SWCON Capstone Design

Physics Based Animation

Synthesis Weapon Attack Animation using Reinforcement Learning

2017103710 김경민

Subject

Weapon attack animation synthesis using deep reinforced learning

- 캐릭터의 걷거나 뛰는 애니메이션은 만들어진 몇 개의 애니메이션을 조합, IK를 적용하여 자연스럽게 표현 가능

- 반면, 공격 애니메이션은 '~를 공격한다' 라는 행동을 행하는 주체뿐이 하니라 공격을 받는 대상, 사용하는 무기에 큰 영향을 받는다.

- 따라서, 무기에 따라서, 시작(준비) 자세에 따라서, 공격 목표 위치에 따라서 Interactive한 Weapon Attack Animation을 합성하려 한다.

Challenge : Adaptive, Realistic

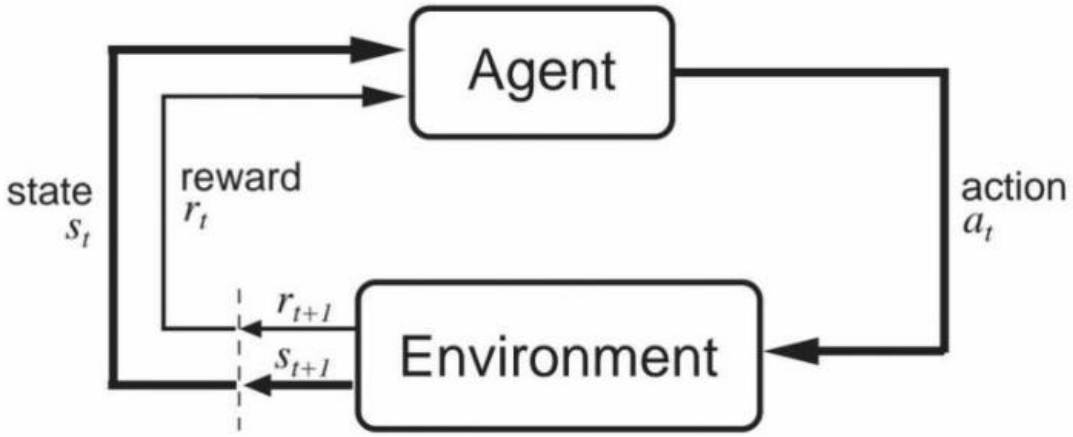
Solution 1 : Inverse Kinematics

- 무기와 팔 길이를 더한 값을 반경으로 하여 타격 지점까지의 반원을 무기 끝 부분이 따라가게 한다.
- 타격 지점을 향해 정확하게 무기를 휘두를 수는 있겠지만, 모션이 부자연스럽다.

Solution 2 : Data driven animation synthesis

- 모션 캡처 데이터를 활용해 신경망을 학습시키고 타격 지점과 무기 무게, joint rotation trajectory를 input으로 두고 자연스러운 애니메이션을 출력한다.
- Reference : **Neural State Machine for Character-Scene Interactions** (2019, ACM Transactions on Graphics)
- 다만, 영향을 끼치는 요소를 추가할수록 각각의 경우에 대한 데이터를 준비해야 하므로 자연스러운 모션을 생성하기 위해서는 매우 많은 학습데이터가 필요하다.

Proximal Policy Optimization



Reinforcement Learning

- State : Agent가 관측하는 환경 정보
- Action : 관측한 정보를 기반으로 추론한 결과
- Reward : 추론한 Action에 대한 보상

- Step : 이전 Action을 받고 State, Reward를 보내는 한 주기
- Episode : Agent가 이루고자 하는 목표가 성공, 실패, 혹은 정해진 시간이 끝나서 환경이 Reset되는 주기

