SINCE 1947
SEO KYEONG
UNIVERSITY

[지능 시스템 4차 과제]

과목명	지능 시스템
담당 교수님	조영완 교수님

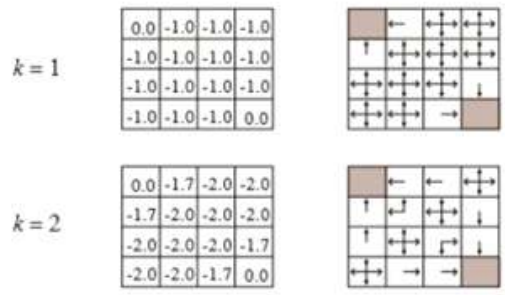
전공	컴퓨터공학과
학번	2018305068
이름	정동기

제출일	2022.5.24

숙제 4 (Dynamic Programming)

- 다음의 7 × 7 Grid map에서 최종 도착 목표 지점이 (3,4)이고 ▲로 표시된 위치에 장애물이 있을 때, 출발점 ■에서 목표지점을 찾아 가는 행동을 Dynamic Programming의 Policy Iteration과 Value Iteration을 이용해서 구하시오. Iteration 회수(예를 들어, $k = 0, 5, 10, 20, 50$ 등)에 따른 각 상태에서의 value table 값과 action list를 결과로 제시하시오. (제출: 5월 27일 6시까지, Google Classroom)

과제를 하기에 앞서 Policy Iteration과 Value Iteration에 대해 설명하도록 하겠다. Policy Iteration은 정책을 평가하고 정책을 발전시키면서 최적의 정책을 구하는 것이고, Value Iteration은 최적 정책에 대한 가치함수를 구하는 것이다. Policy Iteration을 먼저 설명하자면, 정책 평가와 정책 발전을 번갈아 수행하며 최적 정책을 구하는 방법이다. 정책 평가는 벨만 기대방정식으로 주어진 정책에 대한 가치함수를 구하고 구한 가치함수로부터 새로운 가치함수를 업데이트 하는 방법이다. 예전에 배운 다이나믹 프로그래밍 파트에 예제가 나와있다.



상태 변환 확률이 1, 가감율이 1일 때 상황이다. 상하좌우로 갈 수 있을 때 얻을 수 있는 가치함수의 합이다. (1,0)칸을 예를 들어 설명하자면 (-1/4)+(-1/4)+(-1/4)+0 이기 때문에 -1.7이 된다.(소숫점 1의 자리에서 반올림)

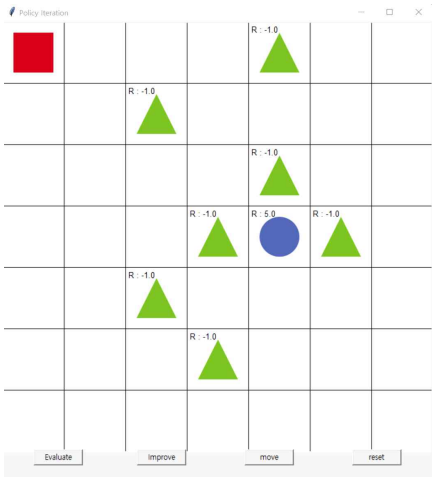
$$v_{k+1}(s) = \sum_{a \in A} \pi(a \mid s)(R_s^a + \gamma v_k(s'))$$

정책 발전은 탐욕 정책을 이용하여 큐 함수가 가장 큰 방향의 정책을 1/n으로 바꾸어 주는 것이다. n은 최댓값이 겹칠 때의 개수를 나타낸다. 이 과정을 반복해서 최적의 방법을 구현해 내는 것이 Policy Iteration이다.
Value Iteration은 한 번의 정책 평가 후 정책이 바로 개선되는 방법이다. 그리디한 방법을 내포하고 있다. 쉽게 말해 보상+감가율*가치함수의 최댓값으로 업데이트해주는 것이다.

