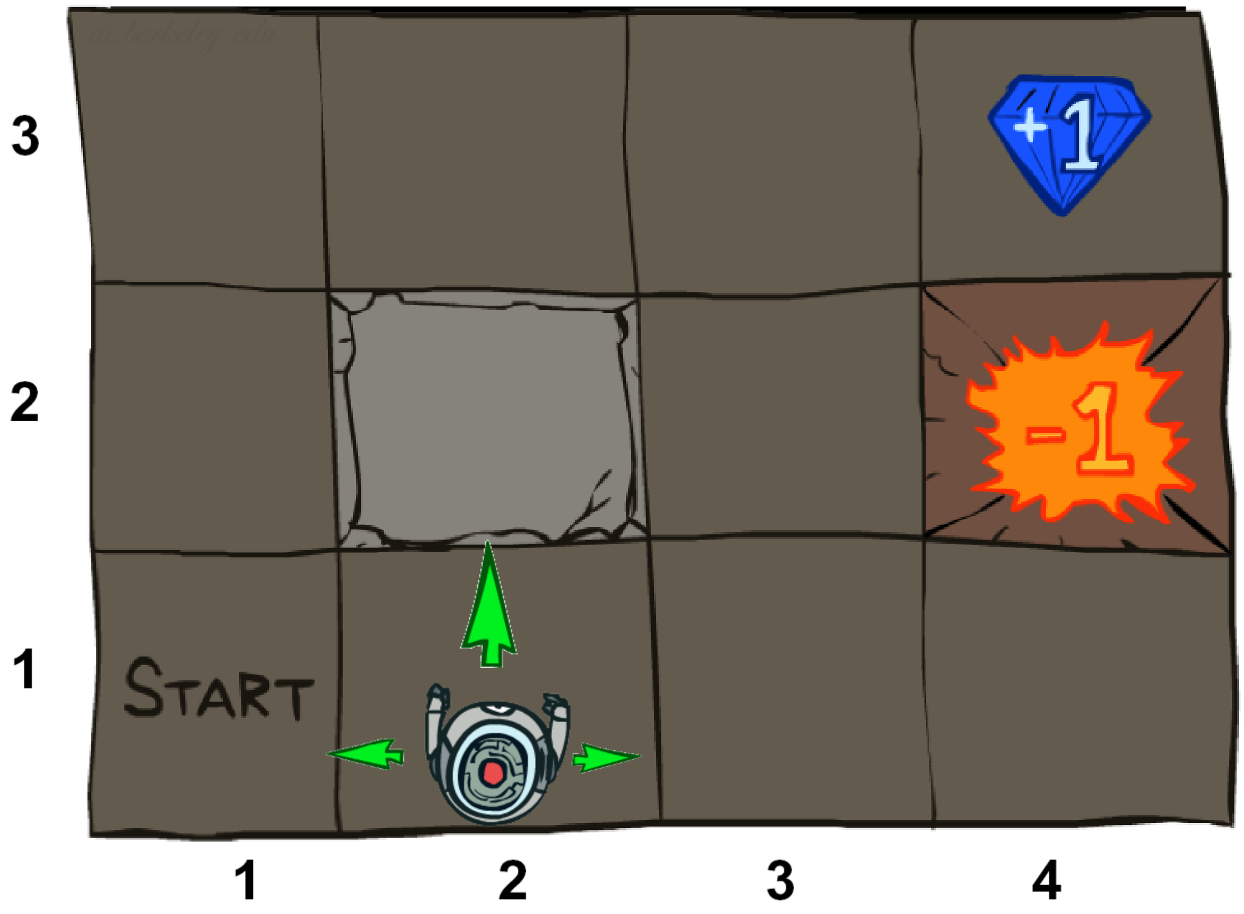
Deterministic Grid World



Stochastic Grid World

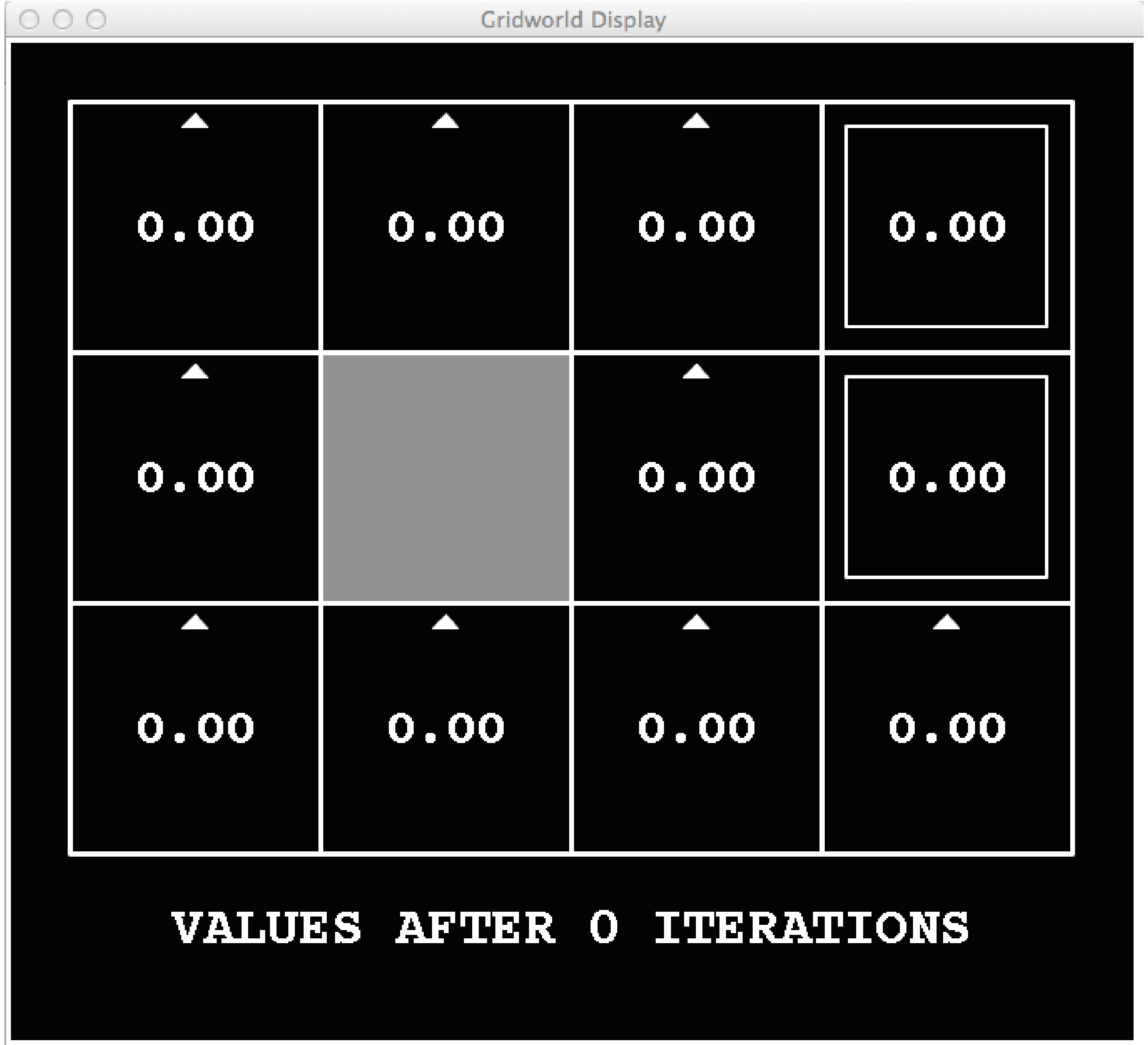


Markov Decision Process



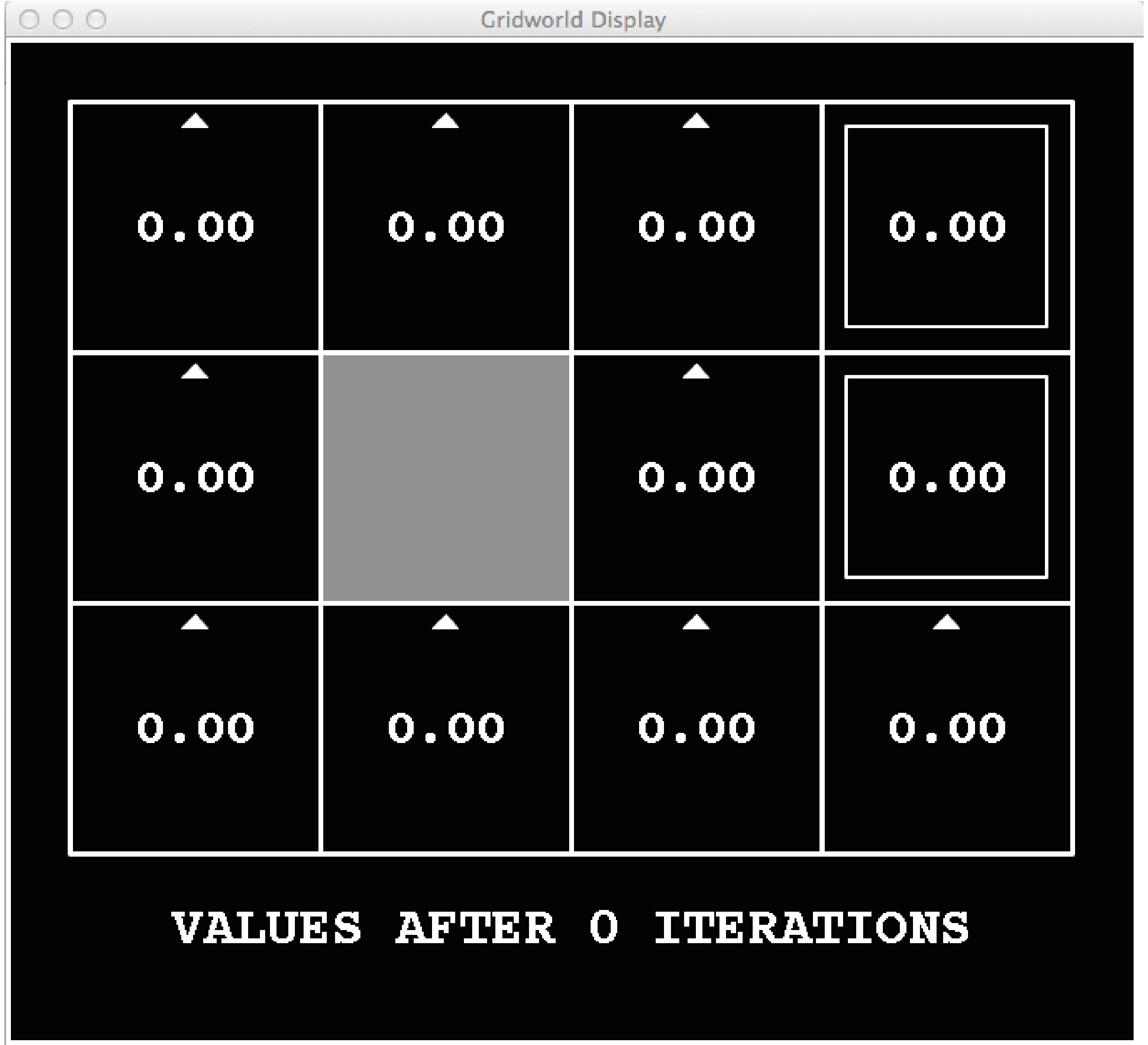
- States: Positions of Robots
- Actions : Movements