- 환경값을 어떻게 받아올 건인지
- Action을 추론할 정책(Policy)을 어떻게 정할 것인지
- 정책이 옳은가를 어떻게 평가할 것인지

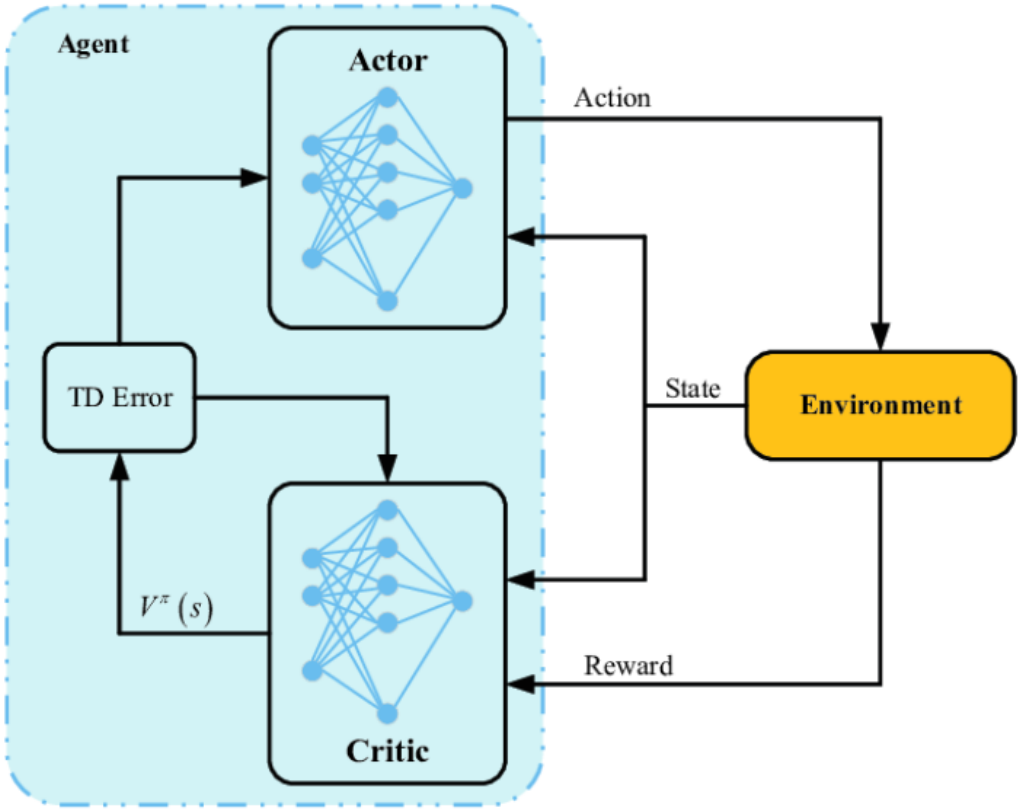
Proximal Policy Optimization

Actor-Critic Algorithm

- 환경값을 어떻게 받아올 건인지
- Action을 추론할 정책(Policy)을 어떻게 정할 것인지 -> Actor (Policy Network)
- 정책이 옳은가를 어떻게 평가할 것인지 -> Critic (Value Network)

Proximal Policy Optimization

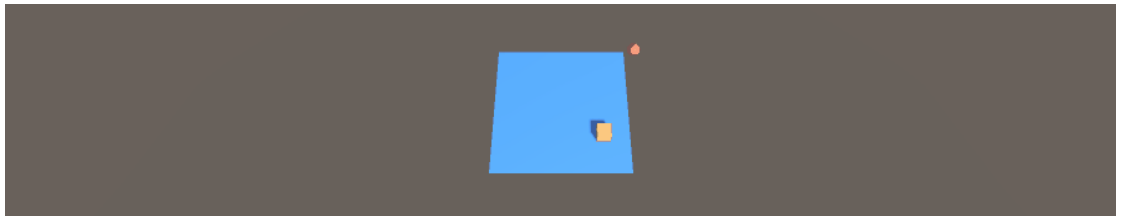
- 다른 강화학습에 비해 안정적이고 계산이 덜 복잡하며,
  많은 상황에 적용이 가능해 아직도 주류가 되는 알고리즘

