RL 스터디

6 회차

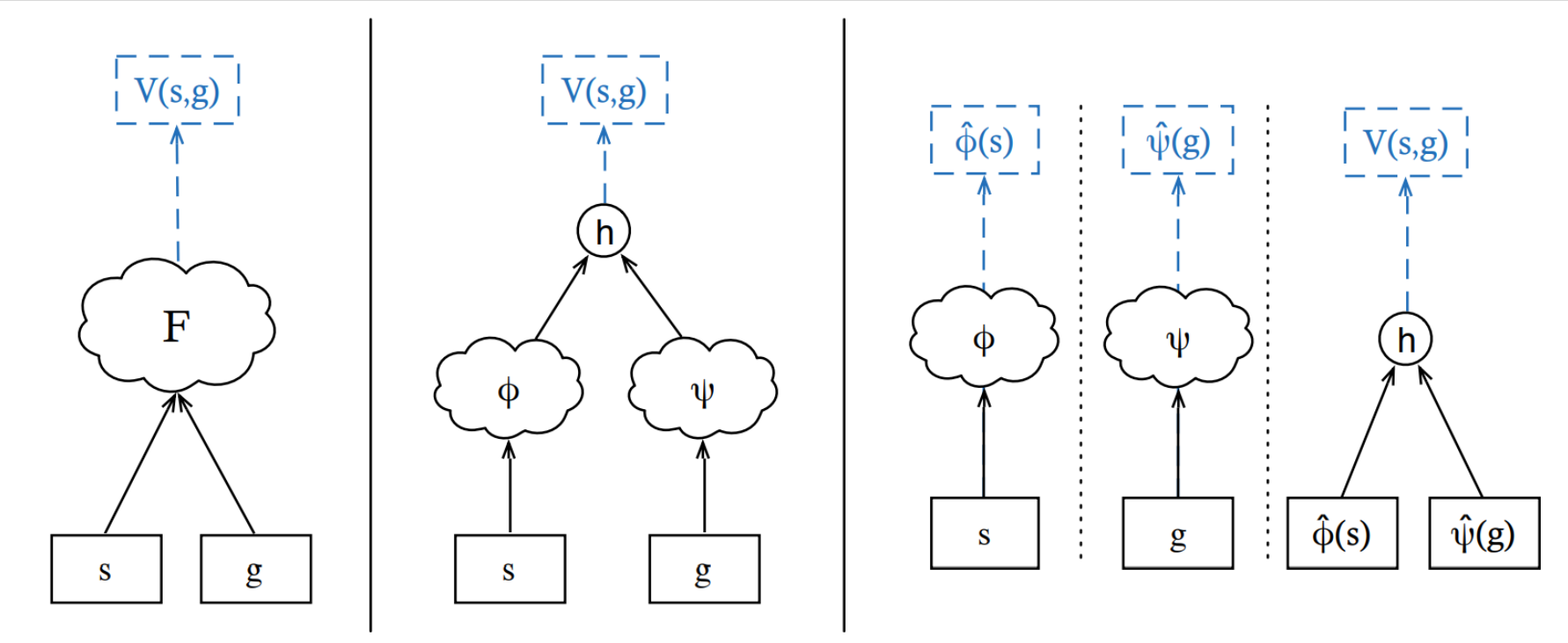
goal ? 이란

V vs G

- State => Value 근사가 이루어지는데 벨만 방정식으로 인해 Value 공간은 실제 Goal 공간과 유사해짐
- Goal 말 그대로 가고자 하는 목표
- 그렇다면 State 와 Goal을 동시에 주어 준다면 Value 근사가 더 잘 이루어지지 않을ㄲ

UVFA – Universal Value Function Approximators

- 단순한 두 입력의 합으로 쓰는게 아니라 비선형 함수를 거친 후 내적 함수를 통해 계산된 값을 이용
- Goal 과 State 계산 첫째 layer 는 가중치를 공유하여 그 연관성을 더욱 증가 시킬 수 있다.