$$max_a \sum_{s'} P^a_{ss'} [R_{t+1} + \gamma v_k(s')]$$

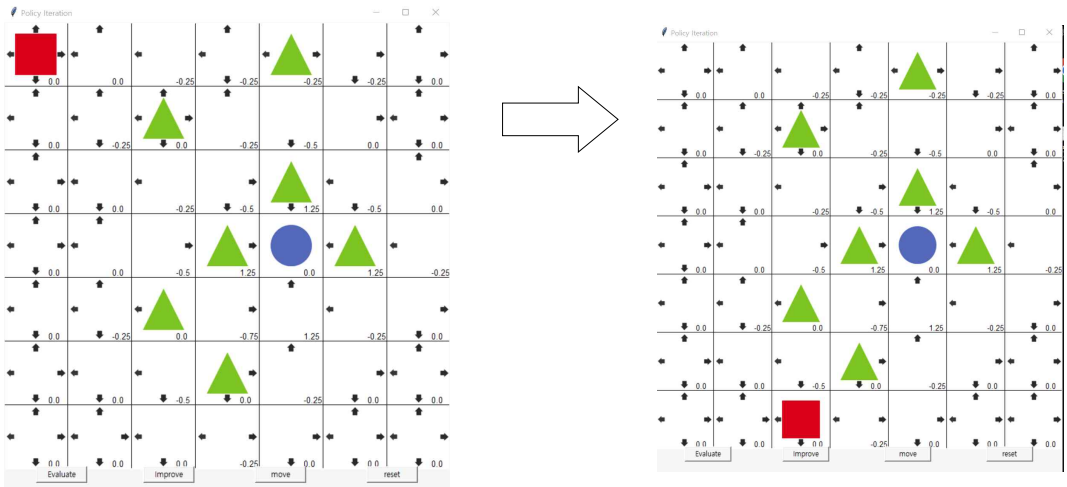
이제 바뀐 코드의 결과를 시연하겠다.
Policy Iteration

0) 기본적인 형태



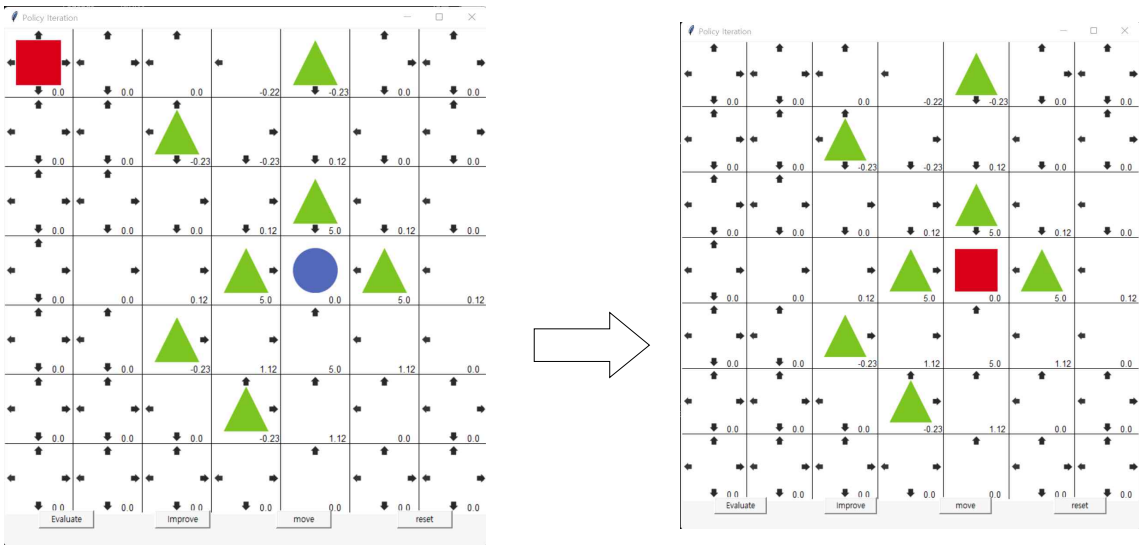
문제를 보여주는 기본적인 형태이다.