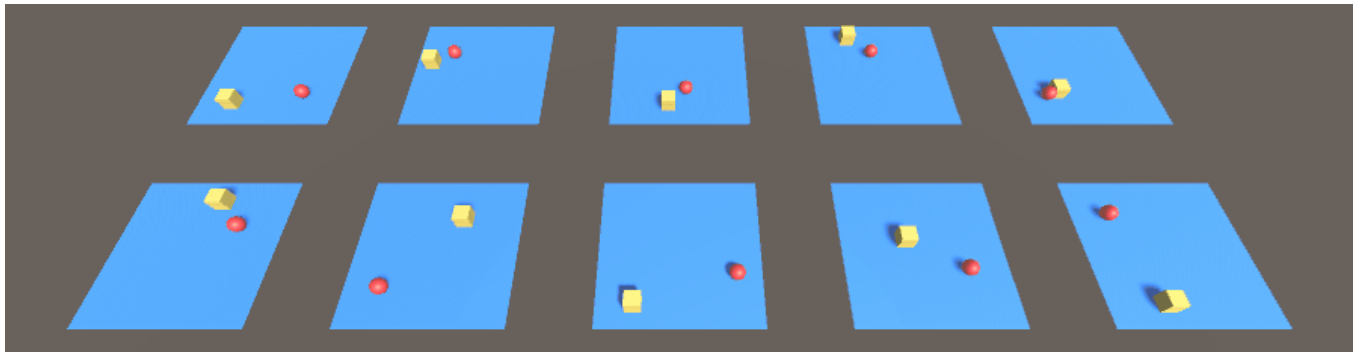


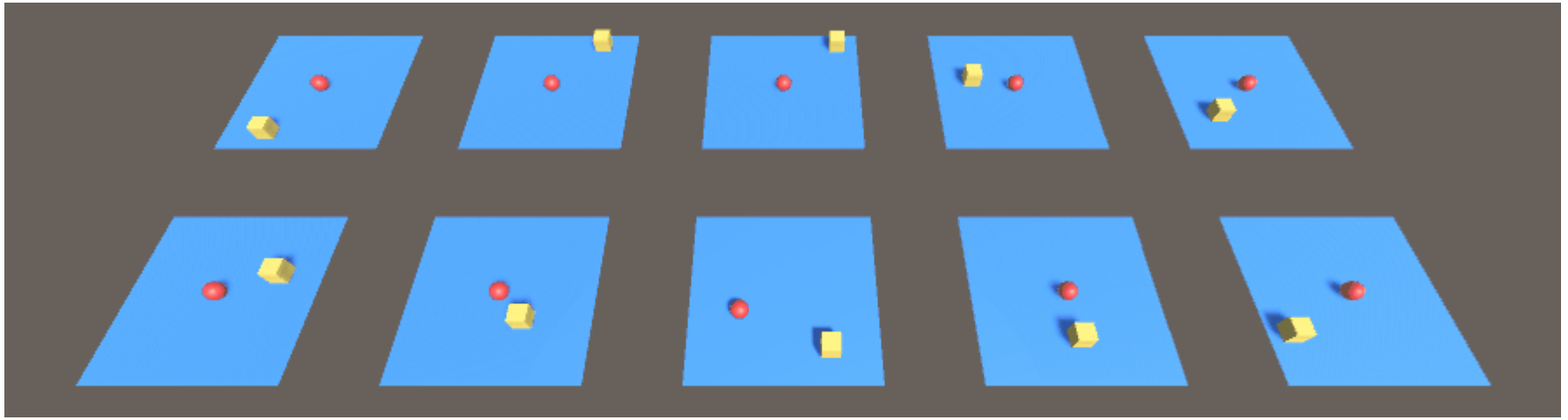
Unity ML-Agents

Training Environment

- 강화학습을 위한 환경으로는
  Mujoco, OpenAI의 Gym, Unity의 ML-Agents 등이 있음.
- 익숙함과 게임에 적용하기 쉬운 이유로 Unity ML-Agents 사용