UVFA

- 첫번째 학습 방식

- S,G를 인수로 넣고 바로 MSE 에러를 통해 학습
- 최적 정책과 어떤 temperature로 간 policy에 대해서 goal의 평균 기대 할인 보상을 타겟으로 사용

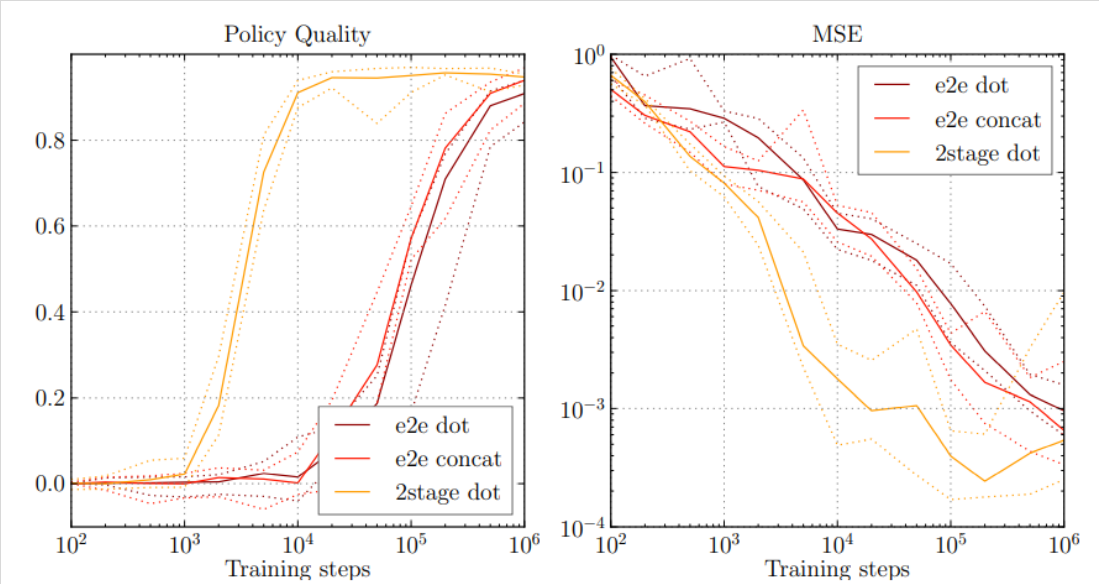
$$\text{(MSE)}\ \mathbb{E}\left[\left(V_g^*(s) - V(s, g; \theta)\right)^2\right]$$