1) 첫 번째 업데이트



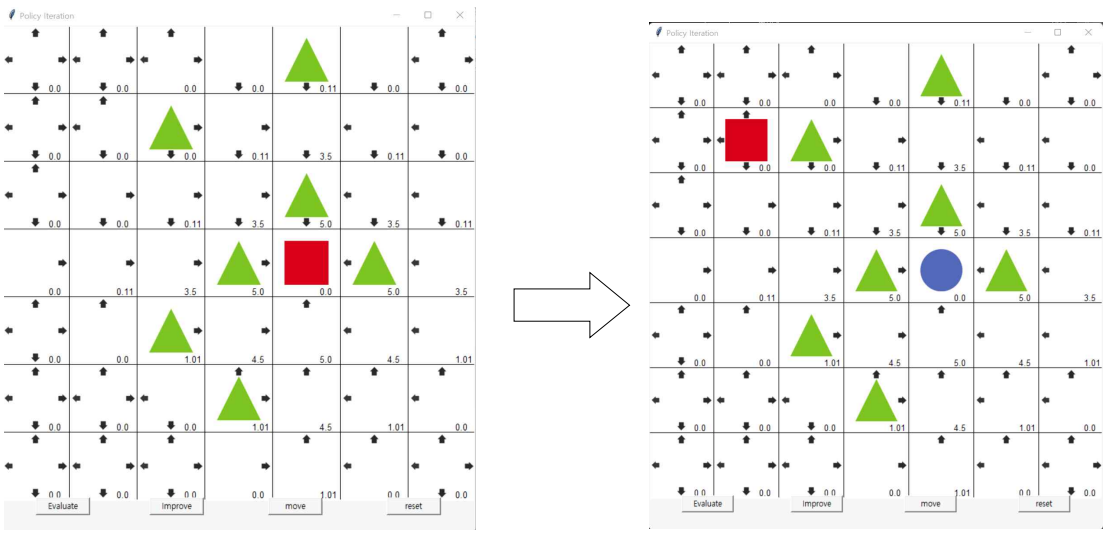
상하좌우 가치함수의 합으로 업데이트 되었다. 도착점인 파란 원으로 도착하기에는 업데이트가 부족하다.

2) 두 번째 업데이트



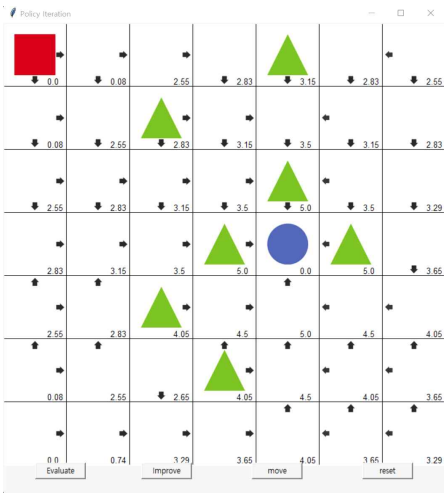
도착지점의 보상이 5 였어서 그런지 초록 세모를 지나가는 모습을 보여준다. 하나의 길로만 가지 않는 모습을 보여주고 있으며, 왔던 길을 돌아가는 모습을 보여준다. x, y 값이 (3, 4)를 넘어가는 모습을 보여준다. 예를 들어 (0, 3)에서 (0, 4)로 내려갈 수 있는 가능성이 있다.

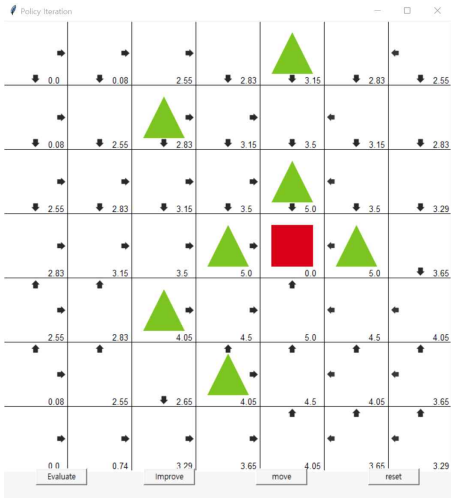
3) 세 번째 업데이트



두 번째 업데이트와 비슷하지만, x, y 값이 (3, 4)를 넘어가지 않는다는 점이 다른 점이다.