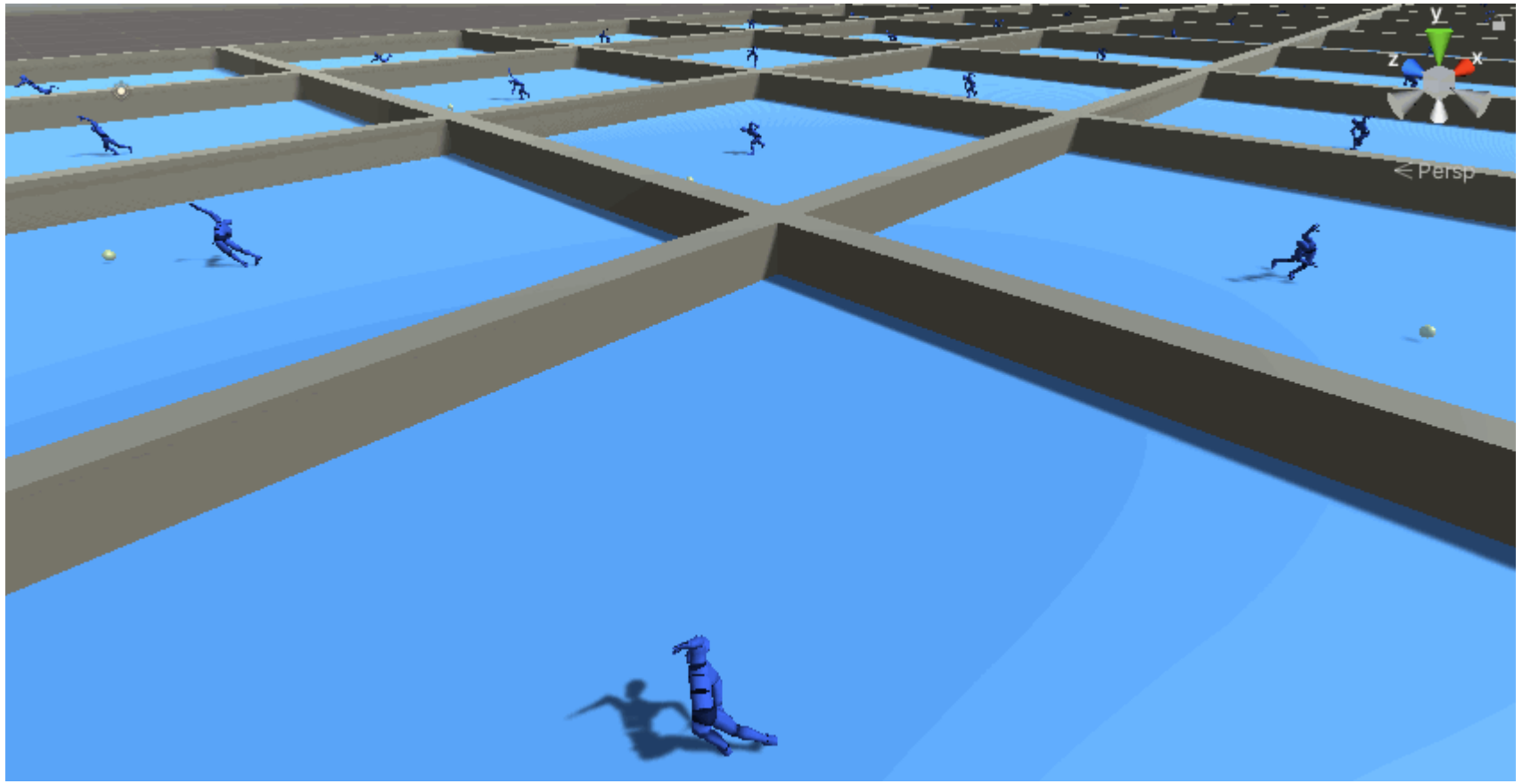






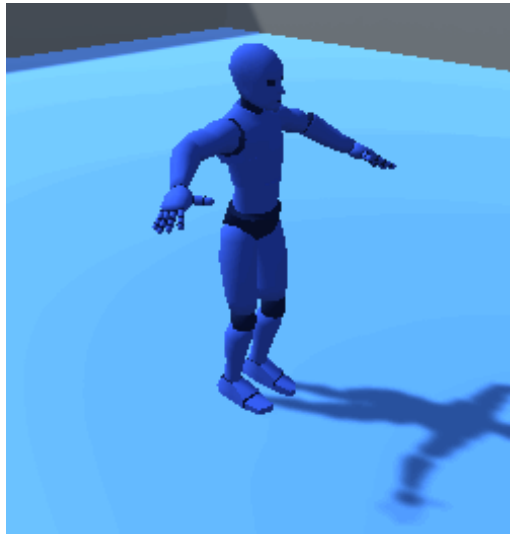
Walk using reinforcement learing

Training walk animation

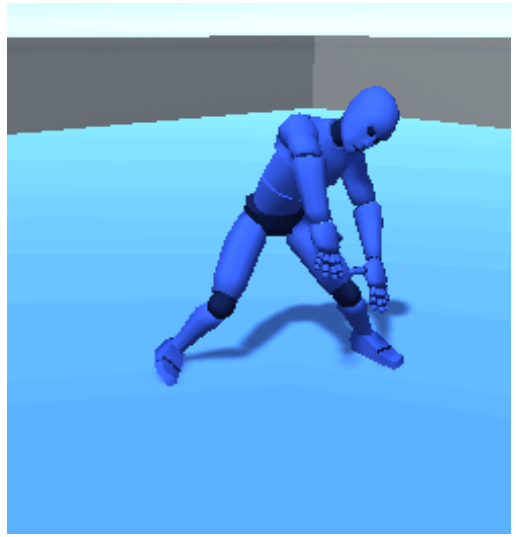