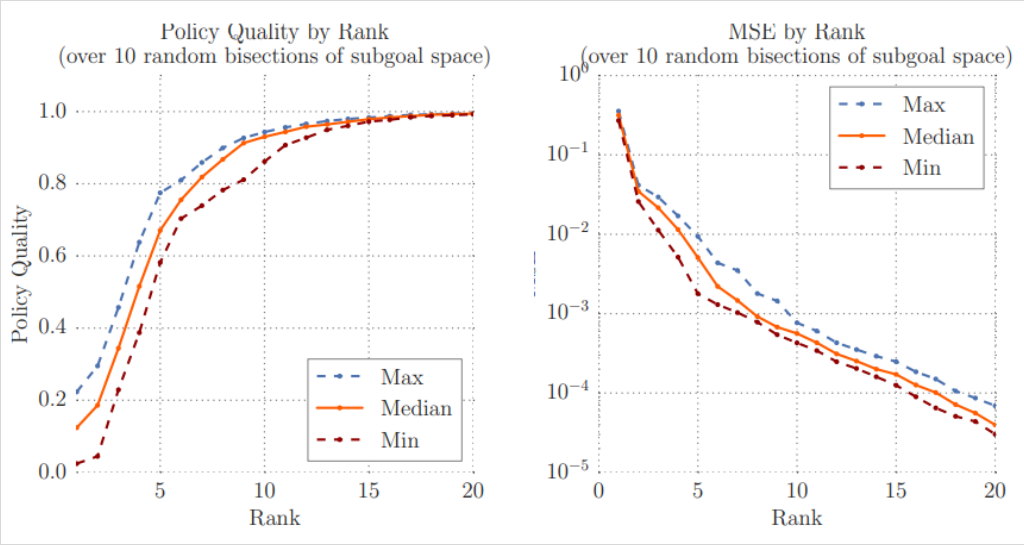
UVFA

- 두번째

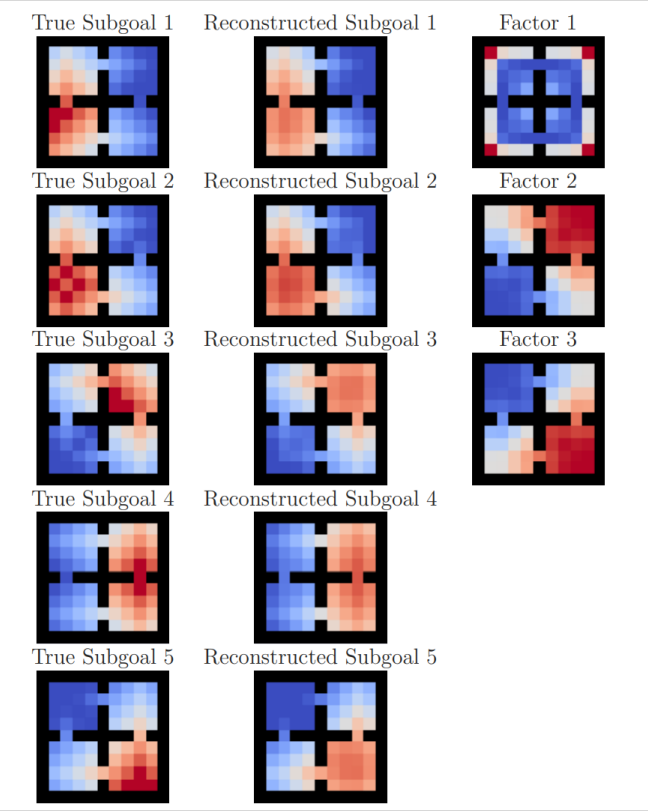
- Vg* 함수로 얻는 데이터 매트릭스를 행렬 분해를 통한 low rank approximation을 거친 값에 대해서 각각
- S 함수와 G 함수를 다변량 회귀를 진행

UVFA


Policy Quality
MSE
e2e dot
e2e concat
2stage dot
Training steps
Training steps


Policy Quality by Rank
(over 10 random bisections of subgoal space)
MSE by Rank
(over 10 random bisections of subgoal space)
Max
Median
Min
Policy Quality
Rank
Rank

UVFA


True Subgoal 1
Reconstructed Subgoal 1
Factor 1
True Subgoal 2
Reconstructed Subgoal 2
Factor 2
True Subgoal 3
Reconstructed Subgoal 3
Factor 3
True Subgoal 4
Reconstructed Subgoal 4
True Subgoal 5
Reconstructed Subgoal 5

UVFA 알고리즘

- Trasition 추출을 통한 Qg 함수 학습
- 이를 바탕으로 M 제작
- 이를 s, g 임베딩 네트워크를 학습

Algorithm 1 UVFA learning from Horde targets

Input: rank n, training goals $\mathcal{G}_T$, budgets b_1, b_2, b_3
Initialise transition history $\mathcal{H}$
for $t = 1$ **to** b_1 **do**
  $\mathcal{H} \leftarrow \mathcal{H} \cup (s_t, a_t, \gamma_{ext}, s_{t+1})$
end for
for $i = 1$ **to** b_2 **do**
  Pick a random transition t from $\mathcal{H}$
  Pick a random goal g from $\mathcal{G}_T$
  Update Q_g given a transition t
end for
Initialise data matrix $\mathbf{M}$
for $(s_t, a_t, \gamma_{ext}, s_{t+1})$ **in** $\mathcal{H}$ **do**
  for g **in** $\mathcal{G}_T$ **do**
    $\mathbf{M}_{t,g} \leftarrow Q_g(s_t, a_t)$
  end for
end for
Compute rank-n factorisation $\mathbf{M} \approx \hat{\boldsymbol{\phi}}^\top \hat{\boldsymbol{\psi}}$
Initialise embedding networks ϕ and ψ
for $i = 1$ **to** b_3 **do**
  Pick a random transition t from $\mathcal{H}$
  Do regression update of $\phi(s_t, a_t)$ toward $\hat{\boldsymbol{\phi}}_t$
  Pick a random goal g from $\mathcal{G}_T$
  Do regression update of $\psi(g)$ toward $\hat{\boldsymbol{\psi}}_g$
end for
return $Q(s, a, g) := h(\phi(s, a), \psi(g))$

HER

- 희소 보상 환경에서 에이전트는 무엇을 보고 따라가야 하는가?

