1강 Q-learning

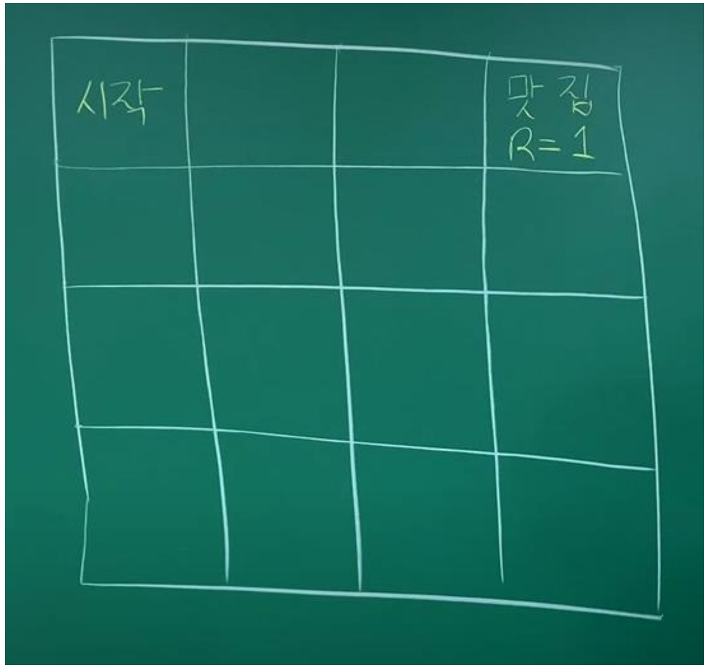
강화학습 연속된 액션을 찾아나가기 <- 보상 줌

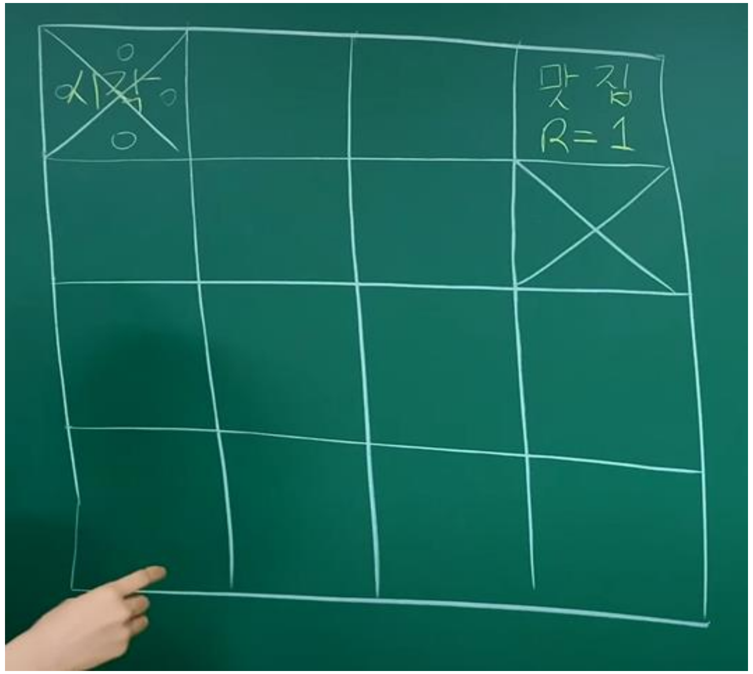
action → action → action → • • •

Goal = maximize Reward

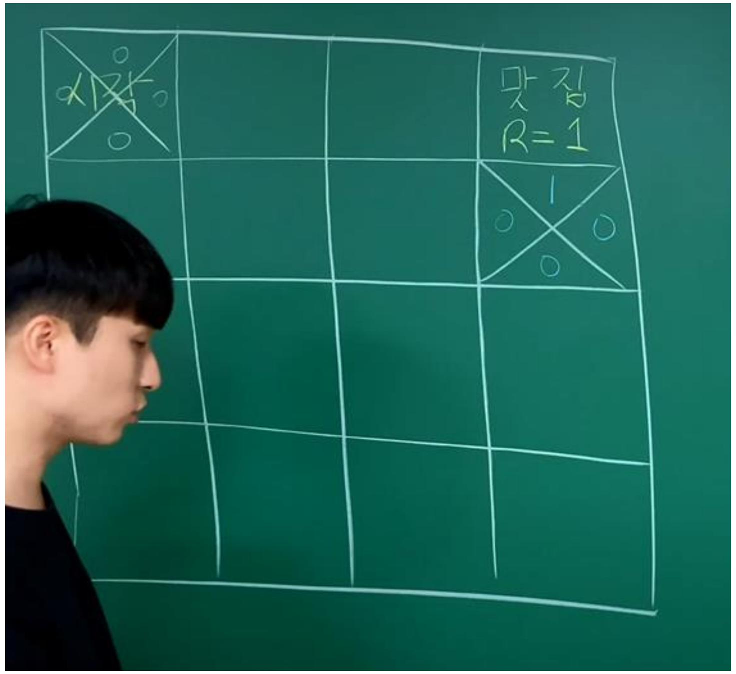
Greedy Action

맛집 찾는 방법

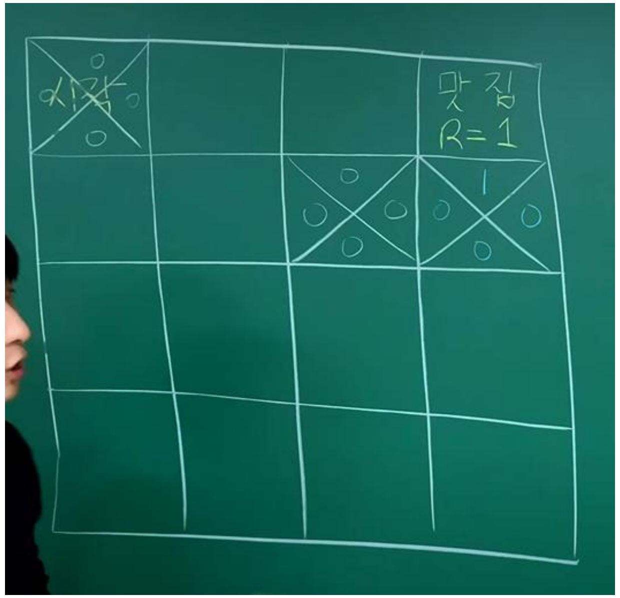


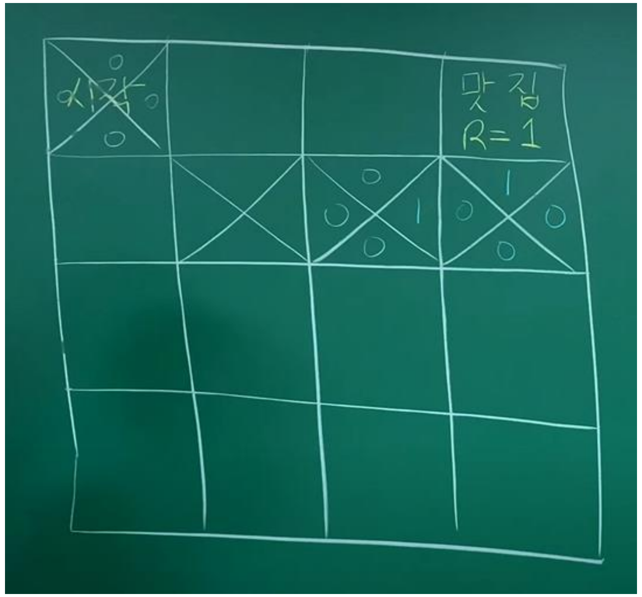


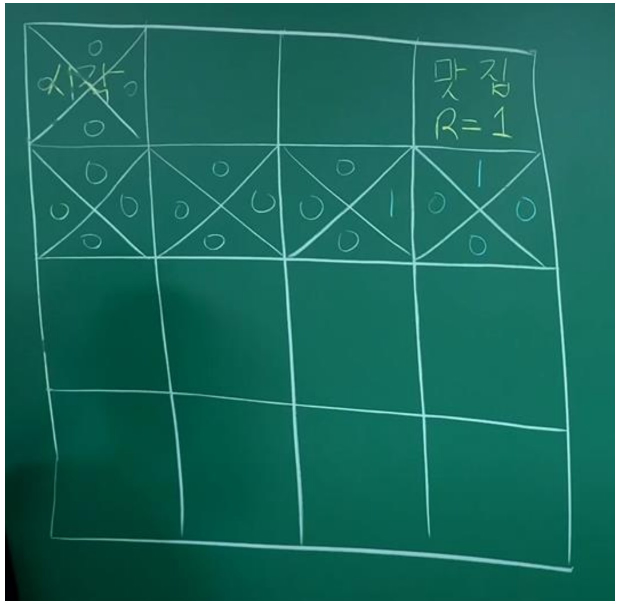
랜덤으로 움직임. 맛집 아래까지 왔다 가정



아래서 위로 갔을 때 맛집 도착 따라서 위쪽 그리드에서 1점이 부여됨



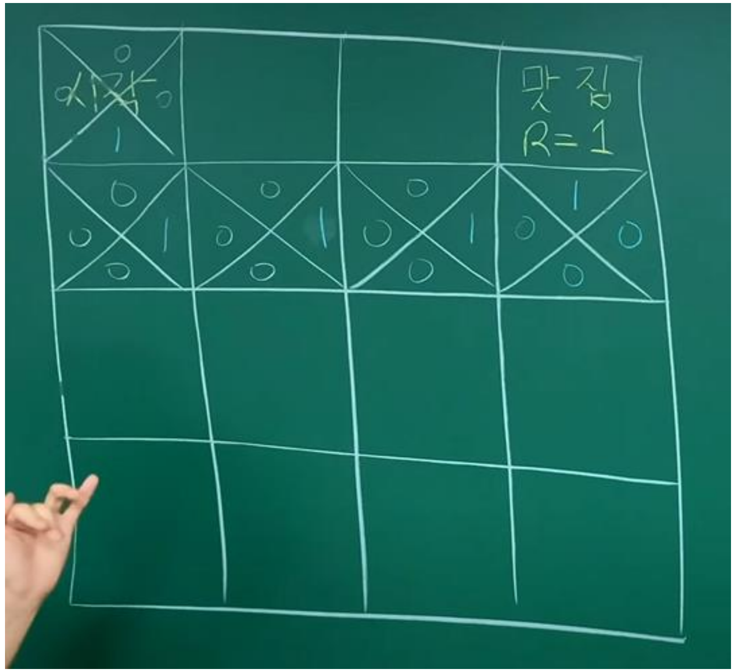




다시 랜덤으로 움직이다 점수 있는 칸 옆으로 감 아직 해당 칸은 다 0점

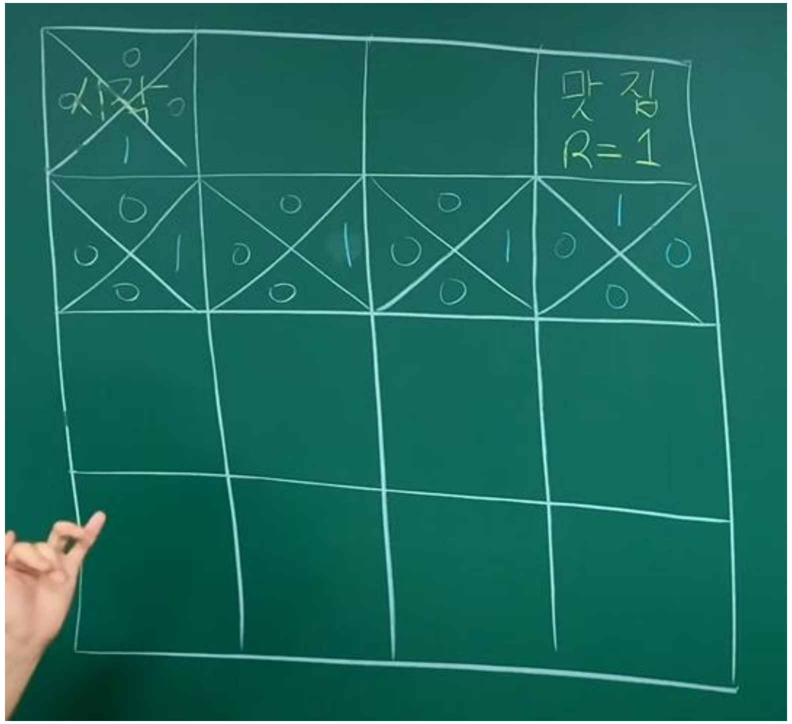