Walk using reinforcement learing

100,000 Step



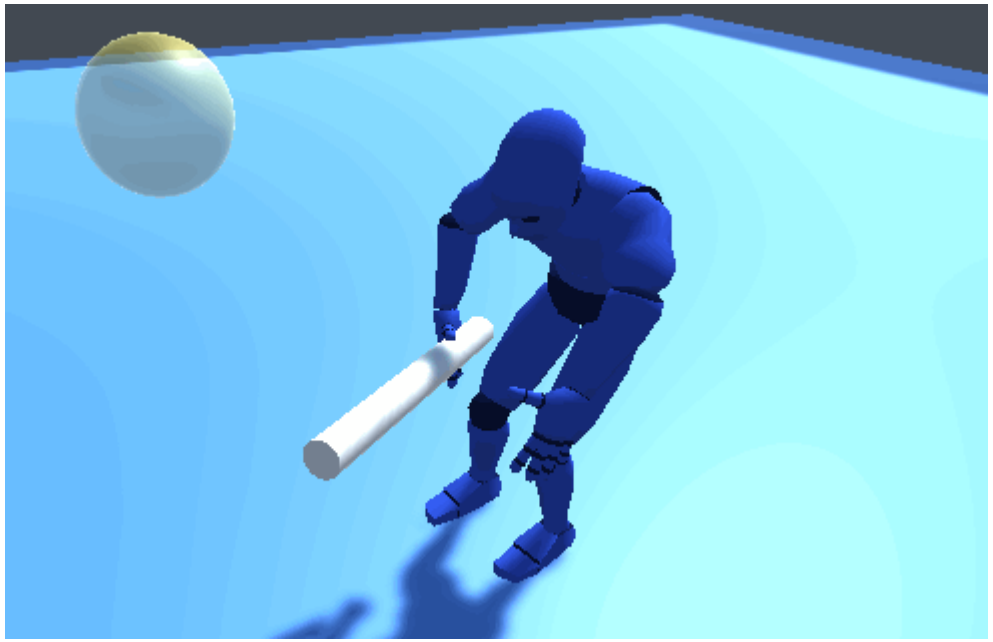
300,000 Step



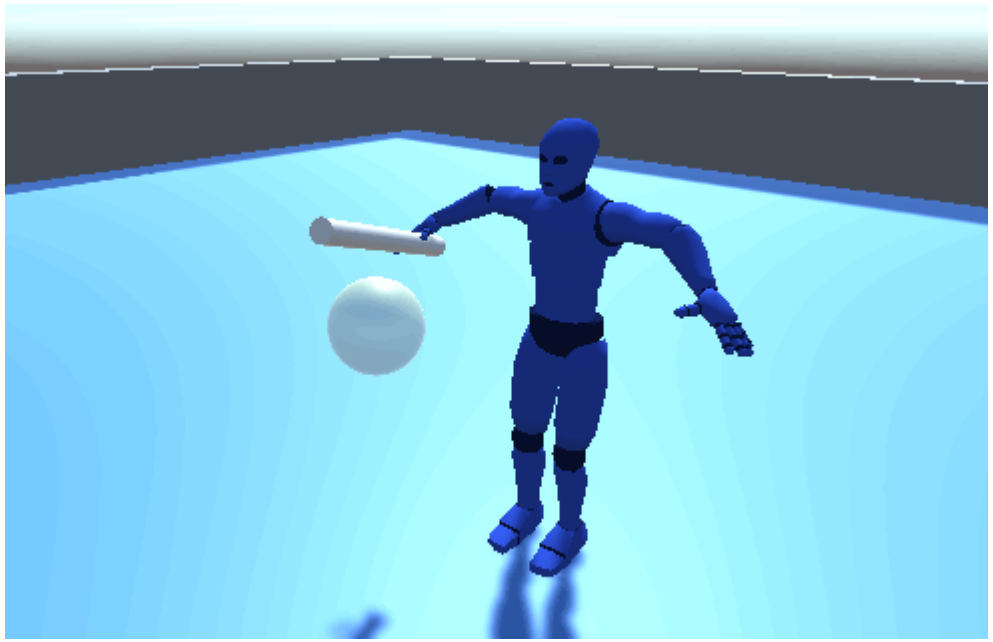
3,000,000 Step

20,000,000 Step

Attack animation