- 보상에 도달하지 못하면 보상근처에 얼마나 가더라도 다 소용없는 일이 된다 + 그쪽으로 안 가려고 한다.
- 해결 방식 =>
- 보상을 새롭게 설정 reward shap
- 이미테이션 학습
- 호기심 기반 학습

- 하지만 인간은 다르다. 농구 골대에 공을 넣는다 하면 골대 근처에서 실패하면 그 근처로 던지려고 시도한다.
- => 목표 기반 학습

HER 알고리즘

- State, goal 을 인풋
- 학습
- Transition 저장
- 다시 transition에 대해 goal 다시 계산
- 반복

Algorithm 1 Hindsight Experience Replay (HER)

Given:

- an off-policy RL algorithm $\mathbb{A}$, ▷ e.g. DQN, DDPG, NAF, SDQN
- a strategy $\mathbb{S}$ for sampling goals for replay, ▷ e.g. $\mathbb{S}(s_0, \ldots, s_T) = m(s_T)$
- a reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \to \mathbb{R}$. ▷ e.g. $r(s, a, g) = -[f_g(s) = 0]$

Initialize $\mathbb{A}$ ▷ e.g. initialize neural networks
Initialize replay buffer R
for episode = 1, M **do**
  Sample a goal g and an initial state s_0.
  for $t = 0, T-1$ **do**
    Sample an action a_t using the behavioral policy from $\mathbb{A}$:
      $a_t \leftarrow \pi_b(s_t||g)$ ▷ $||$ denotes concatenation
    Execute the action a_t and observe a new state s_{t+1}
  end for
  for $t = 0, T-1$ **do**
    $r_t := r(s_t, a_t, g)$
    Store the transition $(s_t||g,\ a_t,\ r_t,\ s_{t+1}||g)$ in R ▷ standard experience replay
    Sample a set of additional goals for replay $G := \mathbb{S}$(**current episode**)
    for $g' \in G$ **do**
      $r' := r(s_t, a_t, g')$
      Store the transition $(s_t||g',\ a_t,\ r',\ s_{t+1}||g')$ in R ▷ HER
    end for
  end for
  for $t = 1, N$ **do**
    Sample a minibatch B from the replay buffer R
    Perform one step of optimization using $\mathbb{A}$ and minibatch B
  end for
end for