Q-learning 움직이면서 업데이트 오른쪽 칸에 점수가 있으므로 오른쪽 칸으로 움직이게 됨 그리고 오른쪽으로 가면 점수가 있으므로 현재 위치 내 오른쪽 그리드가 1점

반복하다보면 옆 그림처럼 됨 이때 더 좋은 길이 있는데 끝나는 문제점 발생



ε - Greedy

Exploration



<- 이 상황에서 오른쪽으로만 이동하는 길이 더 빠름

따라서 더 좋은 길을 찾기 위해 탐험을 감

ε 만큼은 Random하게, 1-ε 만큼은 Greedy하게 움직인다 (ε은 0~1의 사이의 값)

Exploration(랜덤하게 움직임 = 탐험) vs **Exploitation**(Greedy하게 움직임)

이를 통해서

1. 새로운 path를 찾을 수 있다.
2. 새로운 맛집을 찾을 수 있다.
3. ε이 크면 계속 탐험 / 작으면 계속 Greedy

(decaying)ε - Greedy

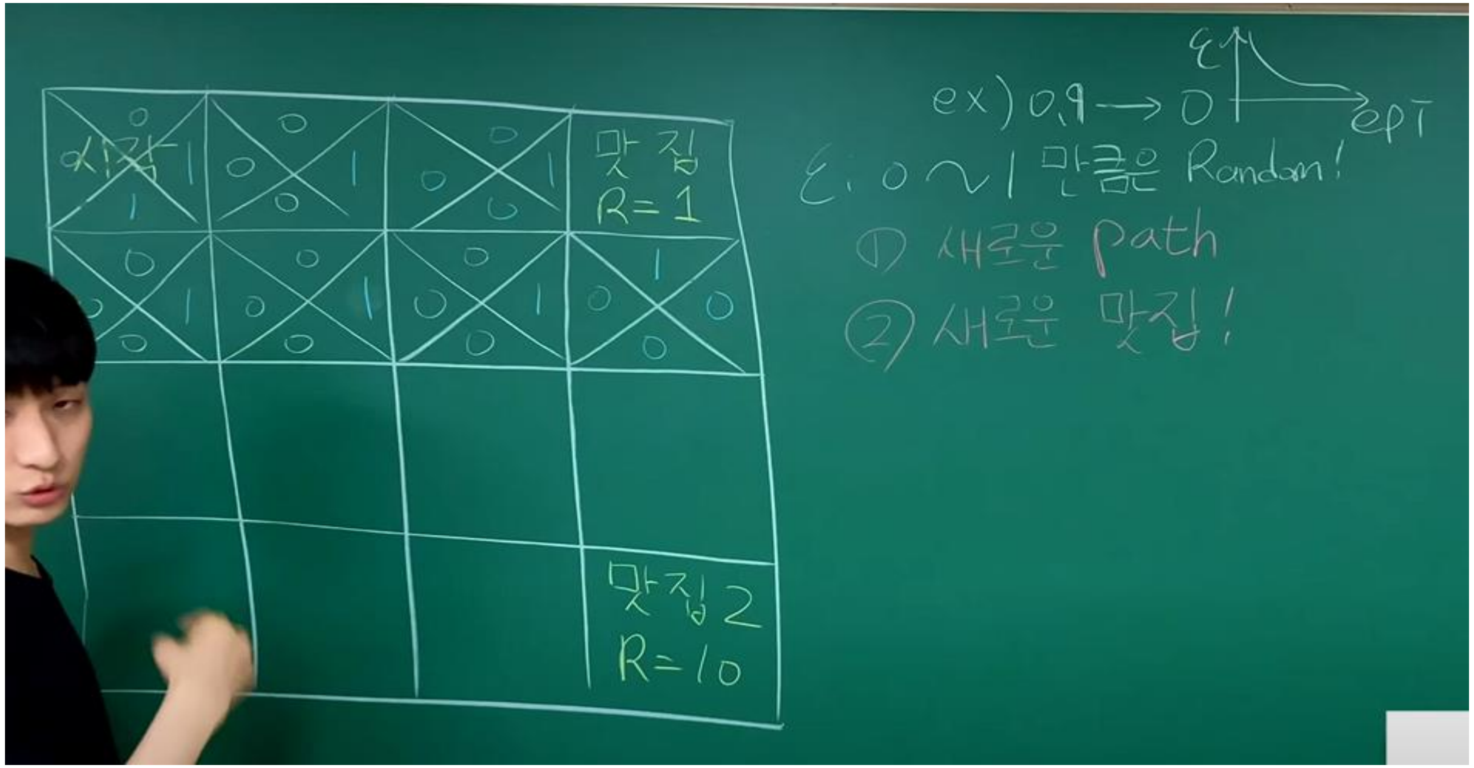
Exploration(랜덤하게 움직임 = 탐험) vs **Exploitation**(Greedy하게 움직임)