- State Transitions : Uncertain resulting state for the chosen direction
- Rewards : small living reward in non-terminate states; big rewards of +1 or -1 in terminate states

Value Iteration



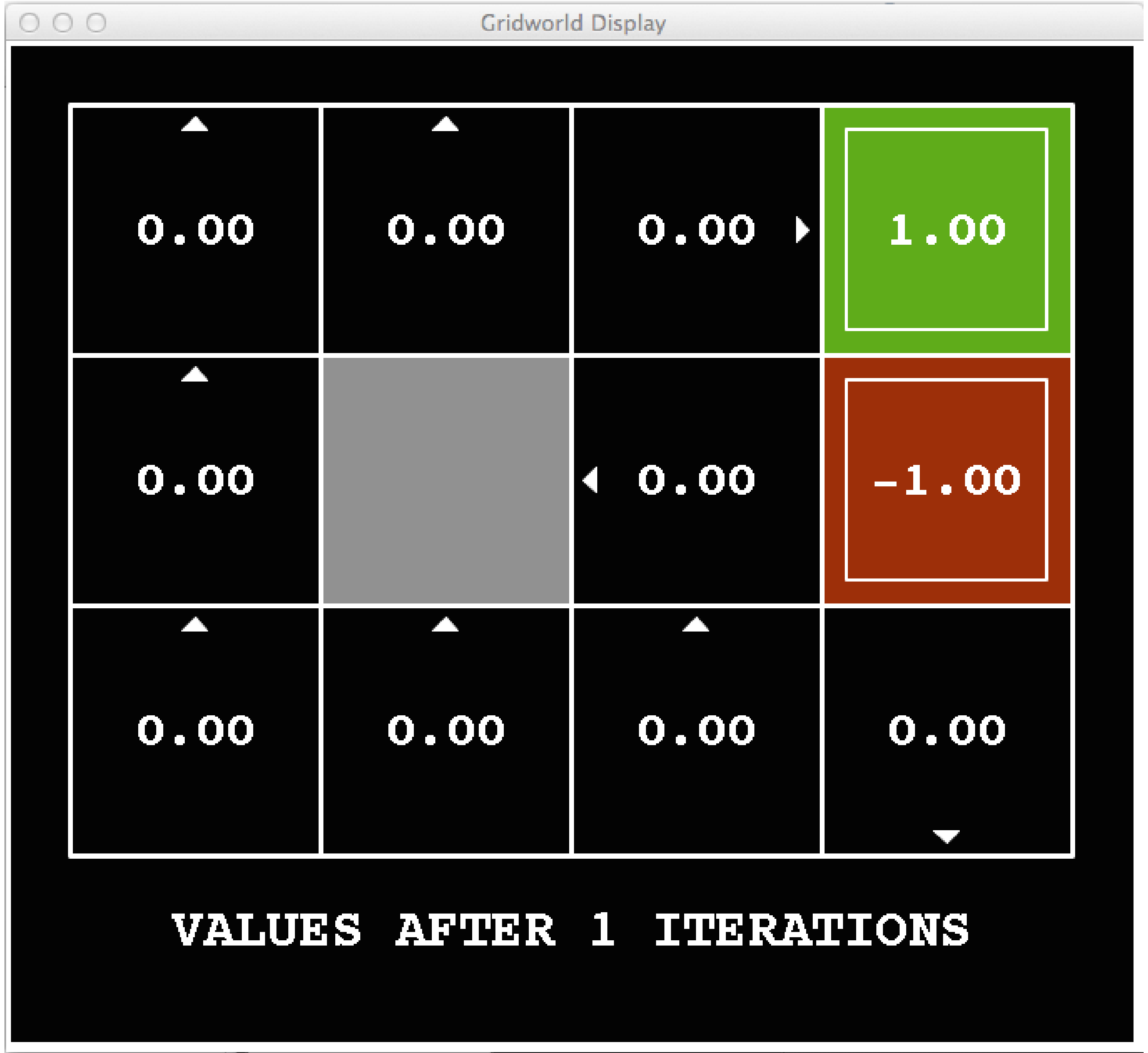
Noise = 0.2
Discount = 0.9
Living reward = 0