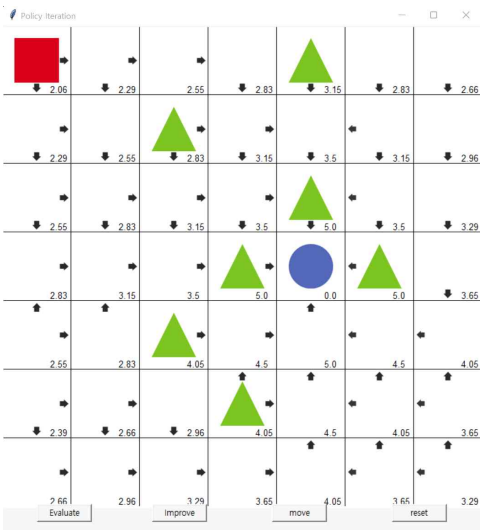
4) 여섯 번째 업데이트

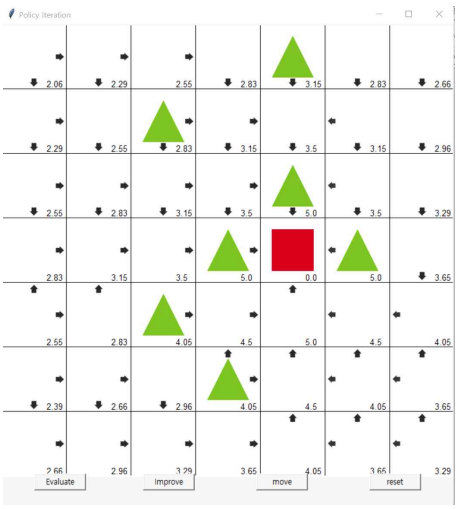




세 번째 업데이트와 거의 동일하지만 다른 점으로는 왔던 길을 돌아가지 않고 한 번에 길을 찾아간 다는 점을 들을 수 있다.

5) 아홉 번째 업데이트

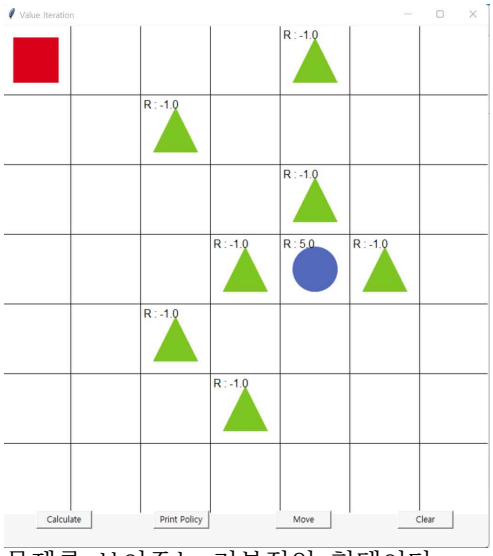




아홉 번째 이후로는 업데이트를 하여도 값이 똑같이 나온다. 여섯 번째와 마찬가지로 다양한 길로 가지만 다시 뒤로 가거나 하지는 않는다.