➔ 둘 중 하나로 치우지면 안 좋음 그래서 사용하는 방법이 (decaying)ε – Greedy

(decaying)ε – Greedy

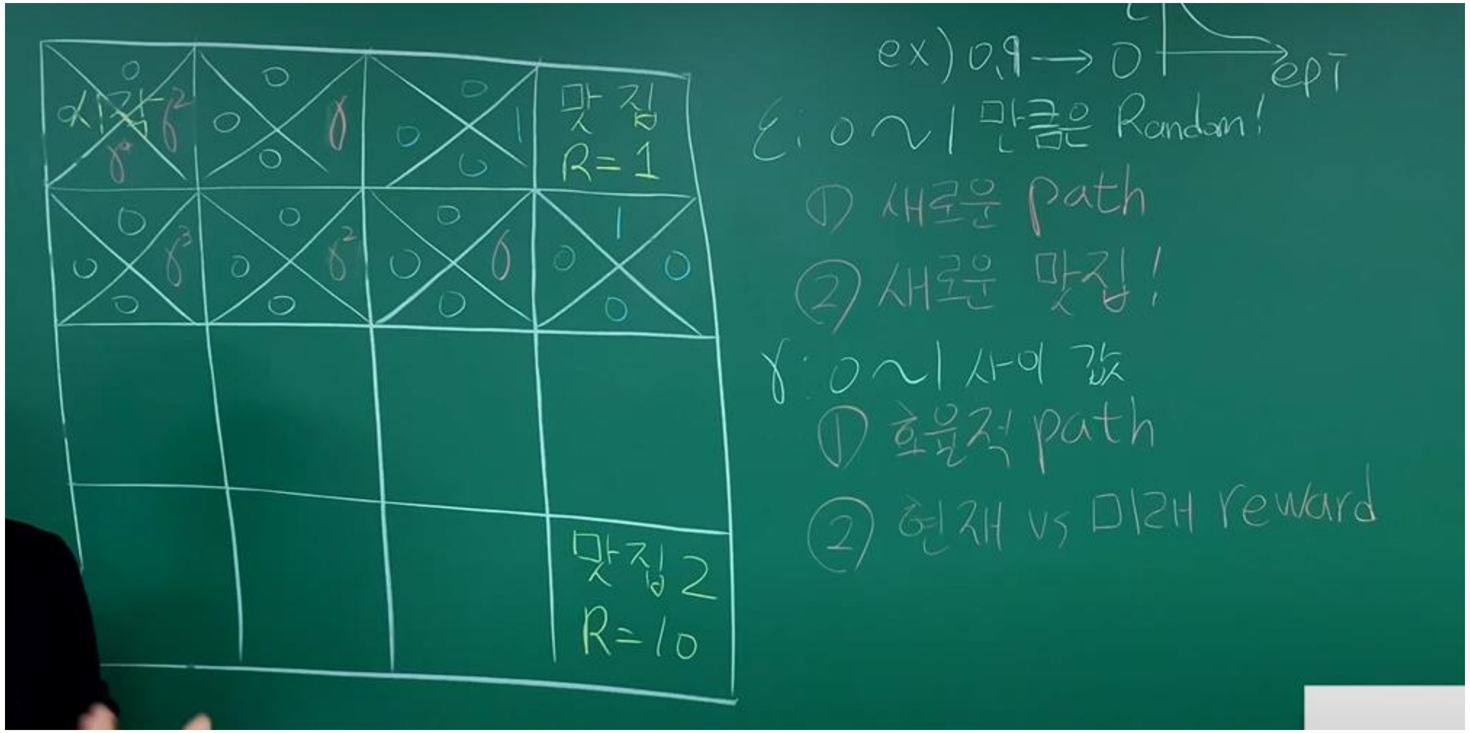
점점 ε값을 줄여가며(탐험을 줄여가며) 학습

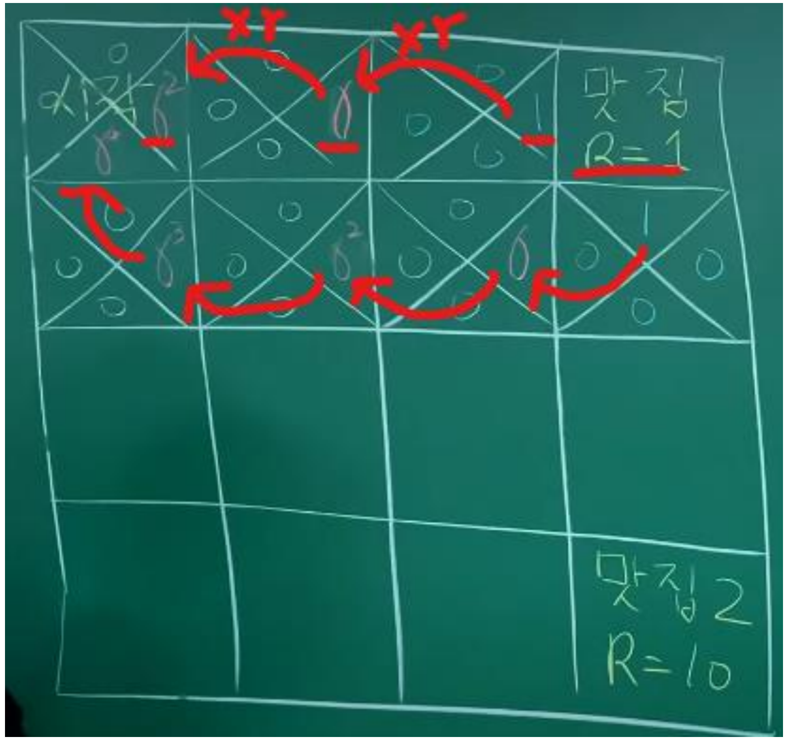
Ex) 처음 0.9로 시작 탐험을 더 많이 했다면 반복하면서 0.8,0.7..0.1 점점 줄여서 탐험 확률을 줄임



<- 첫번째 경로와 두번째 경로 점수 똑같음

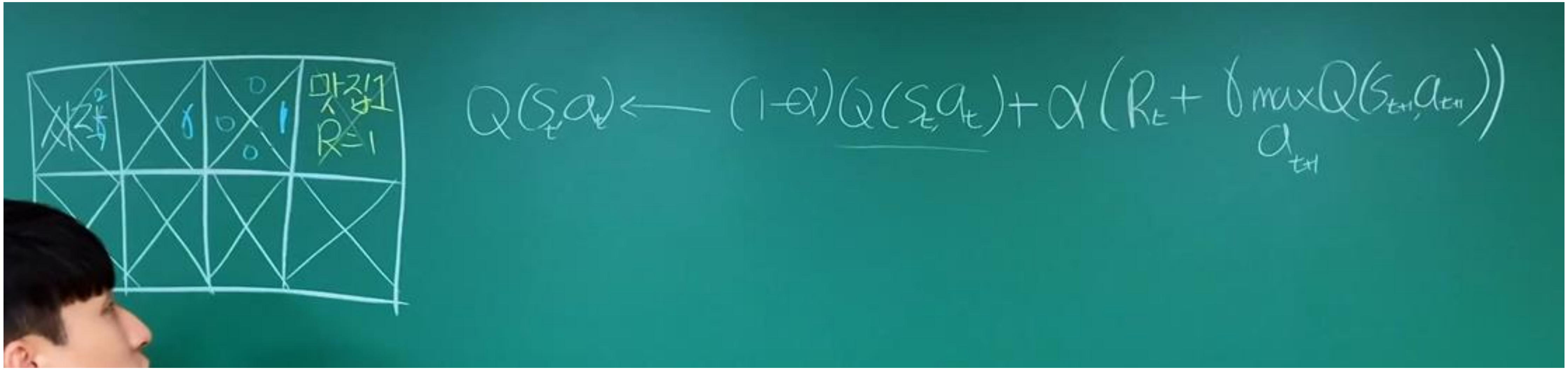
Discount factor





reward 업데이트 할 때 MAX(reward)에서 γ만큼을 곱해서 가져온다(γ는 0~1 사이의 값)
이를 통해서
1. 효율적 path (동일 reward값이라면 경로가 짧은 것)
2. 현재에 관심(γ가 클 때) vs 미래에 관심(γ가 작을 때) reward

Q - Update



$$Q(s_t, a_t) \leftarrow (1-\alpha)\cdot \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \left(\overbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}^{\text{learned value}} \right)$$

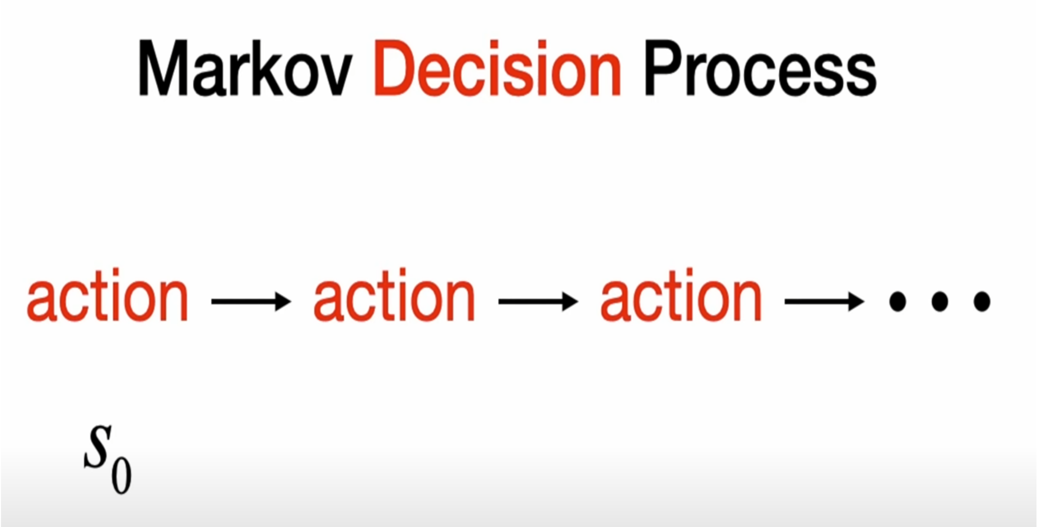
Q - value 는 old value + future value 의 합
α = 새로운 걸(future value) 얼마나 받아 들이냐?