k=0



Noise = 0.2
Discount = 0.9
Living reward = 0

k=1



Noise = 0.2
Discount = 0.9
Living reward = 0

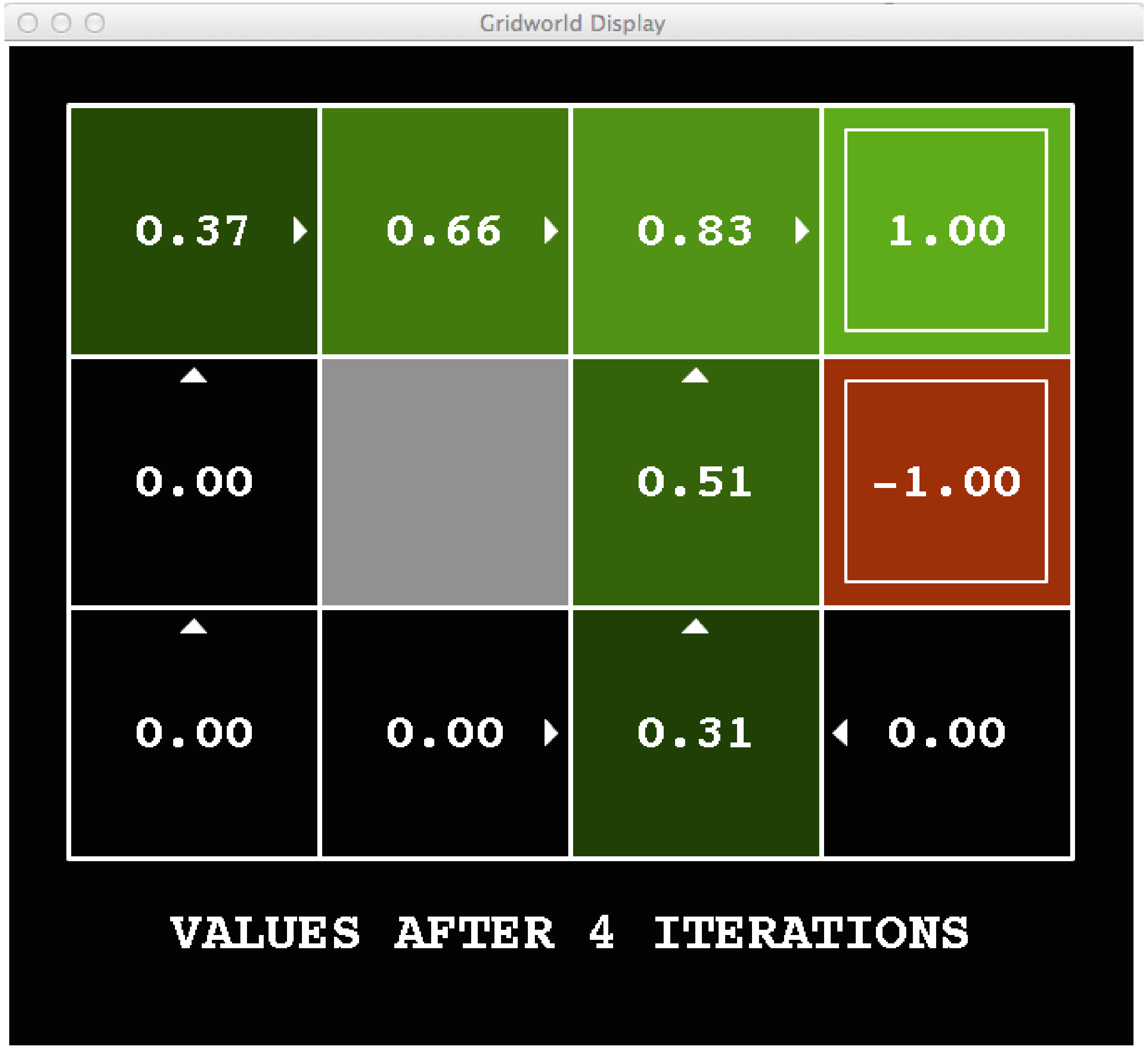
k=2

Noise = 0.2
Discount = 0.9
Living reward = 0

k=3

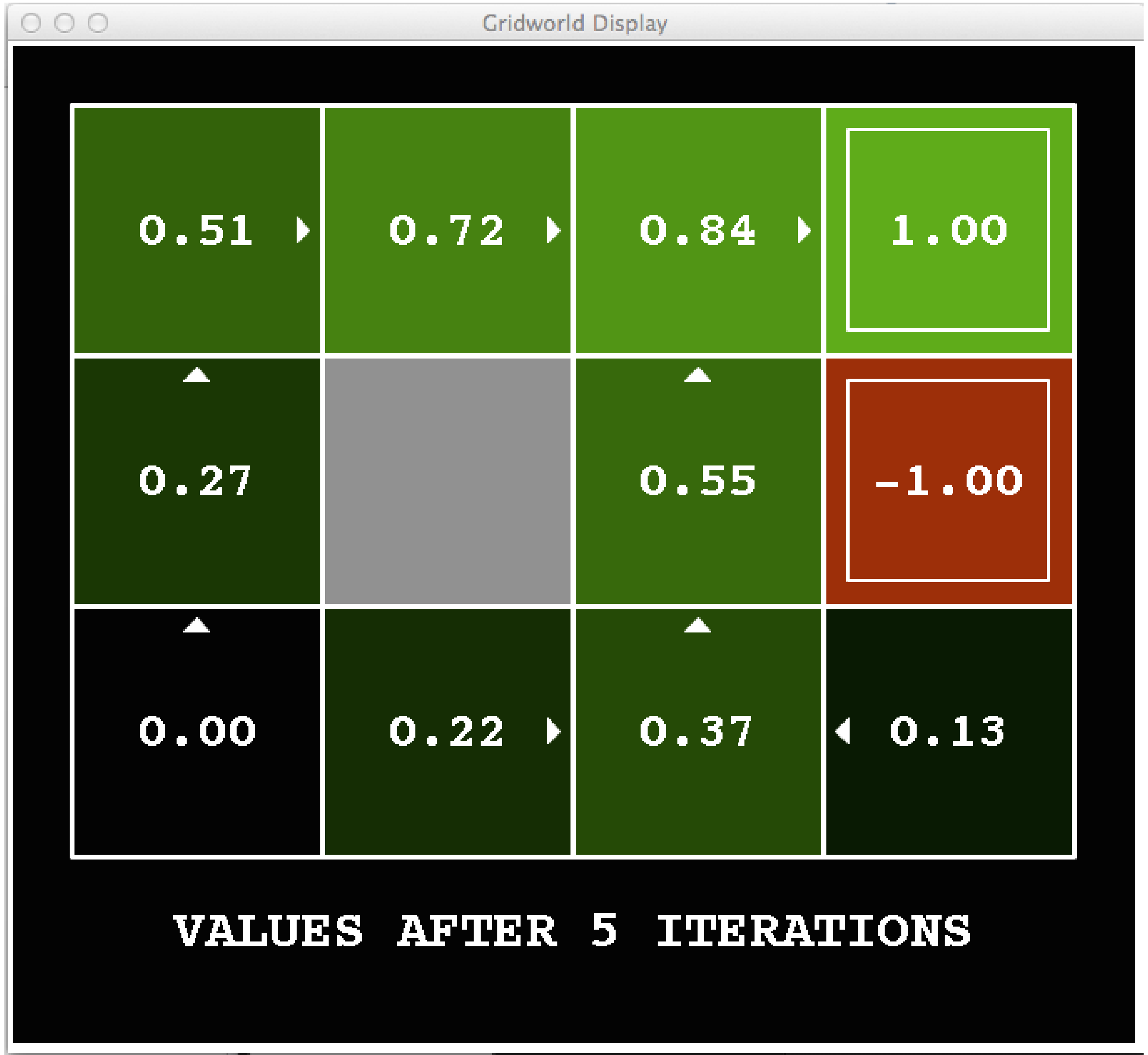
Noise = 0.2
Discount = 0.9
Living reward = 0