학습 환경

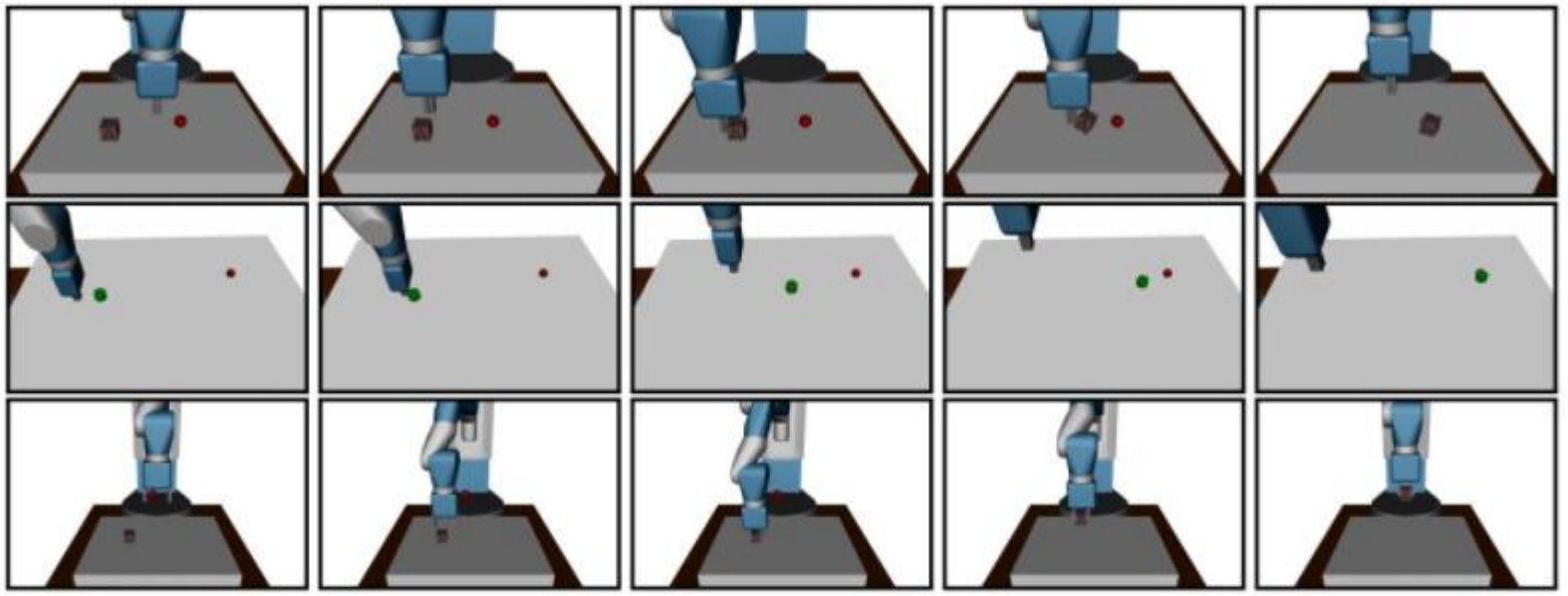


Figure 2: Different tasks: *pushing* (top row), *sliding* (middle row) and *pick-and-place* (bottom row). The red ball denotes the goal position.

골 설계

- State는 물리엔진에서 주어지는 여러가지 속도나 가속도 위치 등의 요소
- 골은 오브젝트의 위치를 나타내며 보상
- 아래는 목표를 잡고 학습할 때의 보상

$$\mathcal{G} = \mathbb{R}^3 \text{ and } f_g(s) = [|g - s_{\mathbf{object}}| \leq \epsilon]$$

Goal 재 설계

1. final

   한 episode의 마지막 state에 가지는 값을 goal로 한다. 그리고 이를 바탕으로 episode 내 k개의 랜덤 state를 샘플링

2. future

   replay를 episode 내 k개의 random state를 고르고 여기서 한 episode내 k개의 state 이후에 관측되는 값을 goal로 가진다.