2-1강
Markov Decision Process (MDP)

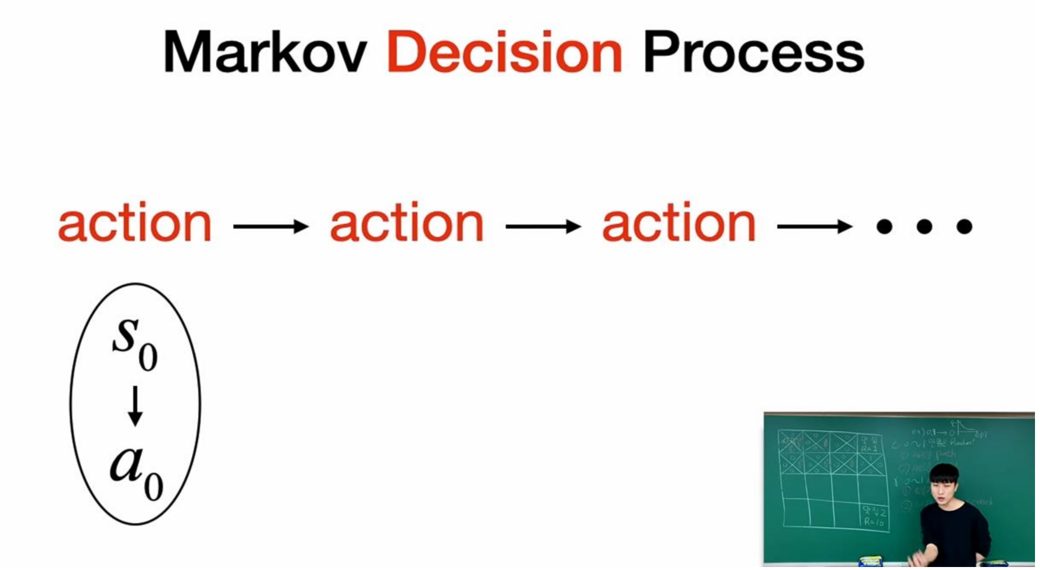
Markov Decision Process

action ⟶ action ⟶ action ⟶ • • •

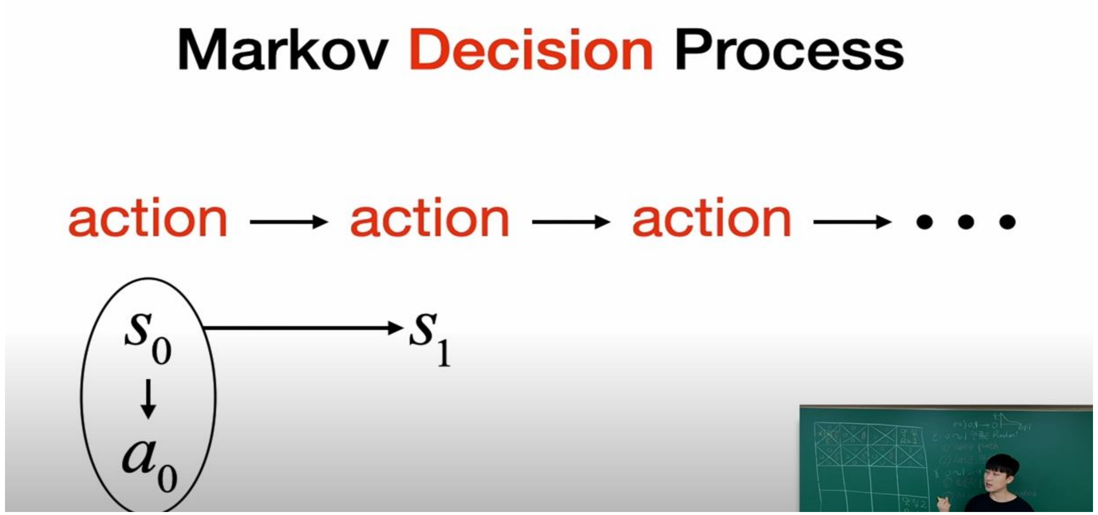
Markov Decision Process -> 액션을 해나가는 프로세스



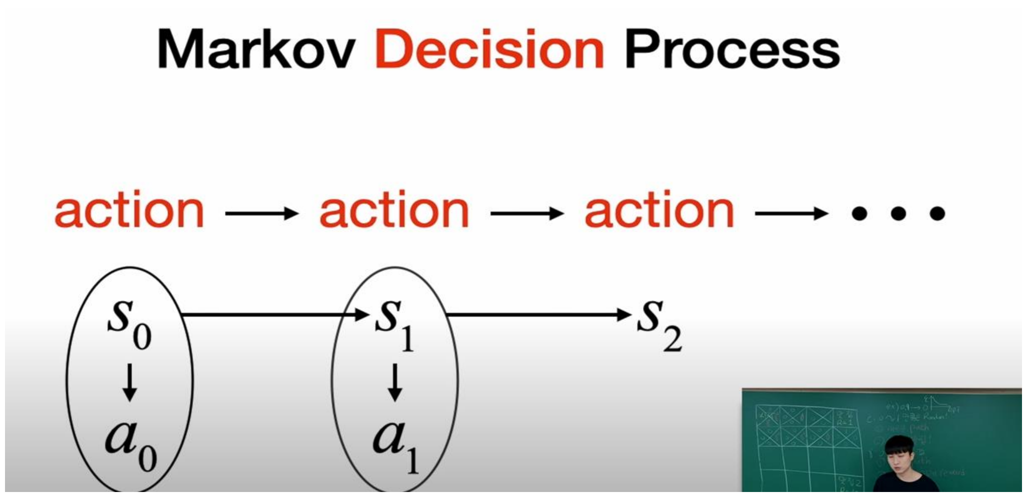
S_0 : 시작 상태 ex (0,0) 이나 맛집 탐방에서 시작 칸

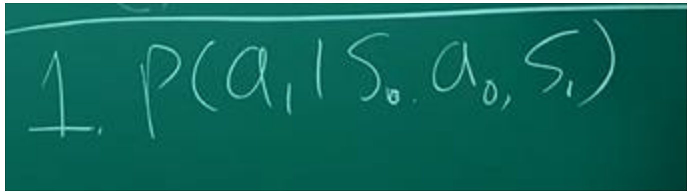


S_0에서 a_0 행동은 함 ex) 맛집탐방 일 때 액션을 한다 그럼 다른 칸으로 움직이게 됨 다른칸 s_1

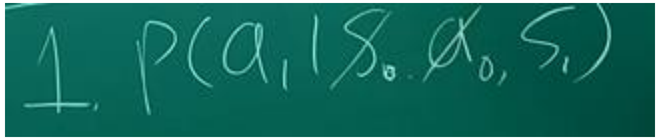


따라서 S_0에서 a_0 행동 -> s_1이 된다.





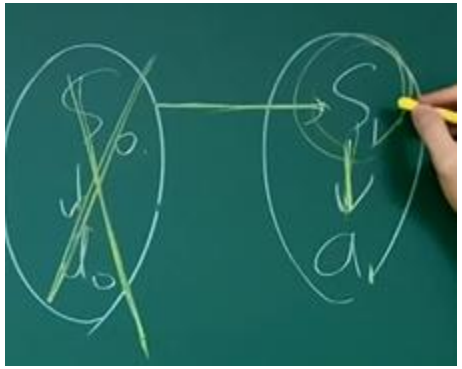
S_0, a_0 , s_1이 주어질 때 a_1 이 일어날 상황(확률)



S_0, a_0 이 필요 없어짐

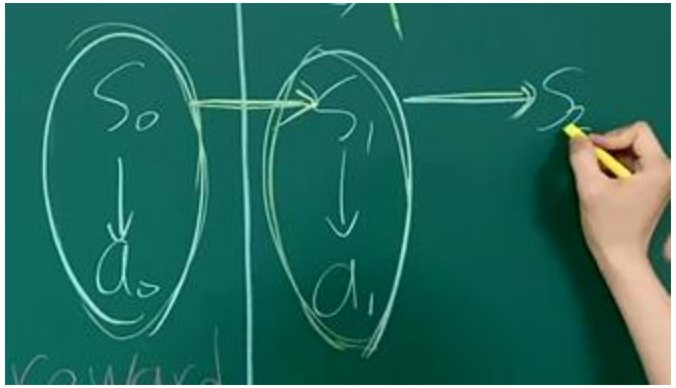
➔ s_1는 S_0, a_0 의해서 넘어간 값

➔ s_1는 S_0, a_0 의해 넘어간 값이기 때문에 S_0, a_0 몰라도 됨



s_1는 S_0, a_0 정보 흡수되어 있음

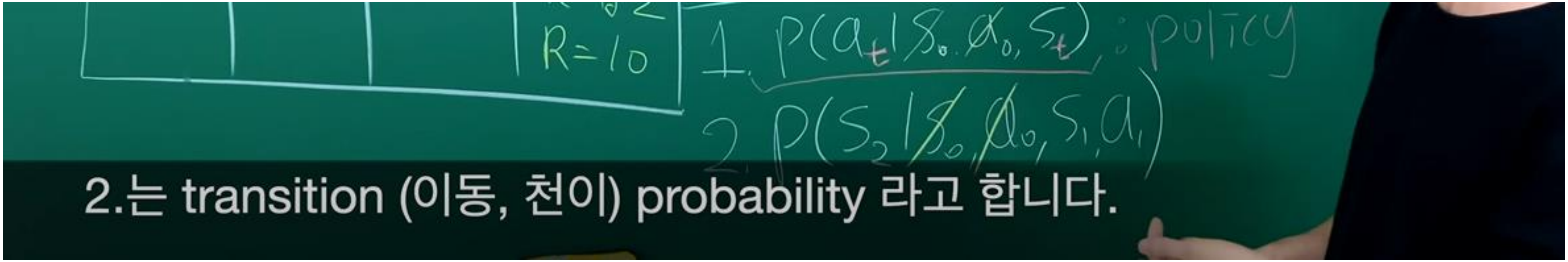
S_0에서 a_0행동 해서 s_1로 넘어가게 되고 s_1에서 a_1 행동했을때 s_2 가 될 상황(확률)