k=4



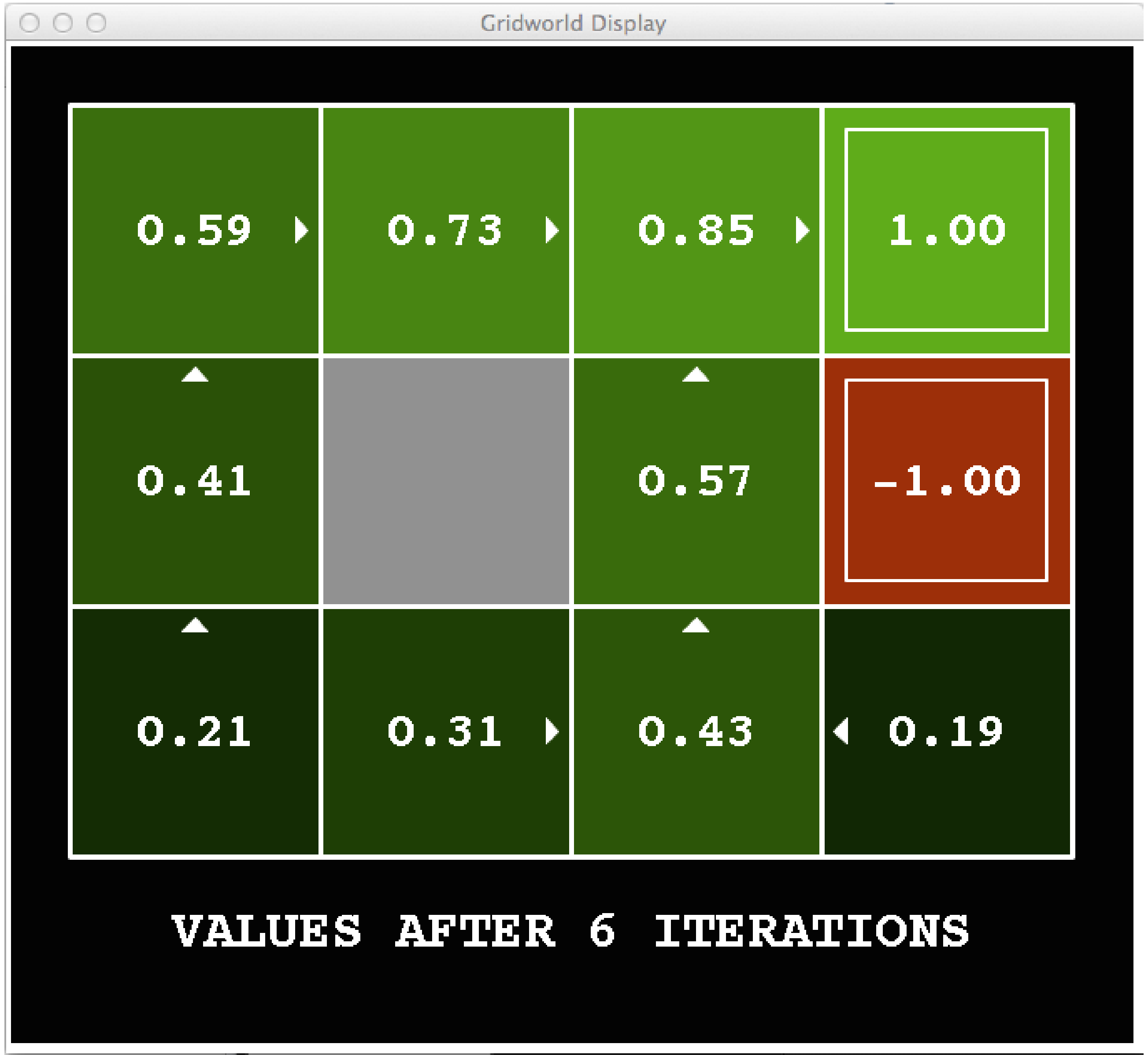
Noise = 0.2
Discount = 0.9
Living reward = 0

k=5



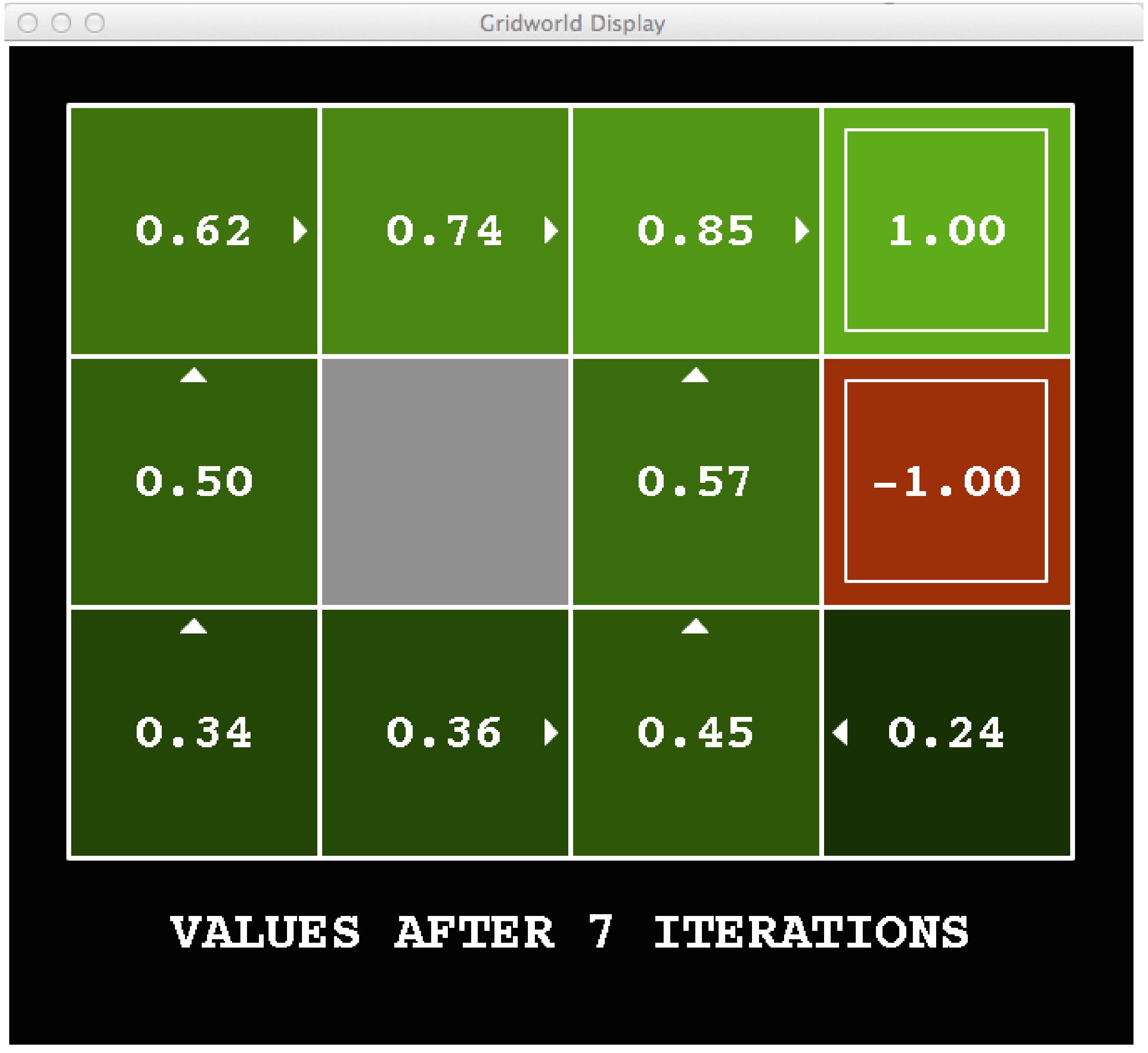
Noise = 0.2
Discount = 0.9
Living reward = 0

k=6



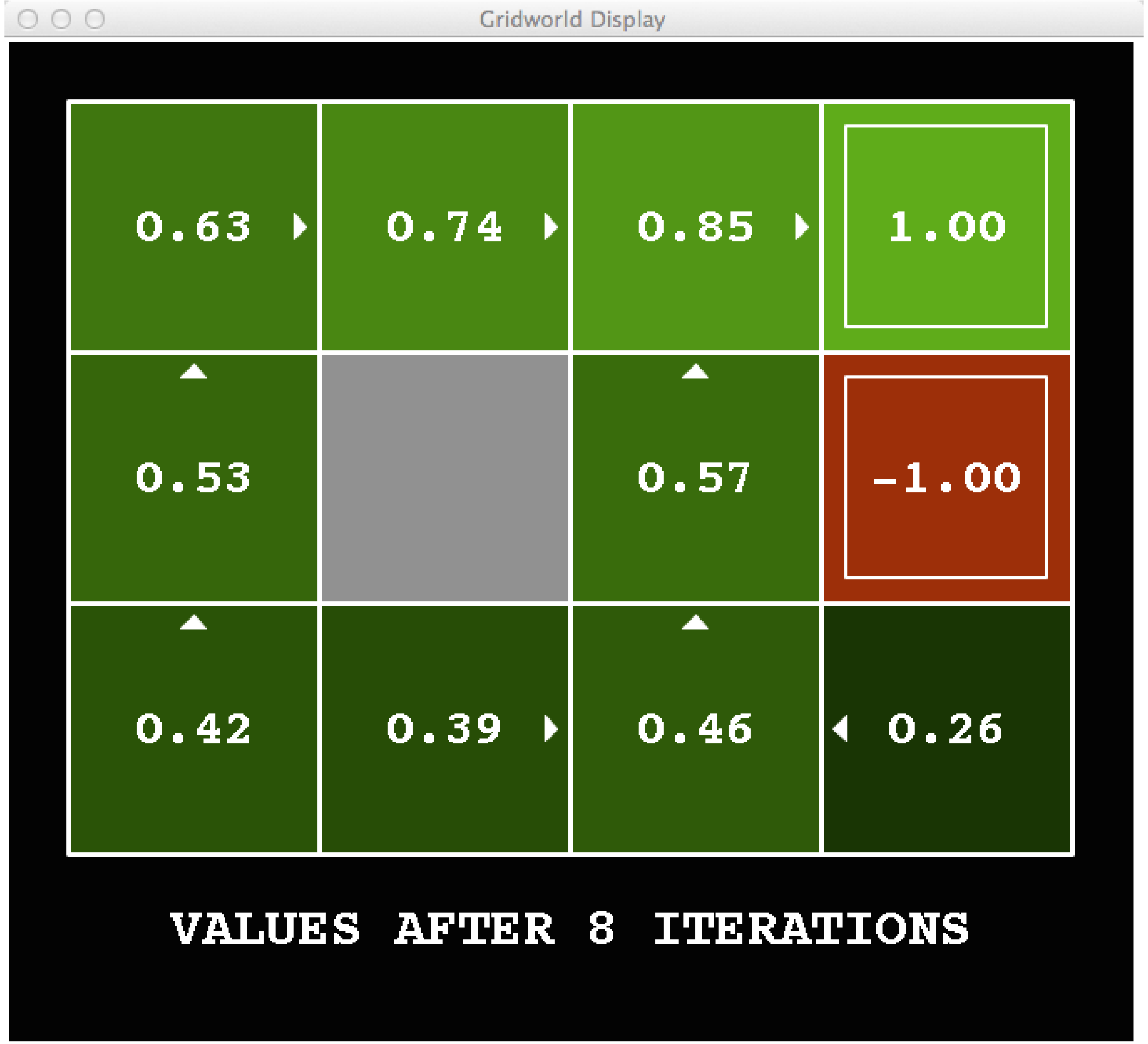
Noise = 0.2
Discount = 0.9
Living reward = 0