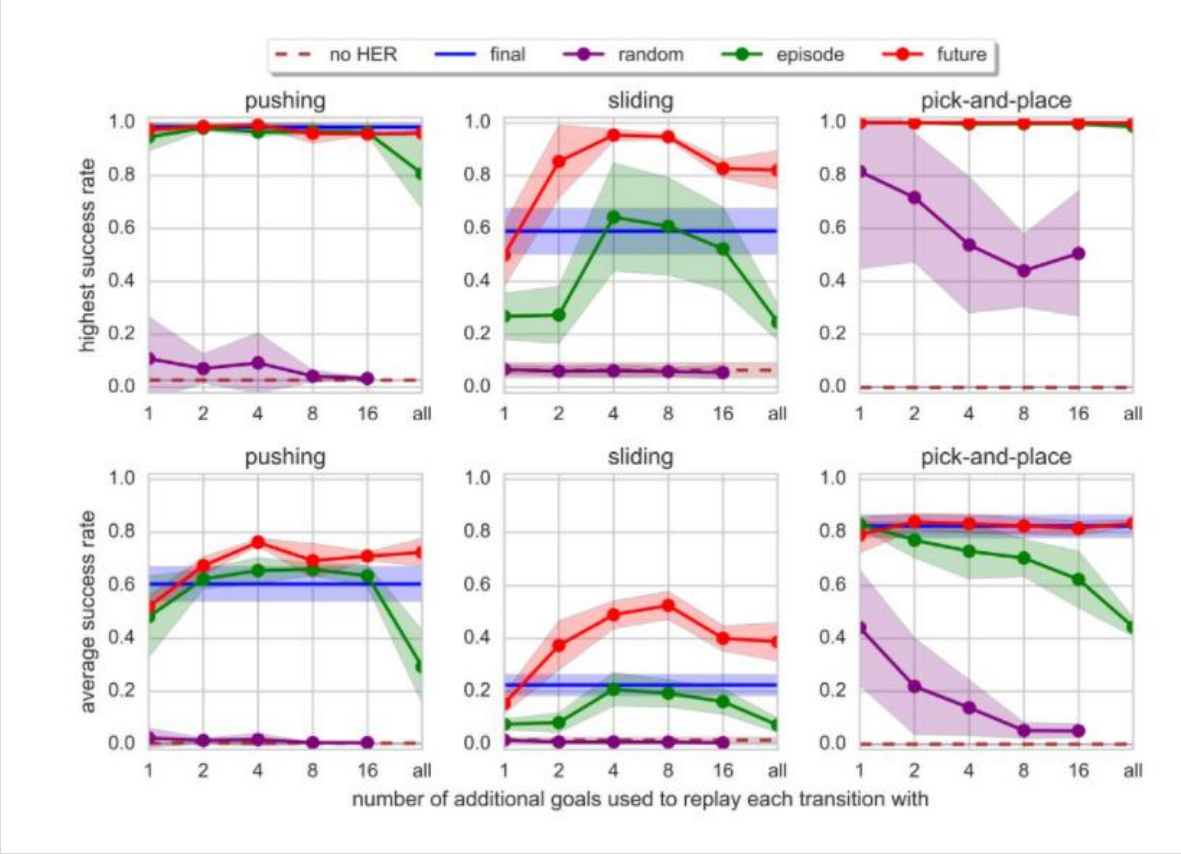
3. episode

   한 episode 내 랜덤한 state에 가지는 값을 goal로 한다. 그리고 이를 바탕으로 episode 내 k개의 랜덤 state를 샘플링

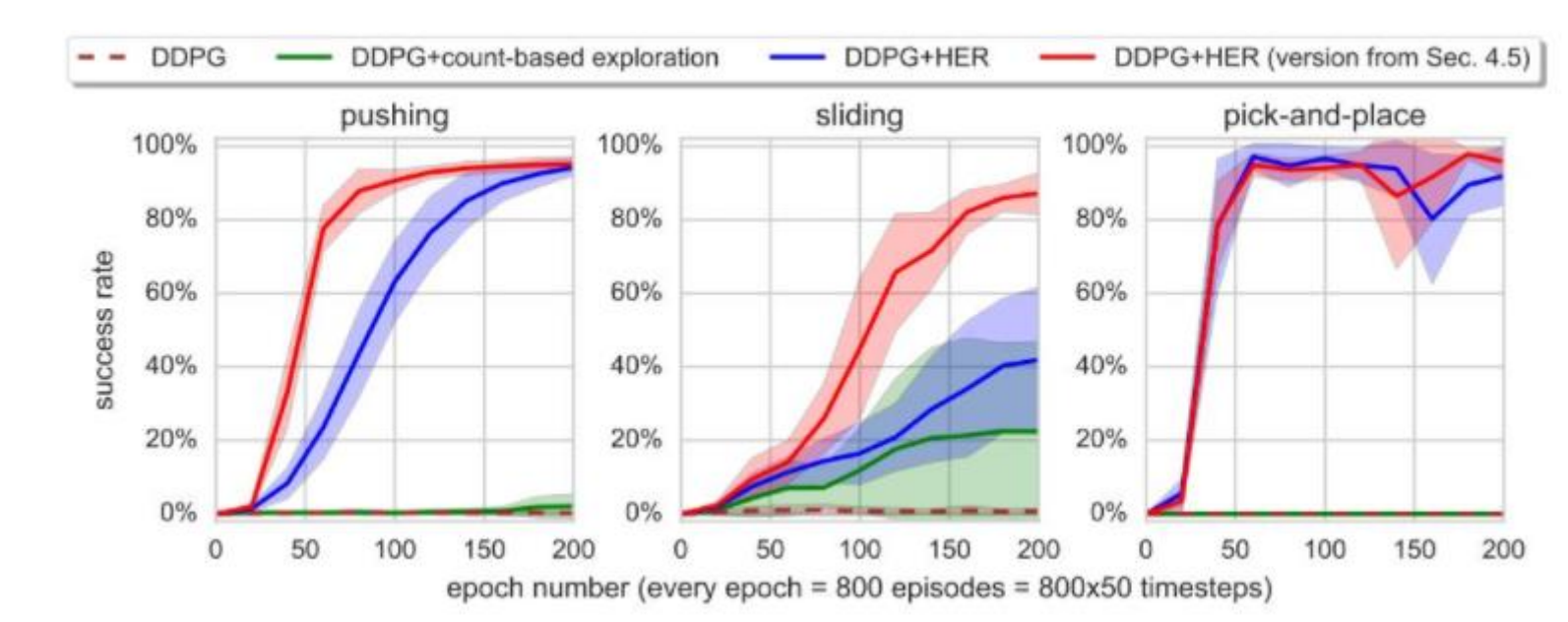
4. random

   episode 관계없이 훈련과정중 아무 랜덤 state를 goal로 한다. 그리고 이를 바탕으로 episode 내 k개의 랜덤 state를 샘플링

성능


no HER
final
random
episode
future
pushing
sliding
pick-and-place
highest success rate
average success rate
1.0
0.8
0.6
0.4
0.2