S_0에서 a_0행동 해야 s_1로 넘어갈 수 있 듯 둘은 세트이 세트를 해야 다음 상태로 변함

S_1, a_1 둘 다 남아있음

정리
어떤 액션을 할지 정하는 것이 policy



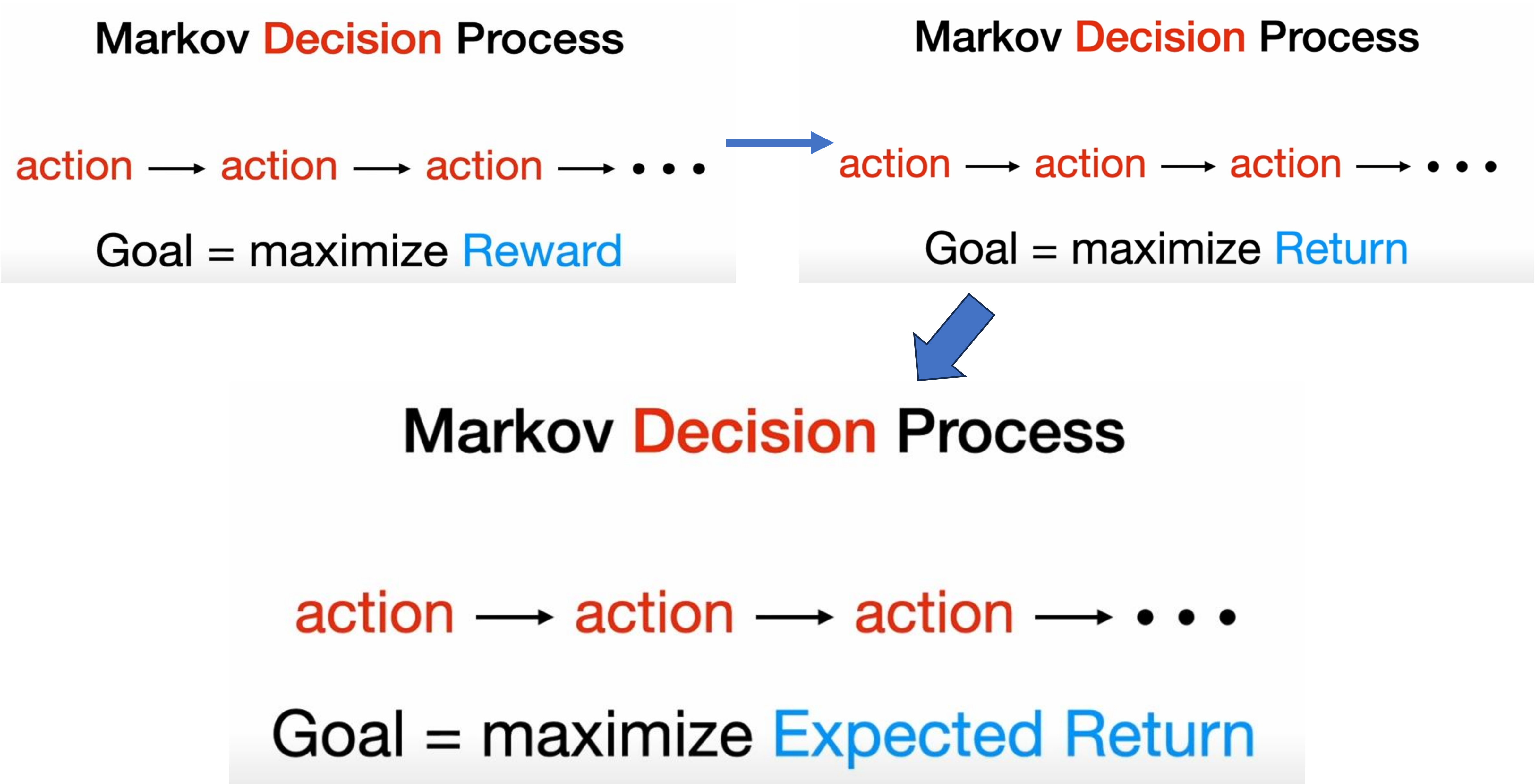
1. $P(a_t|s_{t-1}, a_{t-1}, s^t) = P(a_t|s_t) =$ **Policy**

어떤 상태에서 어떤 액션을 취할까? = Policy
s0, a0와 상관 없다. 왜냐하면 s0, a0는 s1에 모두 담겨 있기 때문

2. $P(s_{t+1}|\ s_{t-1}, a_{t-1}\ , s_t, a_t) = P(s_{t+1}|s_t, a_t)$

=transition probability

Return


Markov Decision Process
action → action → action → • • •
Goal = maximize Reward
Markov Decision Process
action → action → action → • • •
Goal = maximize Return
Markov Decision Process
action → action → action → • • •
Goal = maximize Expected Return

Return $G_t \triangleq R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots$

리턴 = 리워드의 합

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Return Gt는 시간 t이후부터 얻을 수 있는 reward의 합을 의미하며, discount factor γ를 통해 위 식과 같이 정의된다.

결국 평균 return을 maximize하는 policy를 찾는게 강화학습의 목표

2-2강
상태 가치 함수 V & 행동 가치 함수 Q & Optimal policy

강화학습은 expected return을 최대화 하는 것이 목표이고, 그 목표를 달성하는 action을 뱉어주는 policy들을 찾아냄

expected return을 잘 표현하는 2가지 방법인 state value function, action value function

State value function

지금부터 기대되는 Return

따라서 지금 state에 대한 가치 평가를 내려줌

Action value function

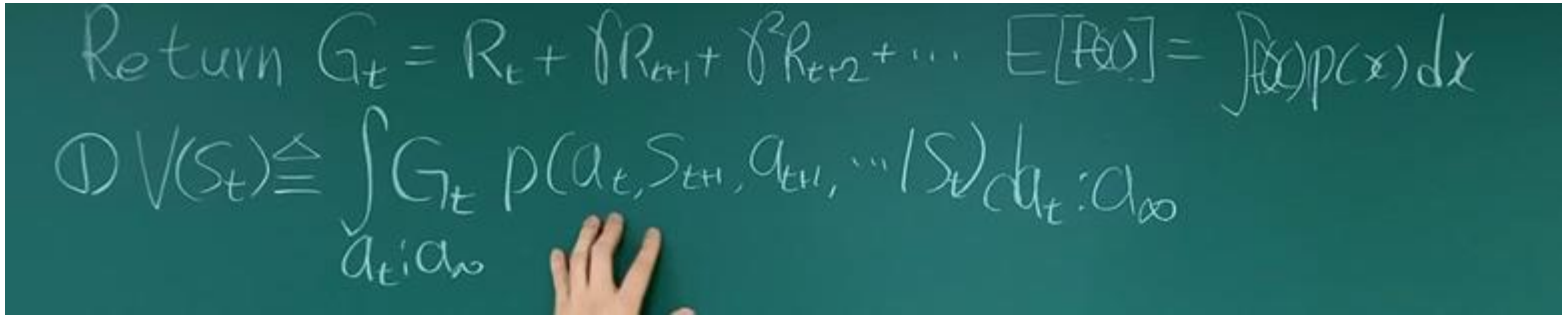
지금 행동으로부터 기대되는 Return

지금 이 state에서의 어떤 행동으로부터 기대되는 Return

Optimal policy

State Value Function 즉 지금부터 기대되는 Return를 최대화 하는것

state value function



행동 하나하나는 이산적인 과정이지만 그 경우의 수는 셀 수 없을 만큼 많기 때문에 기대값 E를 계산할 때에 적분을 한다.
지금 상태에서의 at행동의 경우의 수 중에 하나를 뽑고, 그 다음 st+1상태에서 가능한 at+1경우의 수 중에서 하나를 뽑고... 이것을 무한대로 이어나가는 경우의 수들 각각의 확률을 구해서 리턴과 곱한 뒤 적분해주면 svf가 나온다.

action value function

$$② \; Q(s_t, a_t) \triangleq \int_{s_{t+1}:a_\infty}^{a_t:a_\infty} G_t \, p(s_{t+1}, a_{t+1}, s_{t+2}, a_{t+2}, \cdots \mid s_t, a_t) \, ds_{t+1}:a_\infty$$

현재 state에서 어떤 행동을 했을때부터 쭉 진행한 기대되는 return

Optimal Policy

Return $G_t = R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots$ $\quad E[f(x)] = \int f(x)p(x)\,dx$