k=7



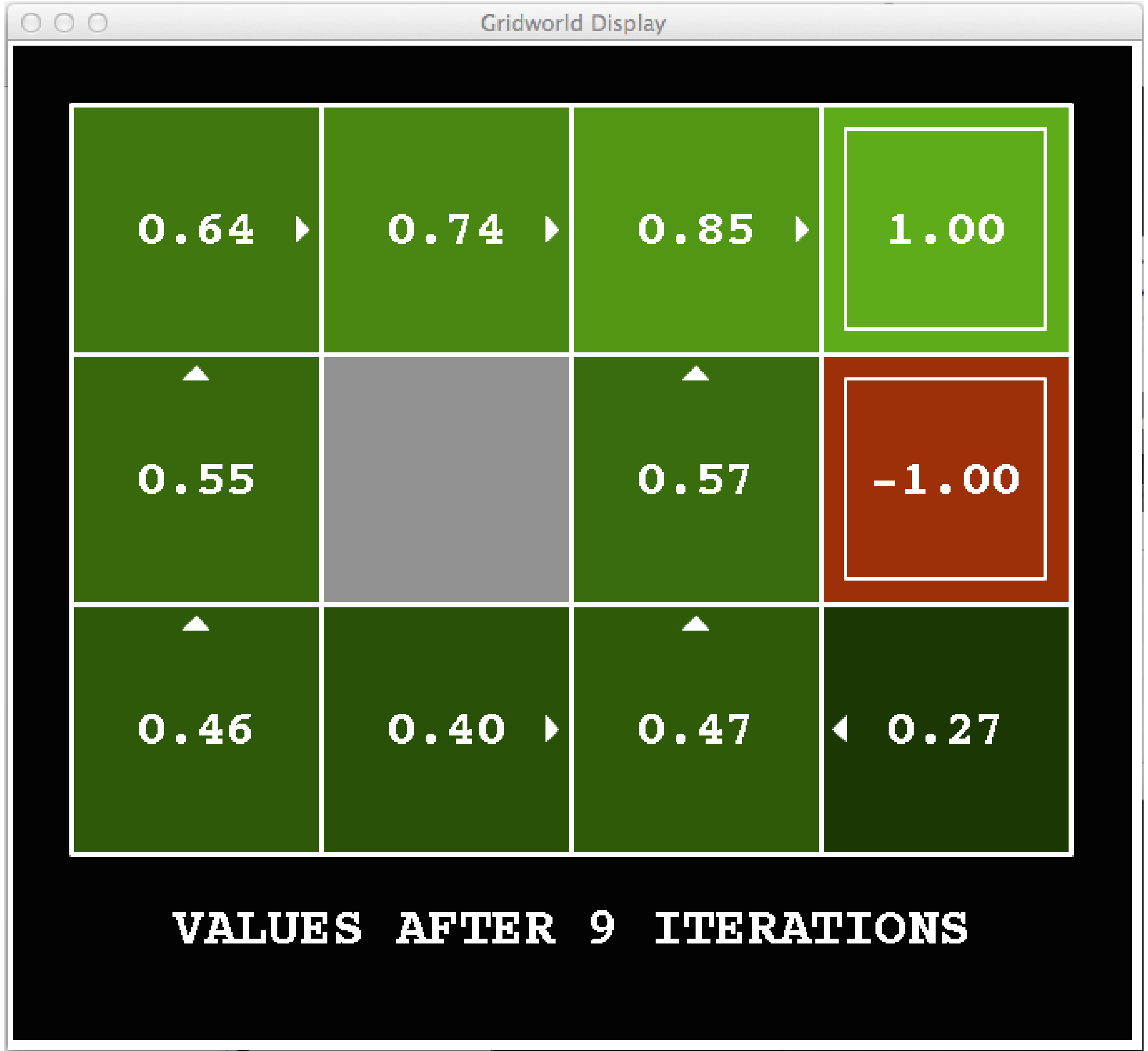
Noise = 0.2
Discount = 0.9
Living reward = 0

k=8



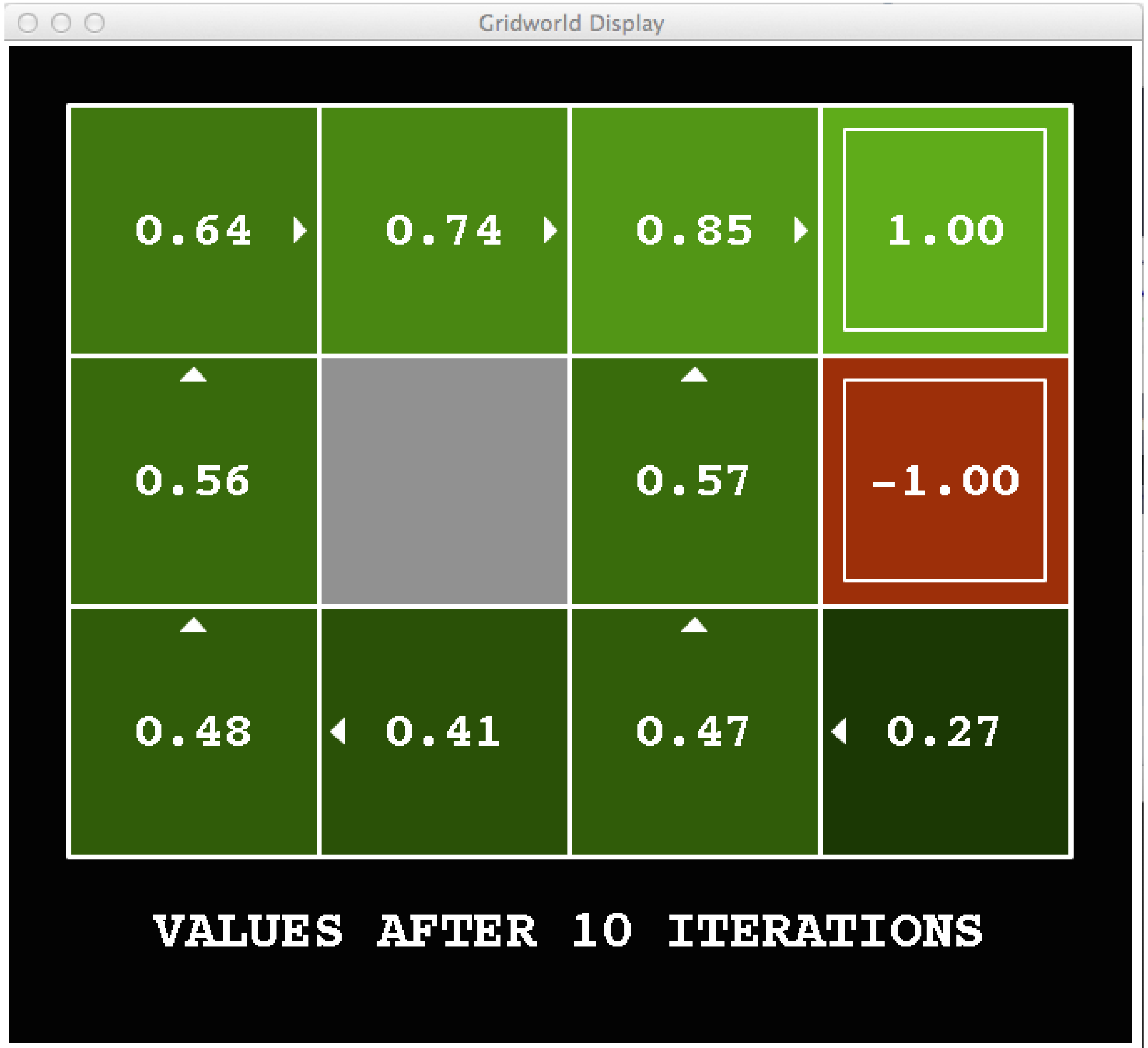
Noise = 0.2
Discount = 0.9
Living reward = 0