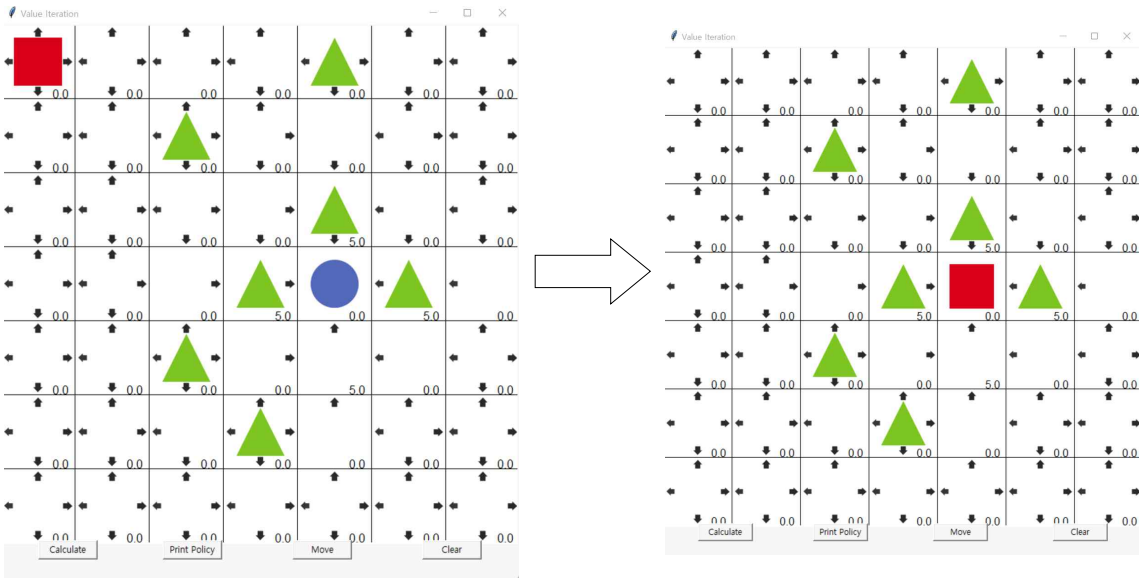
Value Iteration

0) 기본적인 형태



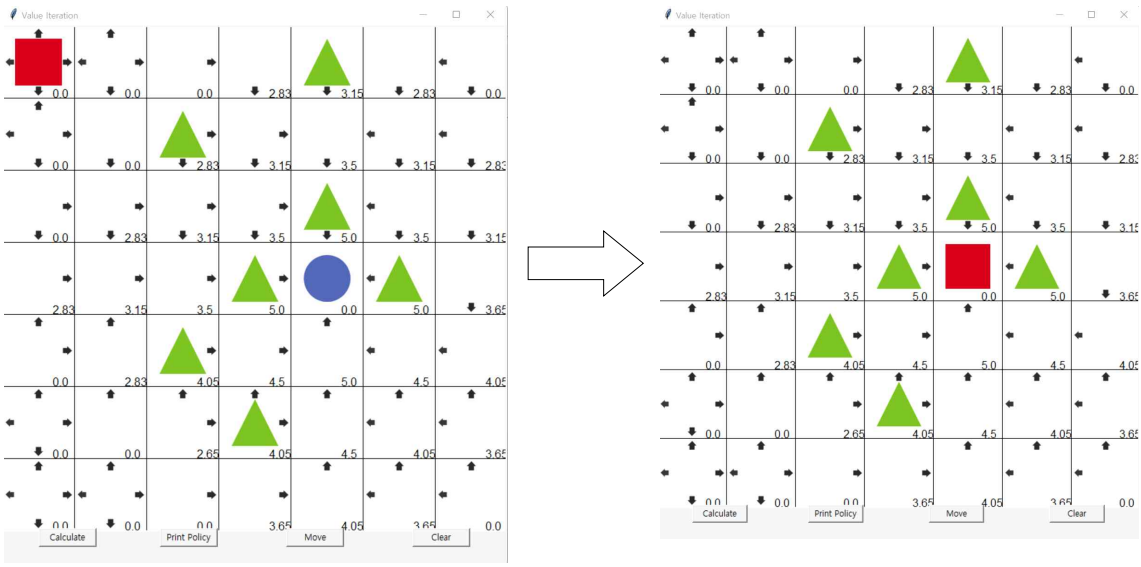
문제를 보여주는 기본적인 형태이다.