0.0
1
2
4
8
16
all
number of additional goals used to replay each transition with

성능


DDPG
DDPG+count-based exploration
DDPG+HER
DDPG+HER (version from Sec. 4.5)
pushing
sliding
pick-and-place
success rate
100%
80%
60%
40%
20%
0%
0
50
100
150
200
epoch number (every epoch = 800 episodes = 800x50 timesteps)

성능

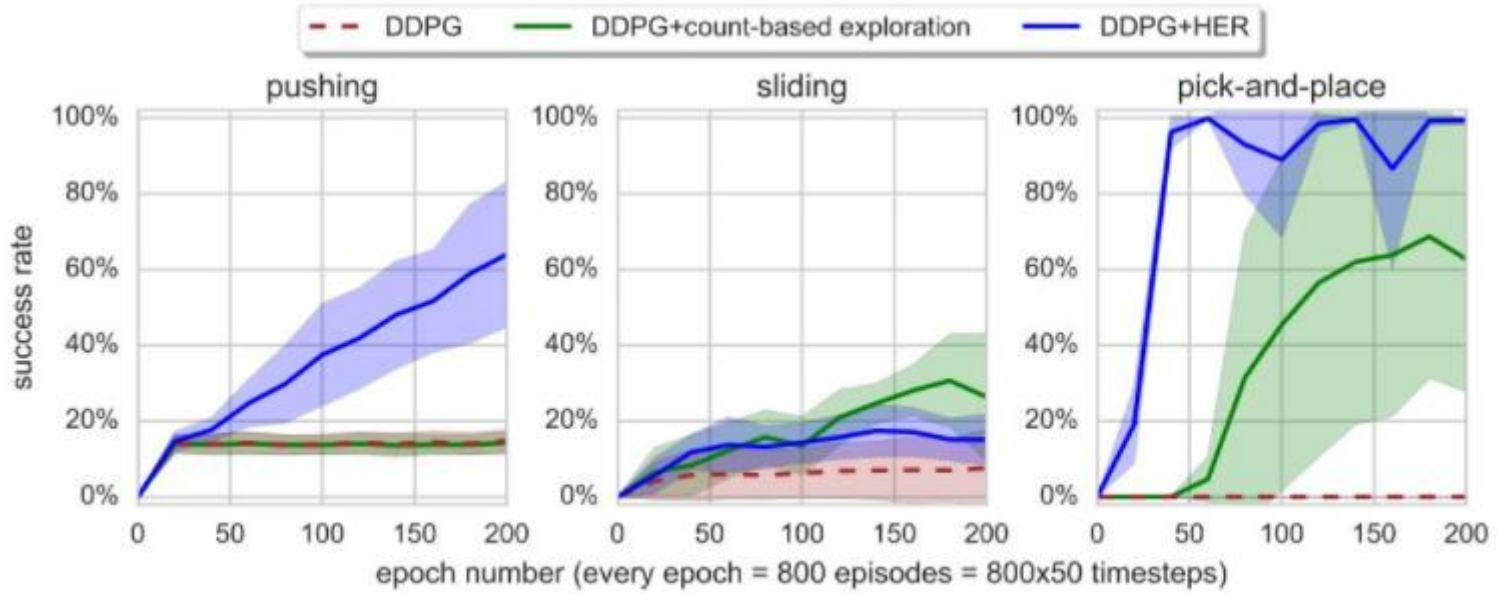


Figure 4: Learning curves for the single-goal case.