k=9



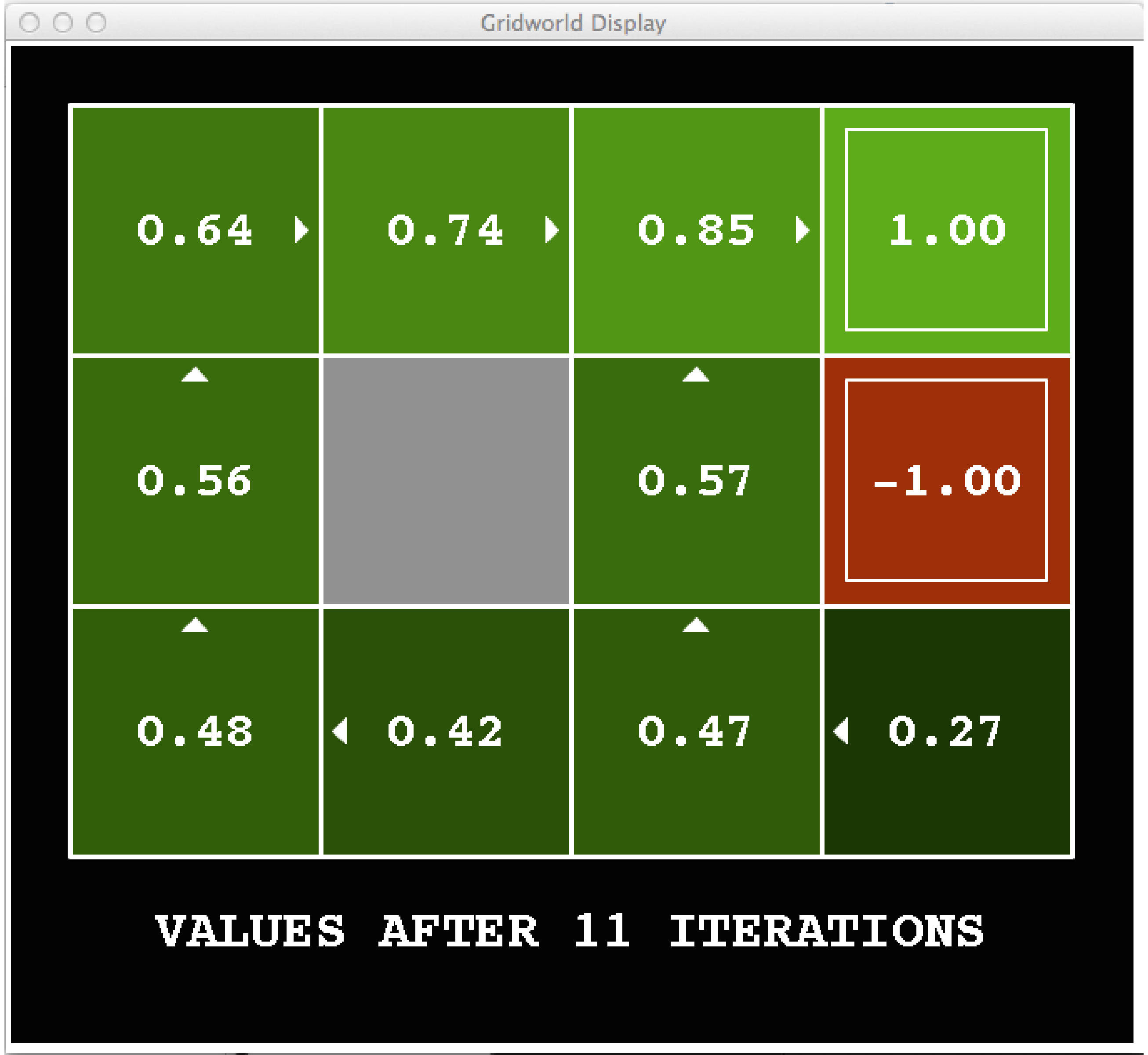
Noise = 0.2
Discount = 0.9
Living reward = 0

k=10



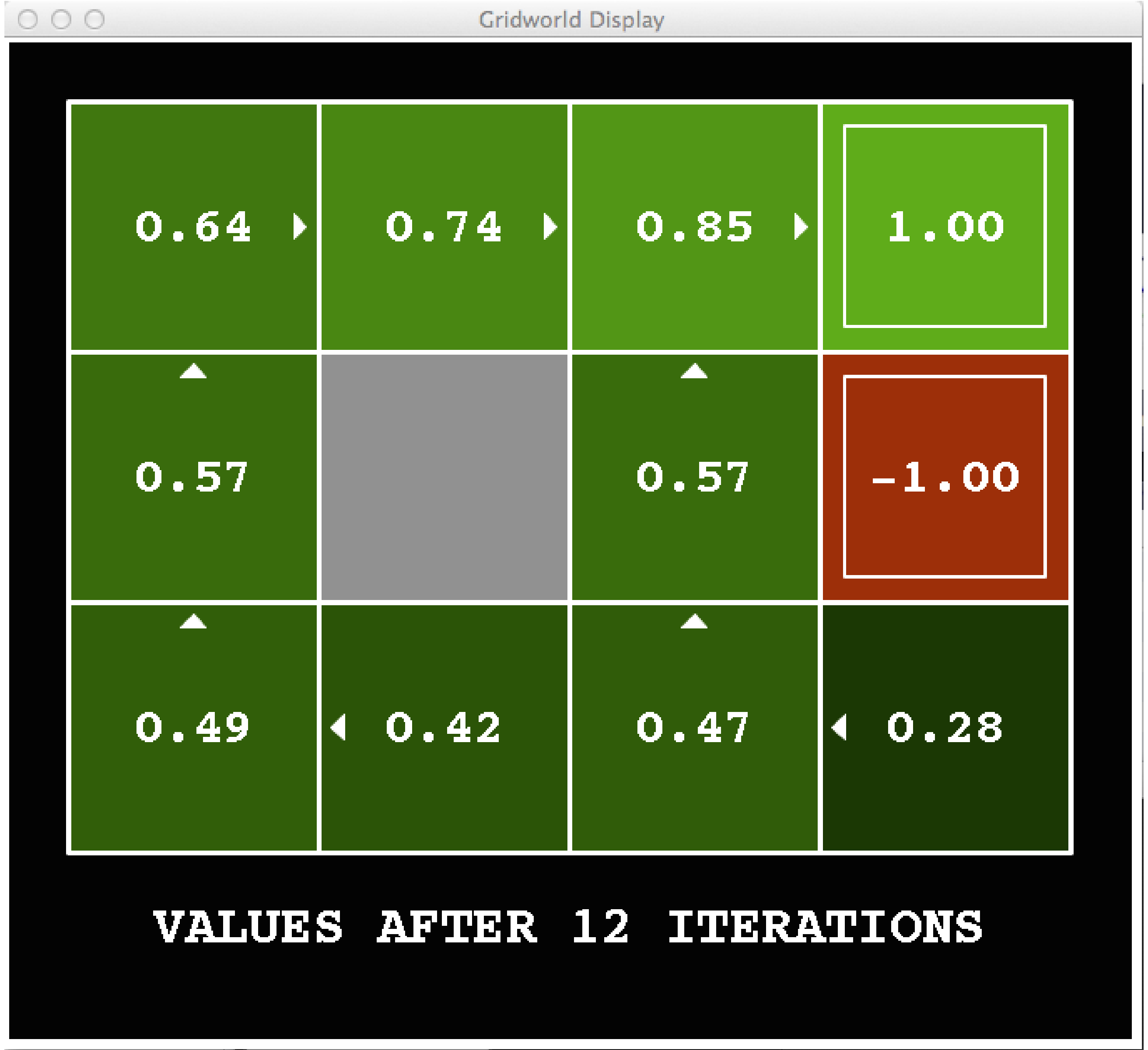
Noise = 0.2
Discount = 0.9
Living reward = 0