Penalty: -100, Reward: impulse * 1 | 3,000,000 step

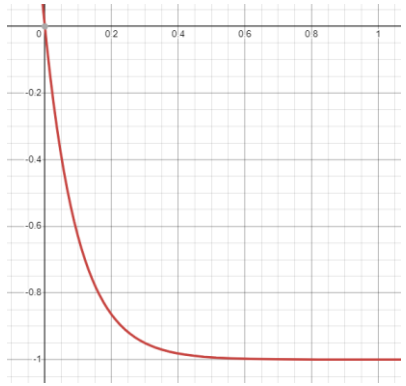


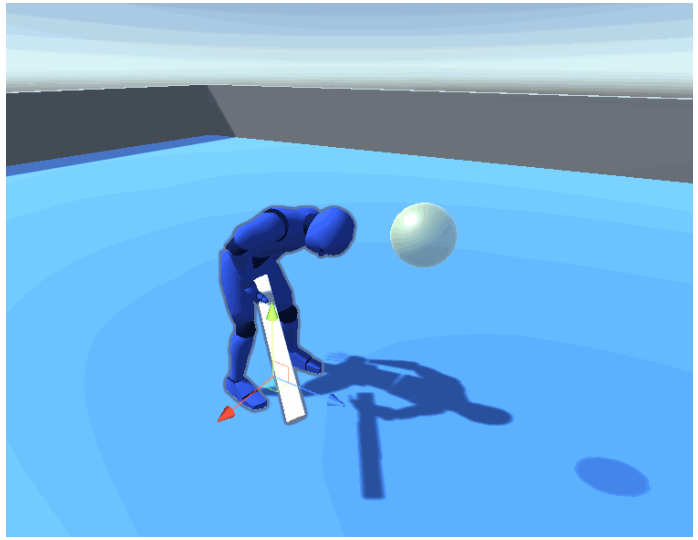
Penalty: -100, Reward: impulse * 100 | 5,000,000 step

