**AI\_Assignment 06 Report**

**1분반 20191244 김현승**

1. **compute\_transition\_matrix**

MxNx4xMxN 크기의 numpy 배열 P를 초기화한 후 P[r, c, a, r’, c’]을 계산하기 위해 5개의 for 문에 진입한다.



현재 state가 terminal state인 경우 모든 a, r’, c’에 대해 P[r, c, a, r’, c’] = 0이다. 현재 state가 terminal state가 아닌 경우 다음과 같이 확률을 계산했다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

우선 현재 취하고자 하는 action에 따라 want, cntclk, clk 좌표를 계산했다. want는 현재 state에서 원하는 action을 취했을 때의 state이고, cntclk는 원하는 방향의 반시계 방향으로 action을 취했을 때 state, clk는 원하는 방향의 시계 방향으로 action을 취했을 때의 state이다. 위와 같이 좌표를 구한 다음 want, cntclk, clk의 좌표에 벽이 있는지 확인한다. 벽이 있다면 해당 좌표를 현재 state로 돌려 놓는다.

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

이후 want, cntclk, clk의 좌표가 (r’, c’)인 경우 각각 D[r, c, 0], D[r, c, 1], D[r, c, 2] 의 확률을 더해주면 P table의 확률 계산이 완료된다.

1. **update\_utility**

U\_next를 MxN numpy 영행렬로 초기화한다. r, c, a에 대해 반복문을 도는데, 이 때 모든 다음 state s’, 즉 (r’, c’)에 대해 P[r, c, a, r’, c’] \* U\_current[r’, c’]의 합을 구하고, 값을 저장해가며 모든 action에 대해 P[r, c, a, r’, c’] \* U\_current[r’, c’]의 합의 최댓값을 구한다.

텍스트, 스크린샷, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

최댓값을 구한 후 정해진 감마 값을 더하고, R[r, c]을 더하면 U\_next[r, c] 값을 구할 수 있다.

**실행 결과**

텍스트, 도표, 스크린샷이(가) 표시된 사진

자동 생성된 설명

1. **value\_iteration**

value\_iteration() 함수에서는 update\_utility를 반복하며 모든 state에 대해 현재 효용과 다음 효용의 차이가 충분히 작아질 때까지 반복한다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

2번에서 구현한 update\_utility() 함수를 이용해 다음 효용을 구하고 calculate\_delta 함수로 현재 효용과 다음 효용의 차이를 확인한다. 만약 모든 state에 대해 두 효용의 차가 epsilon보다 작아지면 calculate-delta() 함수는 True를 반환해 위의 for문을 탈출하게 된다. 그렇지 않으면 최대 100번까지 효용 업데이트를 계속하게 된다.

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

위에서 설명했듯, calculate\_delta() 함수는 모든 state (r, c)에 대해 U\_current와 U\_next 값의 차이를 계산하고 epsilon보다 크거나 같은 값을 갖는 state가 있으면 False, 모두 작으면 True를 리턴하는 함수이다. 차이가 epsilon보다 큰 state를 찾은 경우 바로 반복문을 멈추고 return해 조금 더 빠른 실행 시간을 가질 수 있다.

**실행 결과**

텍스트, 스크린샷, 도표, 라인이(가) 표시된 사진

자동 생성된 설명