k=11



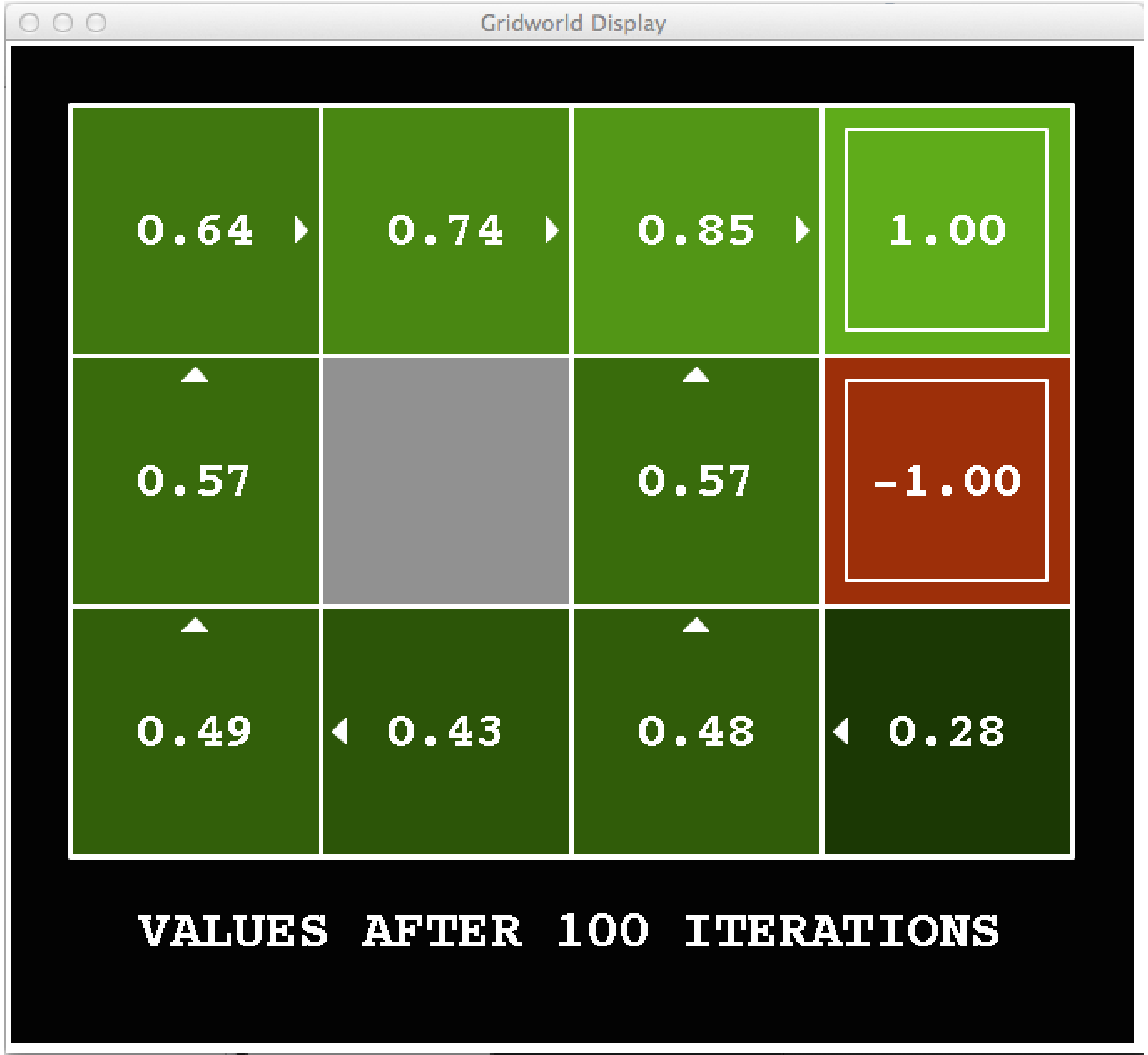
Noise = 0.2
Discount = 0.9
Living reward = 0

k=12



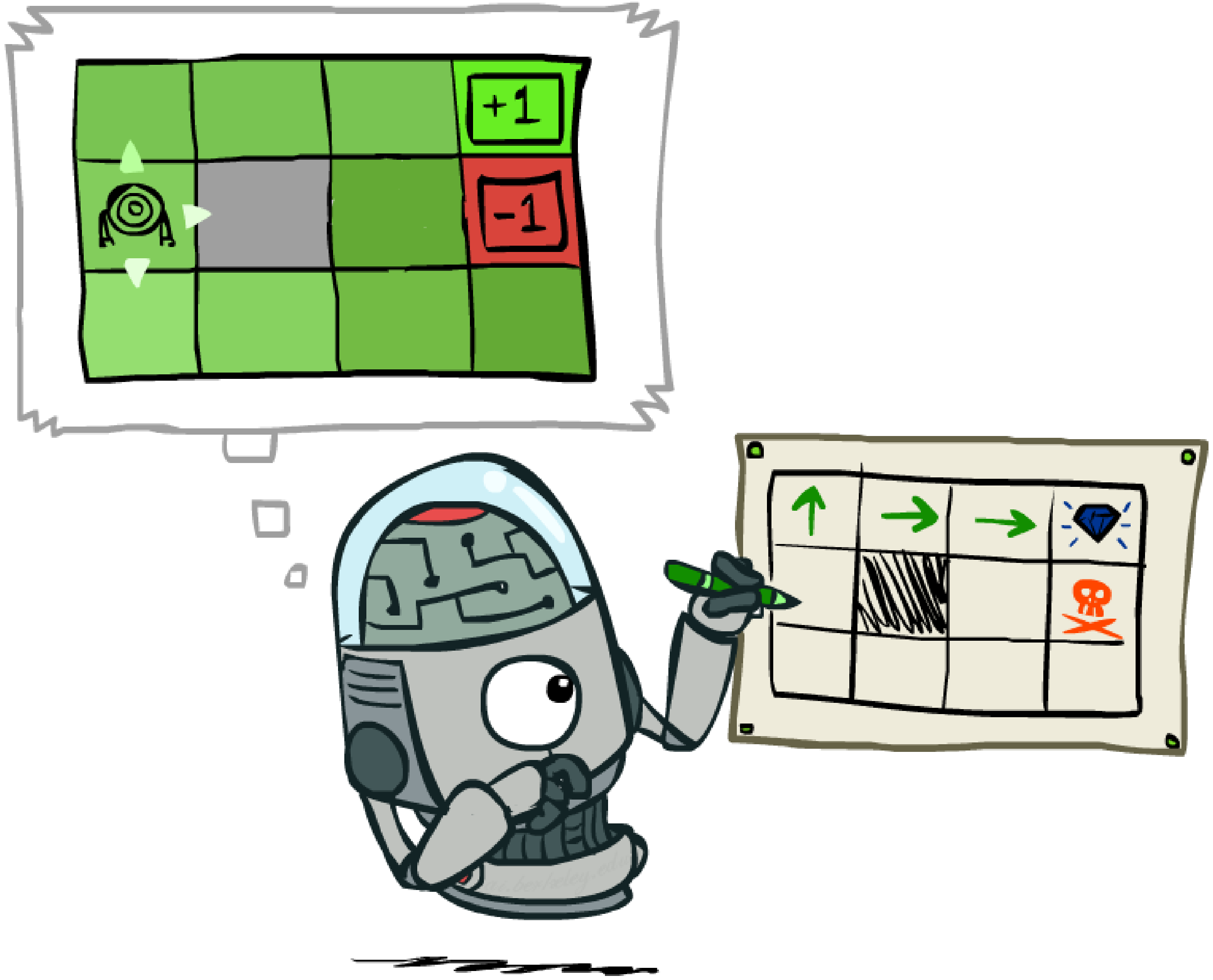
Noise = 0.2
Discount = 0.9
Living reward = 0