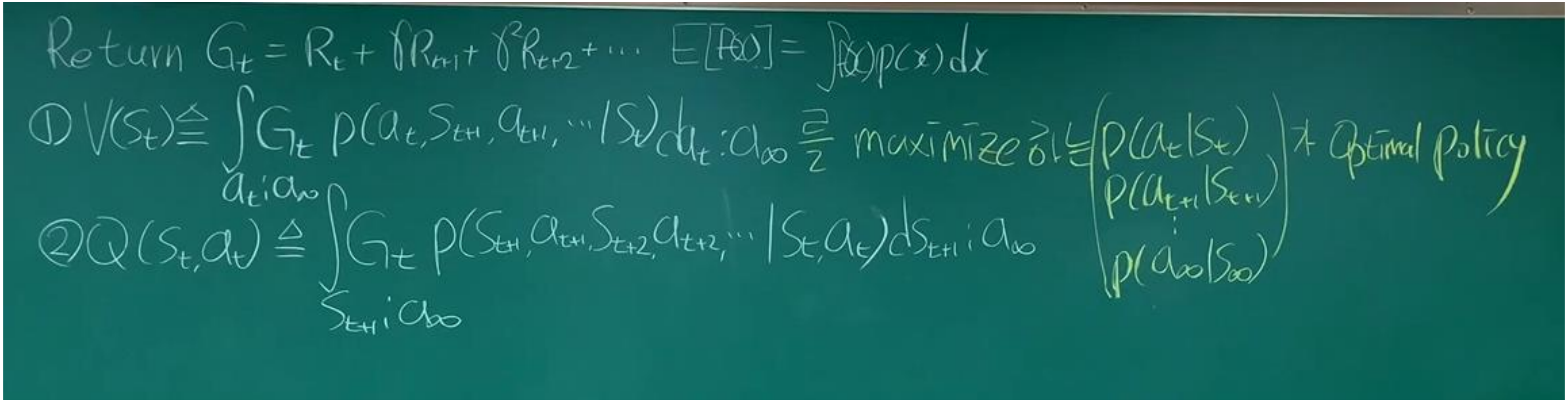
① $V(S_t) \triangleq \int_{a_t:a_\infty} G_t \, p(a_t, S_{t+1}, a_{t+1}, \cdots \mid S_t) \, da_t:a_\infty$ 를 maximize 하는 $\begin{pmatrix} p(a_t \mid S_t) \\ p(a_{t+1} \mid S_{t+1}) \\ \vdots \\ p(a_\infty \mid S_\infty) \end{pmatrix}$ 가 Optimal Policy

② $Q(S_t, a_t) \triangleq \int_{S_{t+1}:a_\infty} G_t \, p(S_{t+1}, a_{t+1}, S_{t+2}, a_{t+2}, \cdots \mid S_t, a_t) \, dS_{t+1}:a_\infty$

V(St)를 maximize하는 action들의 모임

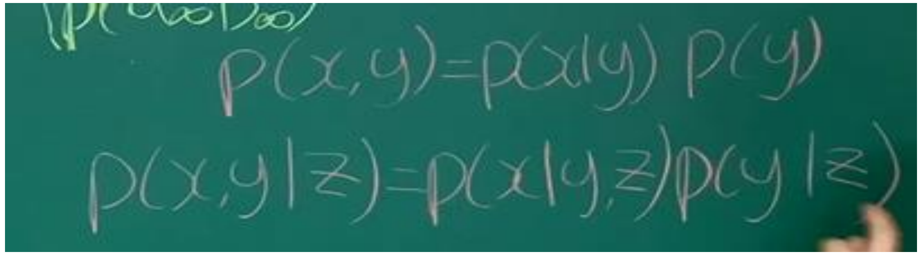
2-3강. 벨만 방정식
(Bellman equation)

Bellman equation

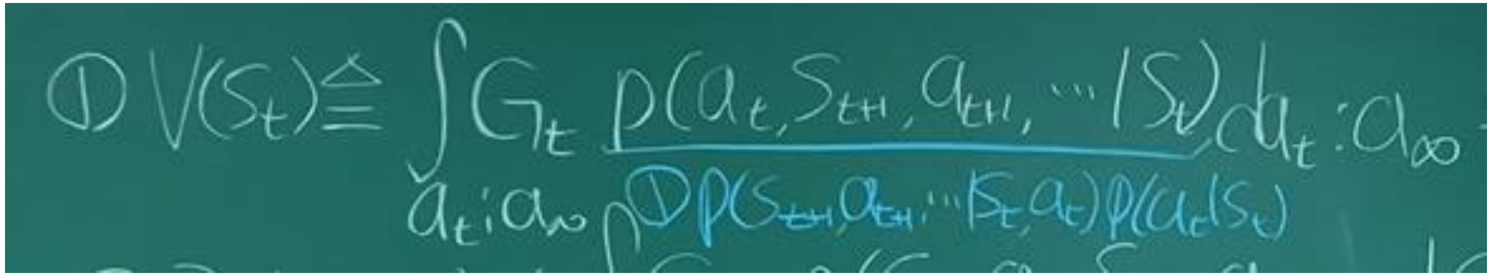


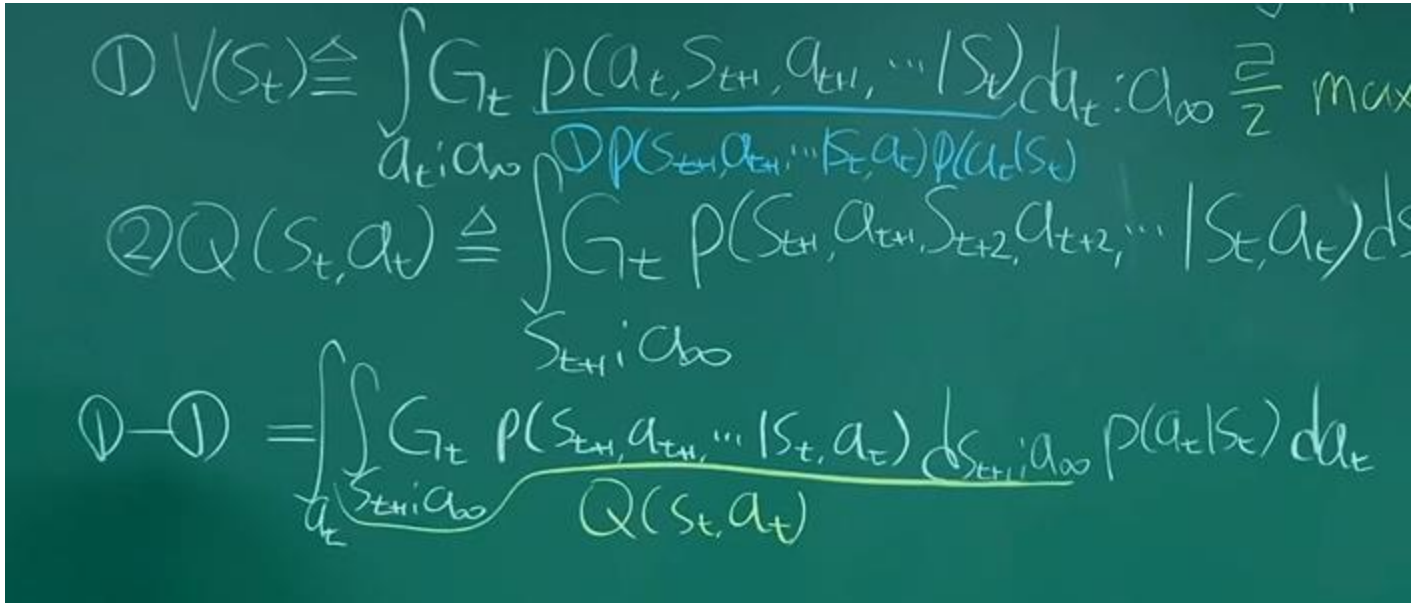
V라는 것을 Q라고도 표현할 수 있고, 이를 바꿔서도 가능하고
다음 t+1로도 표현할 수 있게 해주는 방정식