1) 첫 번째 업데이트



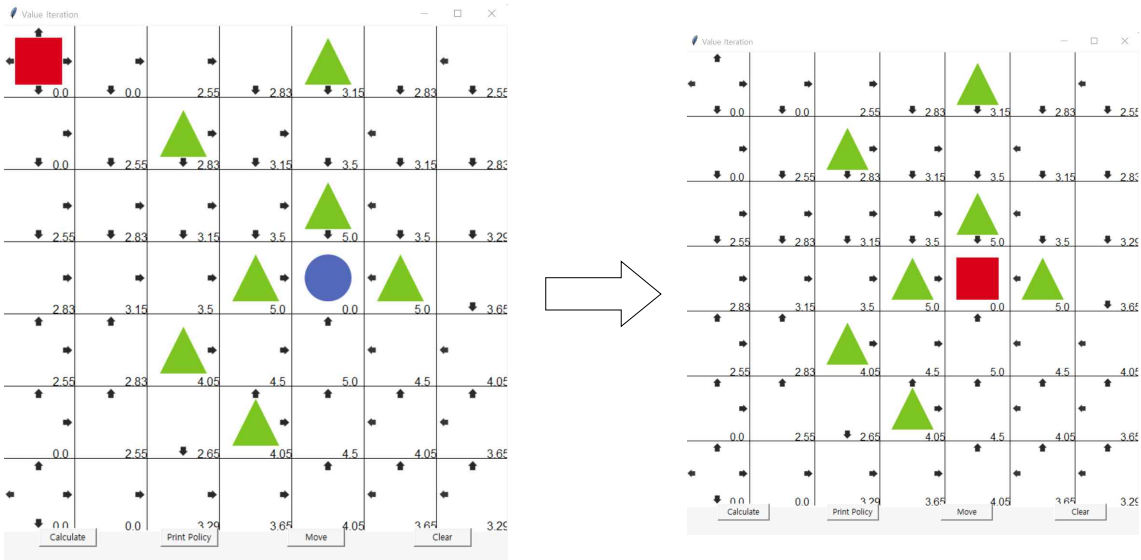
위치에서 상하좌우에서 앞서 말한 식의 값 중 가장 큰 값으로 업데이트 해주는 모습을 볼 수 있다. 도착지점의 보상이 5 였어서 그런지 초록 세모를 지나가는 모습을 보여준다. 하나의 길로만 가지 않는 모습을 보여주고 있으며, 왔던 길을 돌아가는 모습을 보여준다. x, y 값이 (3, 4)를 넘어가는 모습을 보여준다.

2) 네 번째 업데이트



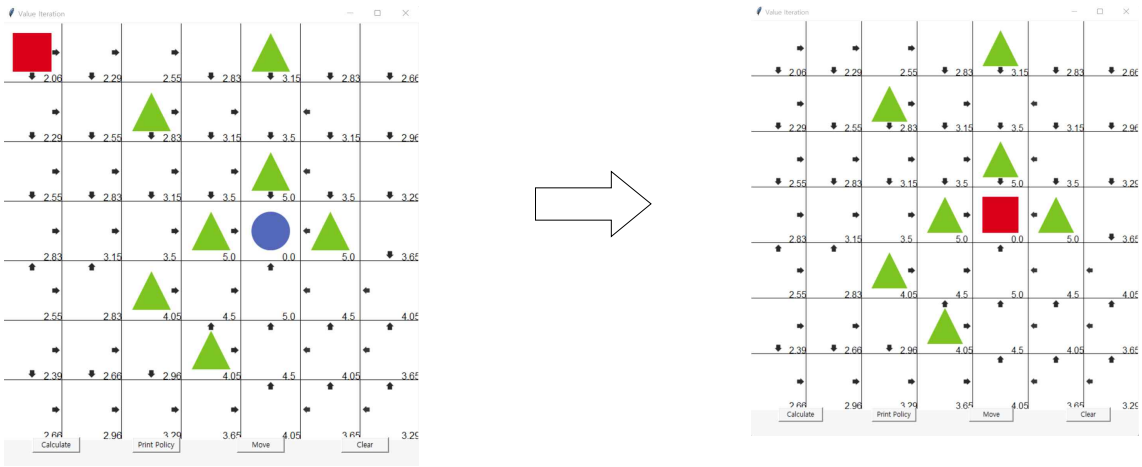
첫 번째 업데이트와 비슷하지만, x, y 값이 (3, 4)를 넘어가지 않는다는 점이 다른 점이다.

3) 다섯 번째 업데이트



네 번째 업데이트와 비슷하지만, 왔던 길을 다시 돌아가지 않는다는 점이 다르다.

4) 일곱 번째 업데이트



일곱 번째 이후로는 업데이트를 하여도 값이 똑같이 나온다. 다섯 번째와 마찬가지로 다양한 길로 가지만 다시 뒤로 가거나 하지는 않는다.