k=100

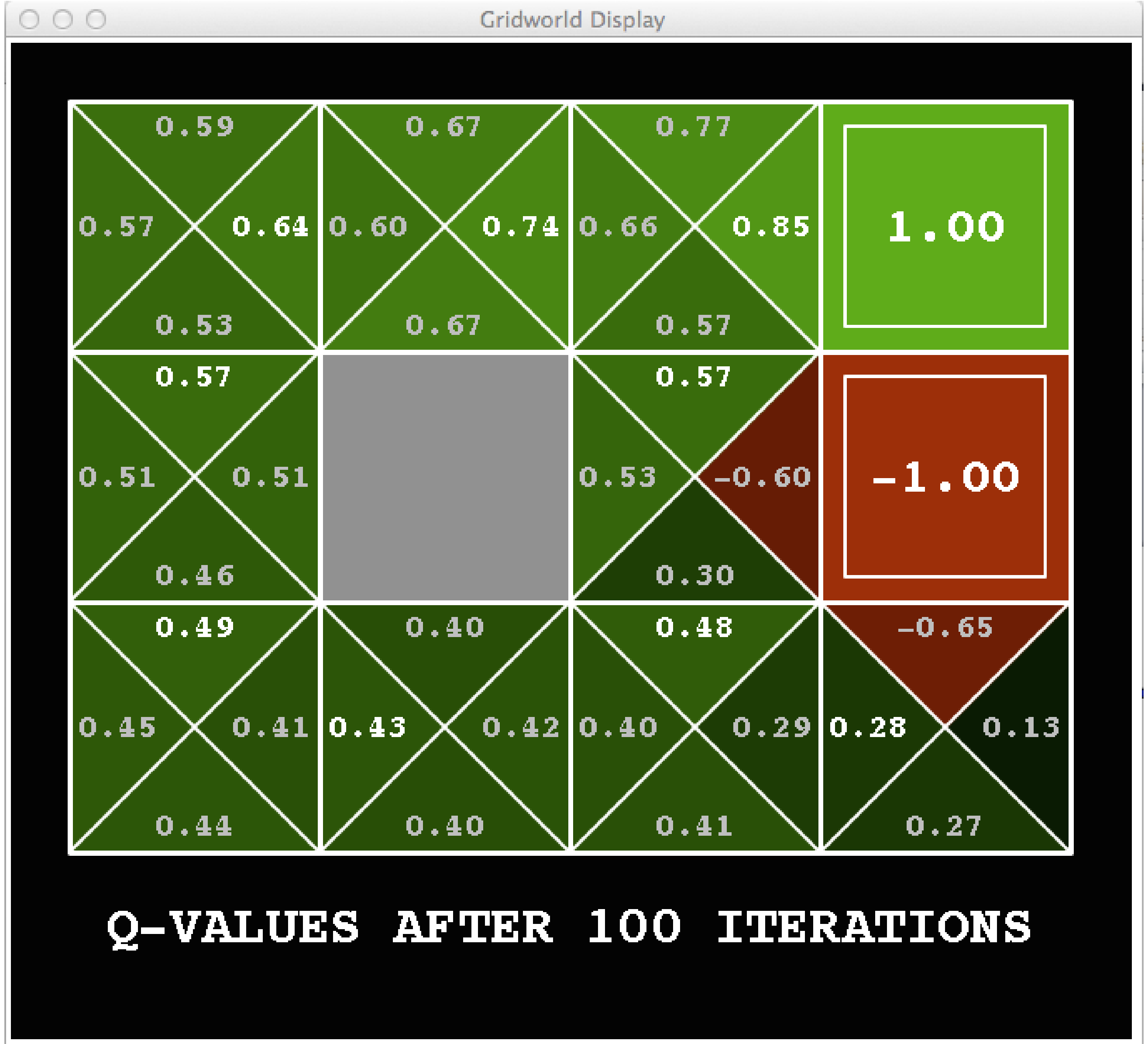


Noise = 0.2
Discount = 0.9
Living reward = 0

Policy Extraction

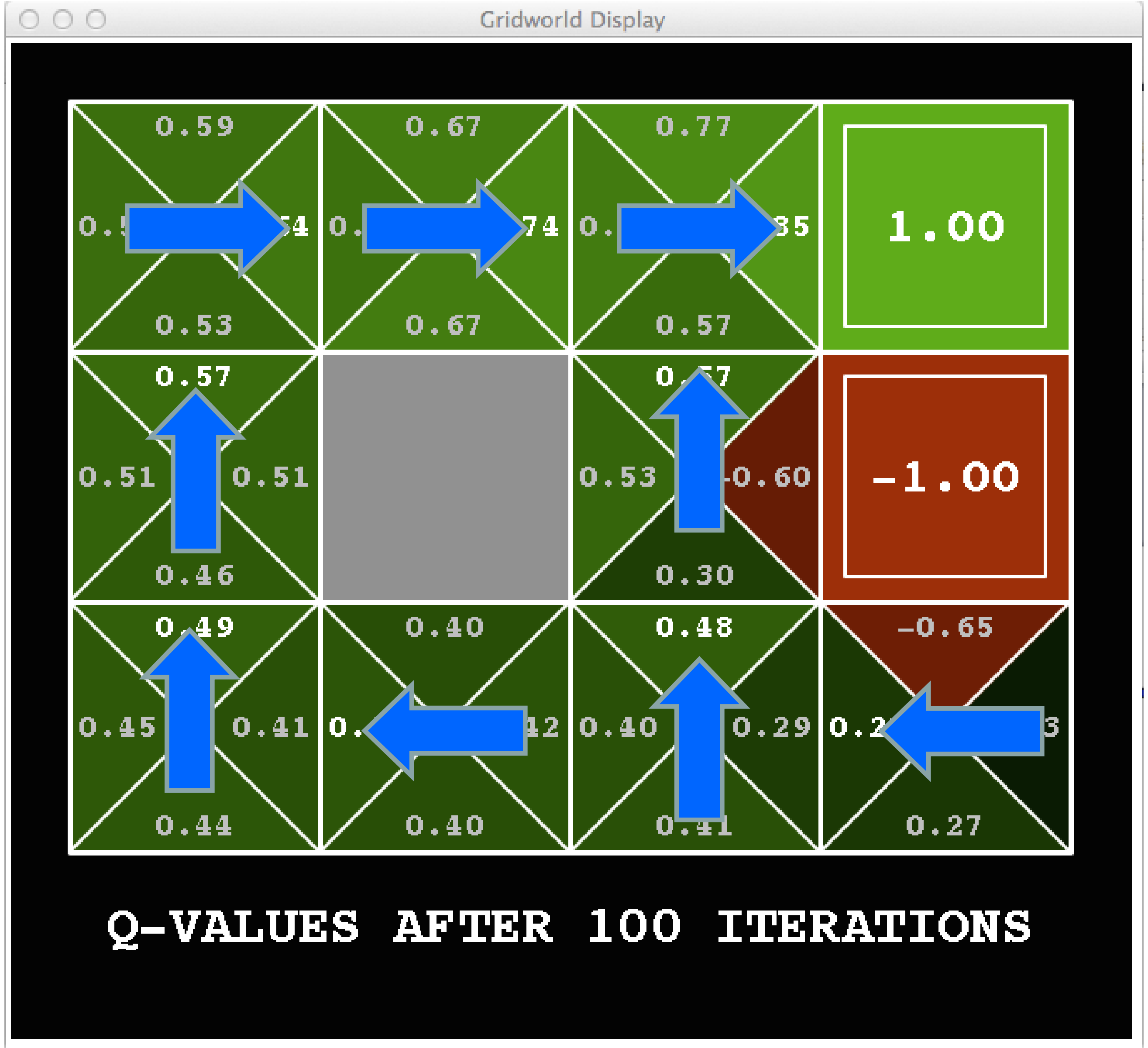


Q Values



Noise = 0.2
Discount = 0.9
Living reward = 0

Optimal Policy



Noise = 0.2
Discount = 0.9
Living reward = 0

Example IV: Jack's Car Rental Problem

- States: Two locations, maximum of 20 cars at each
- Actions: Move up to 5 cars between locations overnight
- Reward: \$10 for each car rented (must be available)
- Transitions: Cars returned and requested randomly
  - Poisson distribution, n returns/requests with prob $\frac{\lambda^n}{n!}e^{-\lambda}$
  - 1st location: average requests $= 3$, average returns $= 3$
  - 2nd location: average requests $= 4$, average returns $= 2$

Shared-bike Relocation Problem

Policy Iteration