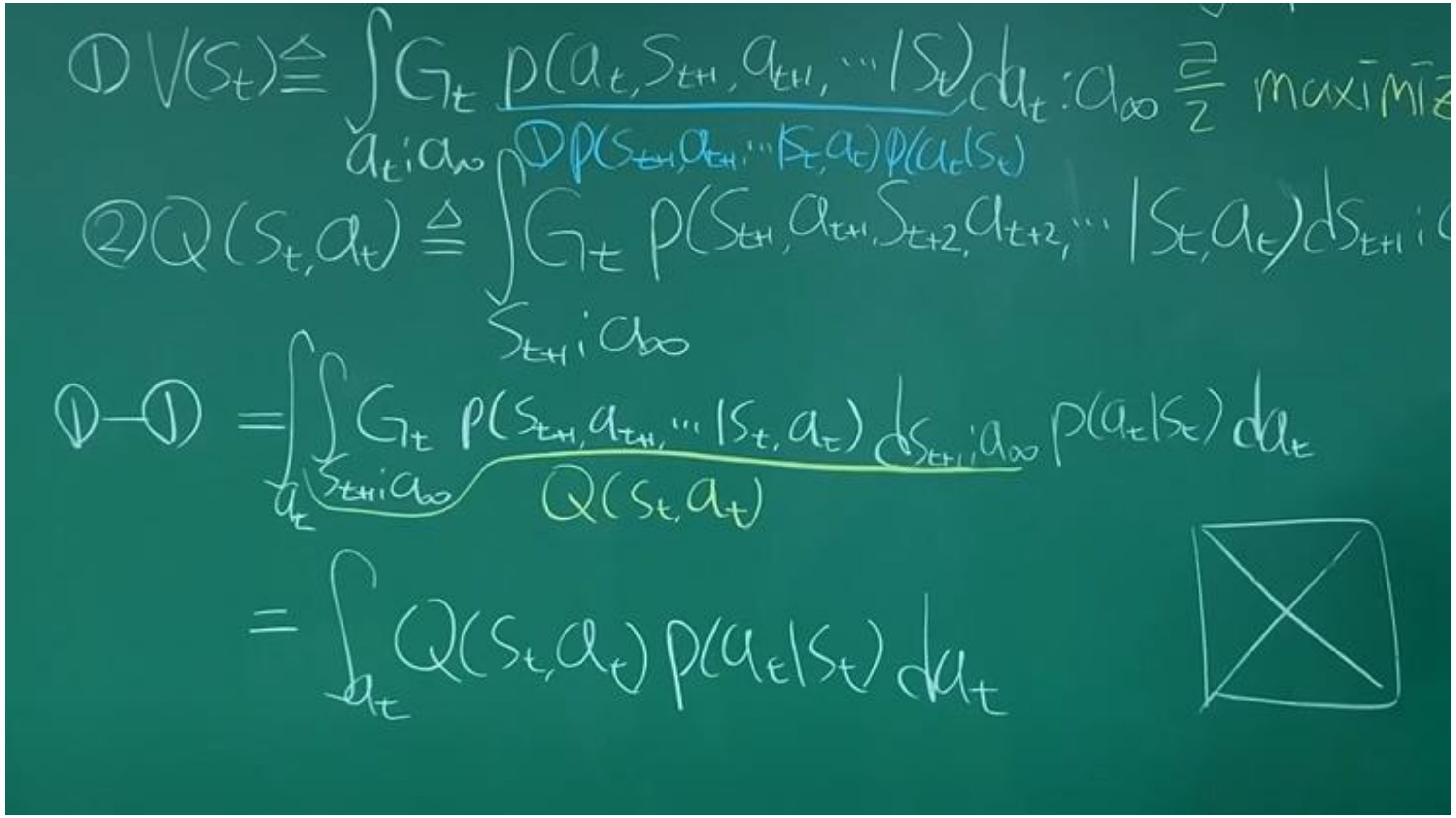
함수 뒤에 뺄 수 있음 **Bayes rule 적용하기**



위 방법으로 a_t를 뺌



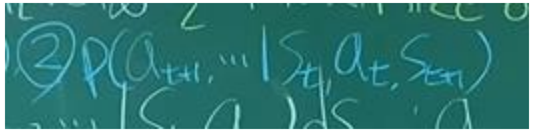




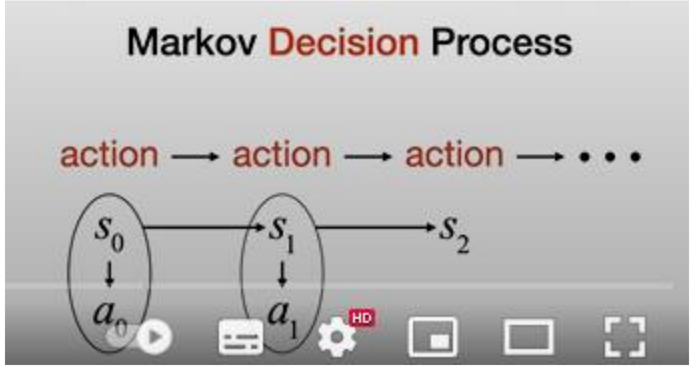
V to Q

State에서 기대되는 return 값은 아래처럼 이해 가능

state안의 action들의 Q값을 평균을 내는 것.

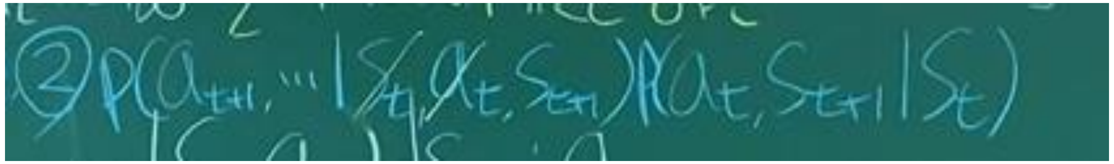


2번째 a_t와 s_t+1도 같이 뺌



Markov Decision process에 따라 St+1은 st at모두의 값을 가지고 있으므로 이 둘을 생략이 가능

전체 식



$$①-② = \int_{a_t, s_{t+1}} \int_{a_{t+1}:a_\infty} (R_t + \gamma G_{t+1})\, p(a_{t+1}, \cdots | s_{t+1})\, da_{t+1}:a_\infty\, p(a_t, s_{t+1} | s_t)\, da_t, s_{t+1}$$

위 감마G_t+1 어디서 ->Gt를 옆과 같이 다르게 표현가능

$$\text{Return } G_t = R_t + \underbrace{\gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots}_{\gamma G_{t+1}}$$

$$① V(s_t) \triangleq \int_{a_t:a_\infty} G_t\, p(a_t, s_{t+1}, a_{t+1}, \cdots | s_t)\, da_t:a_\infty$$ 를 maximize 하는 policy 가 optimal policy

$$① p(s_{t+1}, a_{t+1}, \cdots | s_t, a_t)\, p(a_t|s_t) \quad ② p(a_{t+1}, \cdots | s_t, a_t, s_{t+1})\, p(a_t, s_{t+1} | s_t)$$

$$② Q(s_t, a_t) \triangleq \int_{s_{t+1}:a_\infty} G_t\, p(s_{t+1}, a_{t+1}, s_{t+2}, a_{t+2}, \cdots | s_t, a_t)\, ds_{t+1}:a_\infty$$

$$p(x,y) = p(x|y)\, p($$

$$p(x,y|z) = p(x|y,z)\, p($$

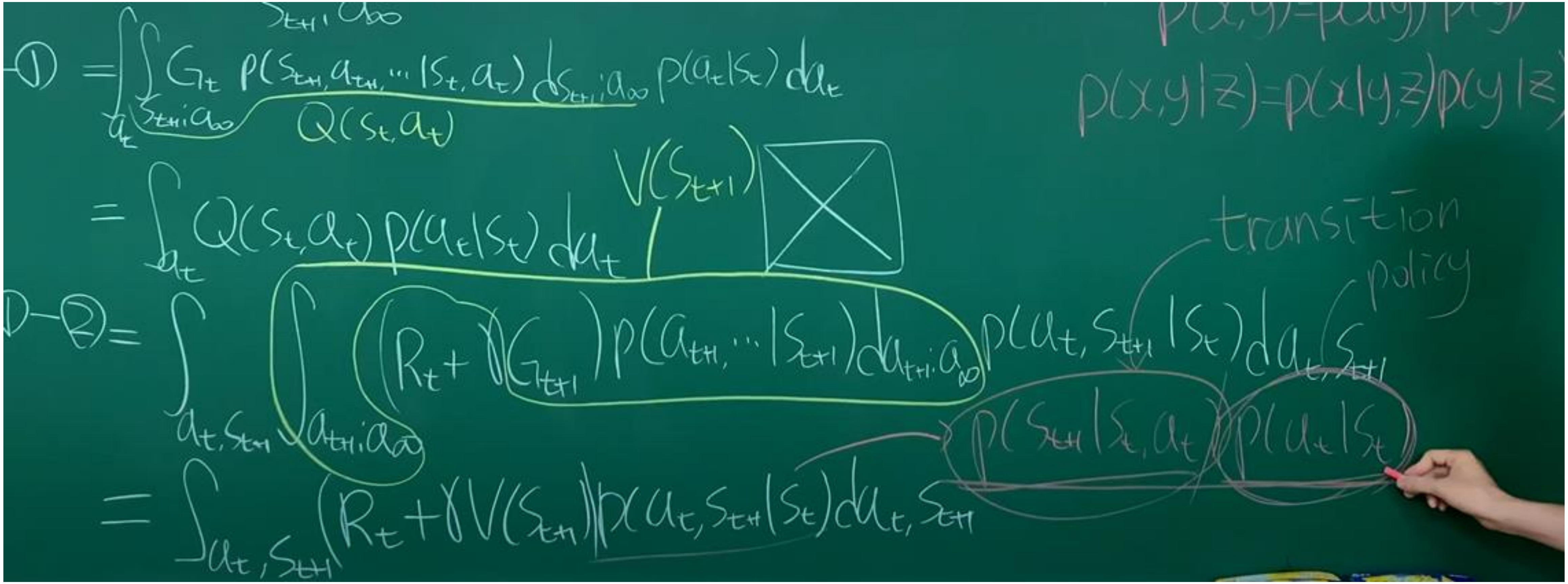
$$①-① = \int_{a_t} \underbrace{\int_{s_{t+1}:a_\infty} G_t\, p(s_{t+1}, a_{t+1}, \cdots | s_t, a_t)\, ds_{t+1}:a_\infty}_{Q(s_t, a_t)}\, p(a_t|s_t)\, da_t$$

$$= \int_{a_t} Q(s_t, a_t)\, p(a_t|s_t)\, da_t$$

$$V(s_{t+1})$$

$$①-② = \int_{a_t, s_{t+1}} \int_{a_{t+1}:a_\infty} (R_t + \gamma G_{t+1})\, p(a_{t+1}, \cdots | s_{t+1})\, da_{t+1}:a_\infty\, p(a_t, s_{t+1} | s_t)\, da_t, s_{t+1}$$

$$= \int_{a_t, s_{t+1}} (R_t + \gamma V(s_{t+1}))\, p(a_t, s_{t+1} | s_t)\, da_t, s_{t+1}$$