Goal 에 따른 보상 설계 +

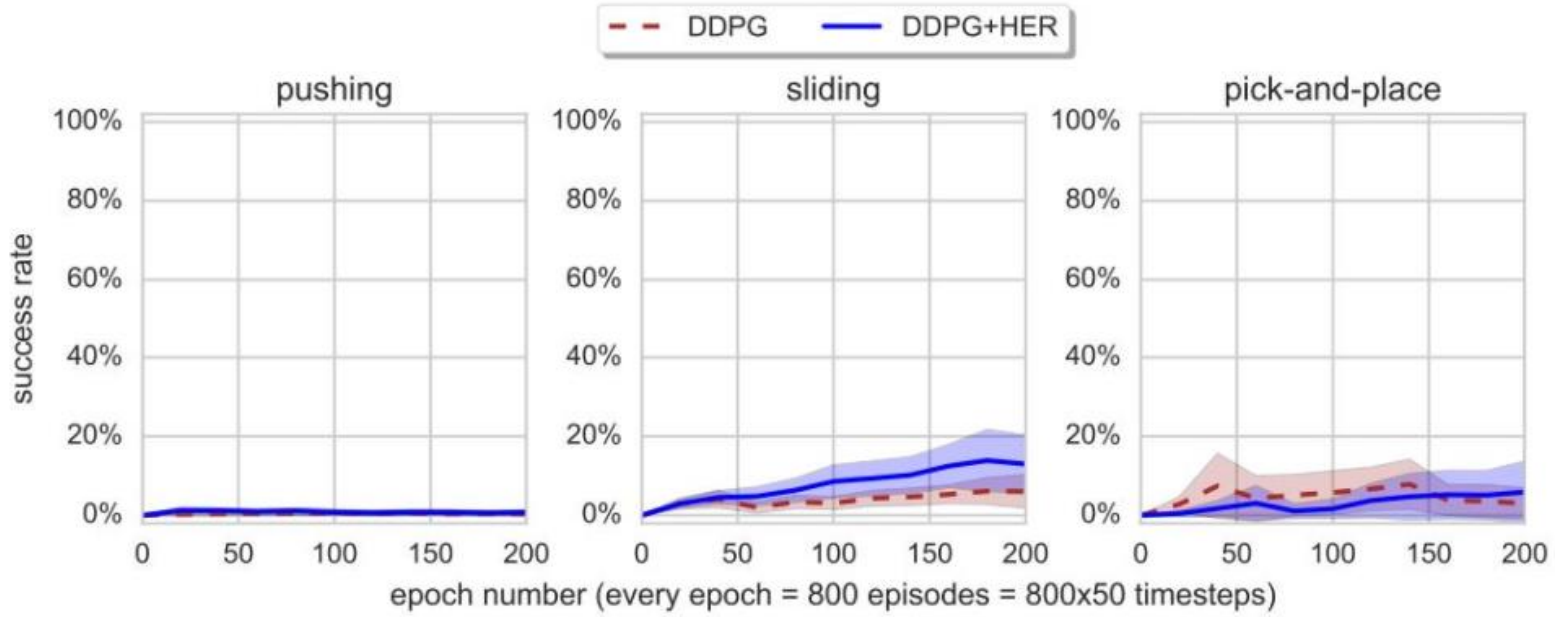


Figure 5: Learning curves for the shaped reward $r(s, a, g) = -|g - s'_{\mathbf{object}}|^2$ (it performed best among the shaped rewards we have tried). Both algorithms fail on all tasks.

Goal 에 따른 보상 설계 +

- 보상함수와 실제 성공간의 괴리가 존재한다
- 패널티가 주어질 경우 아무런 행동을 하지 않는 방향으로 학습되는 경우도 존재