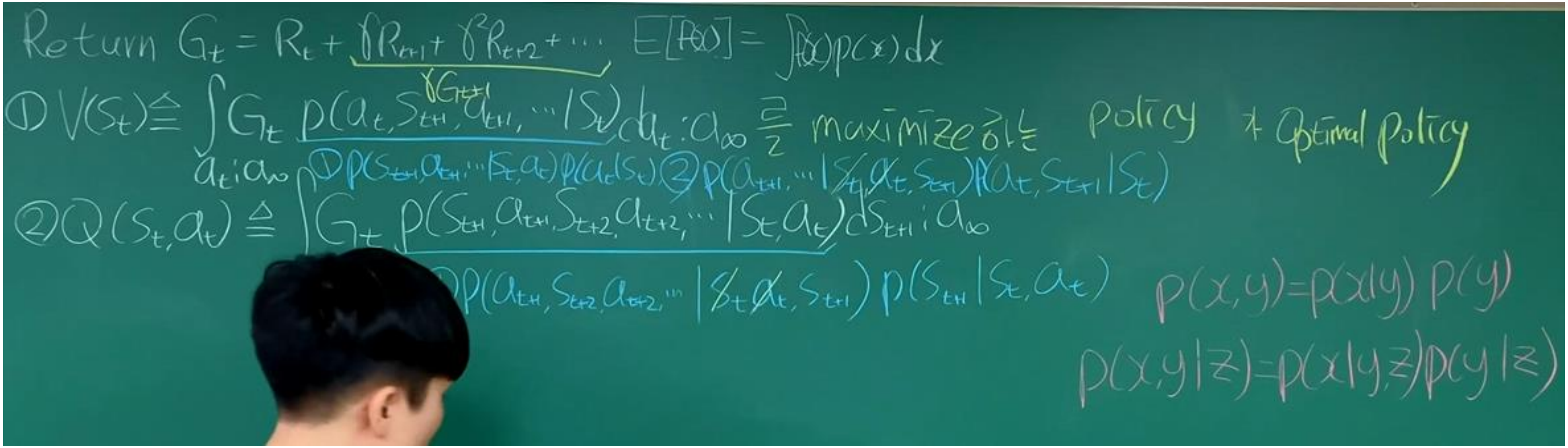


아래식처럼 최대화 하는 policy 구하는 것이 중요한데 위 적분식 다른게 표현한 식(빨강색)을 보면 policy 구할 수 있게됨

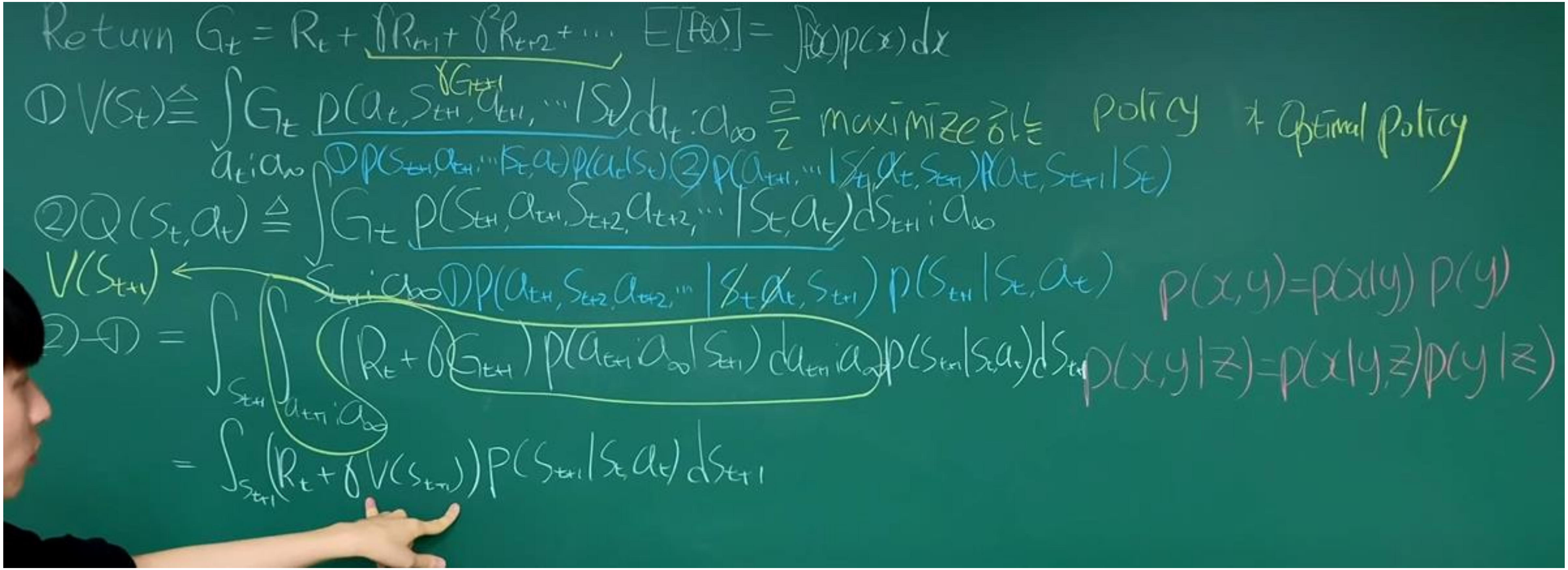
Q value Bellman equation

Q to Vst+1

Q를 next V로 표현이 가능, 숨어있는 V를 찾기

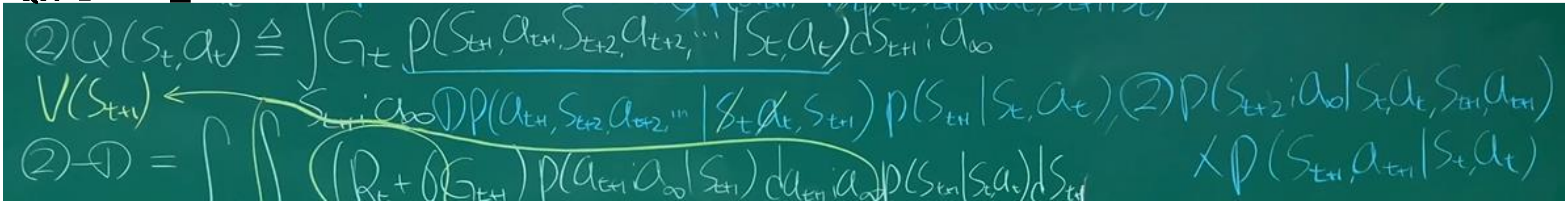


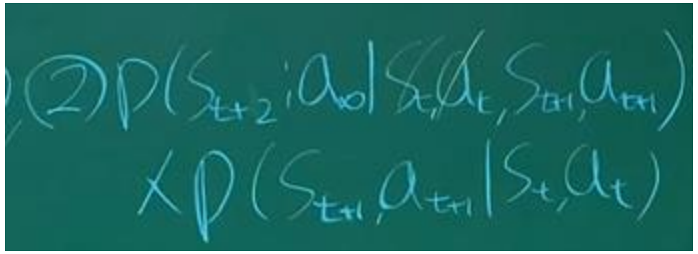
Markov Decision process에 따라 St+1은 st at모두의 값을 가지고 있으므로 이 둘을 생략이 가능

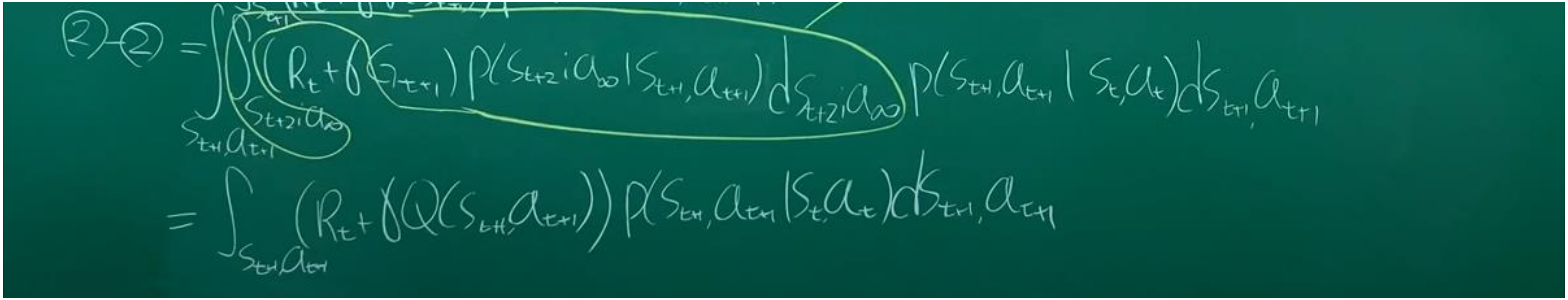


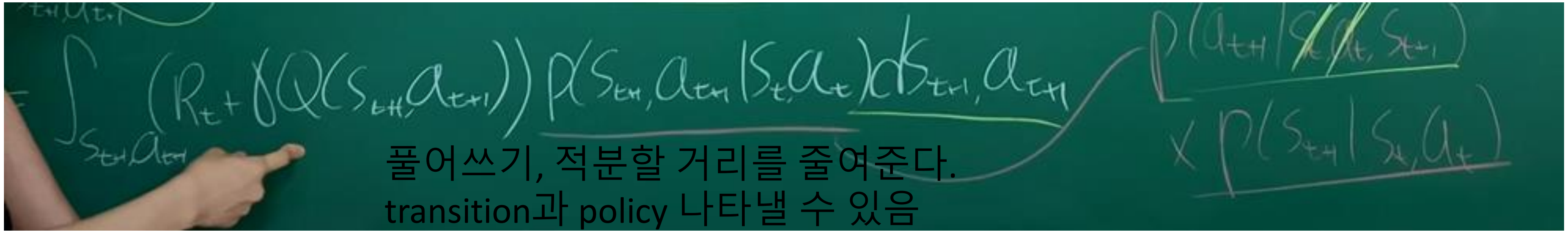
마지막 식 Q를 next V로 표현된걸 확인 가능

Qst+1로 표현









풀어쓰기, 적분할 거리를 줄여준다.
transition과 policy 나타낼 수 있음