Reward function

넘어지지 않기 위해, 무게중심이 안정적인 위치에 있을수록 보상 증가

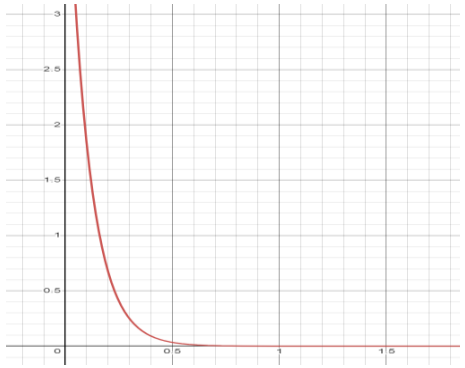
$$R_{stable} = e^{-10\,dist} - 1$$

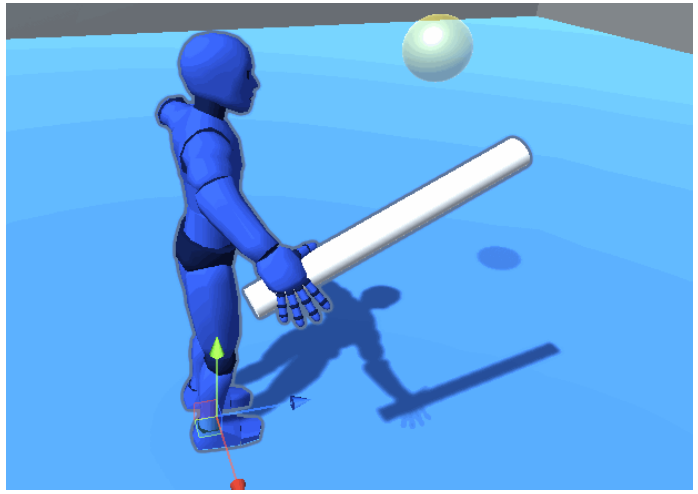




목표를 강하게 타격할수록 보상 증가

$$R_{aim} = e^{-10\,dist(target,weapon)} * weapon\_speed$$

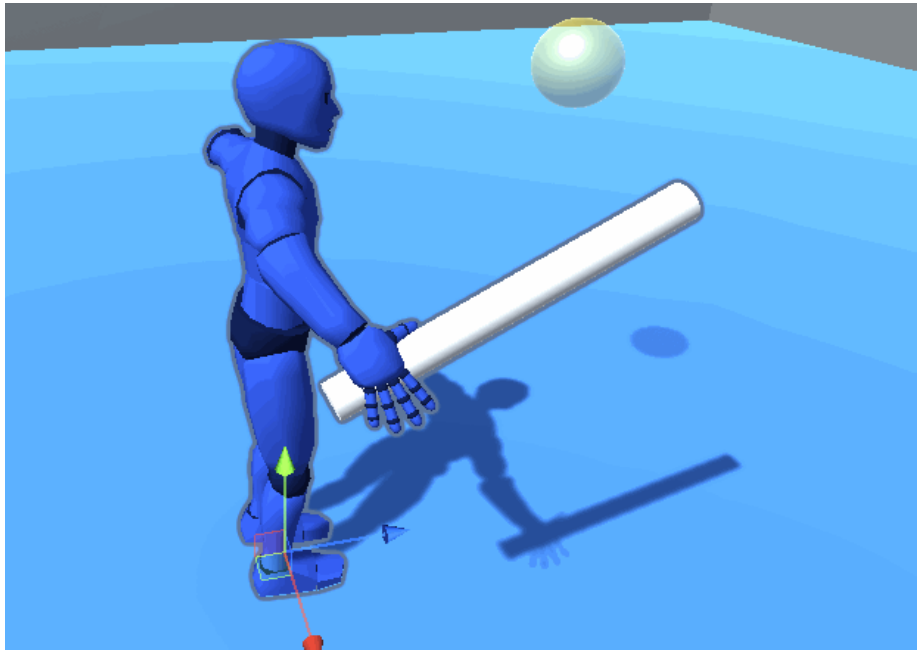




Limitation

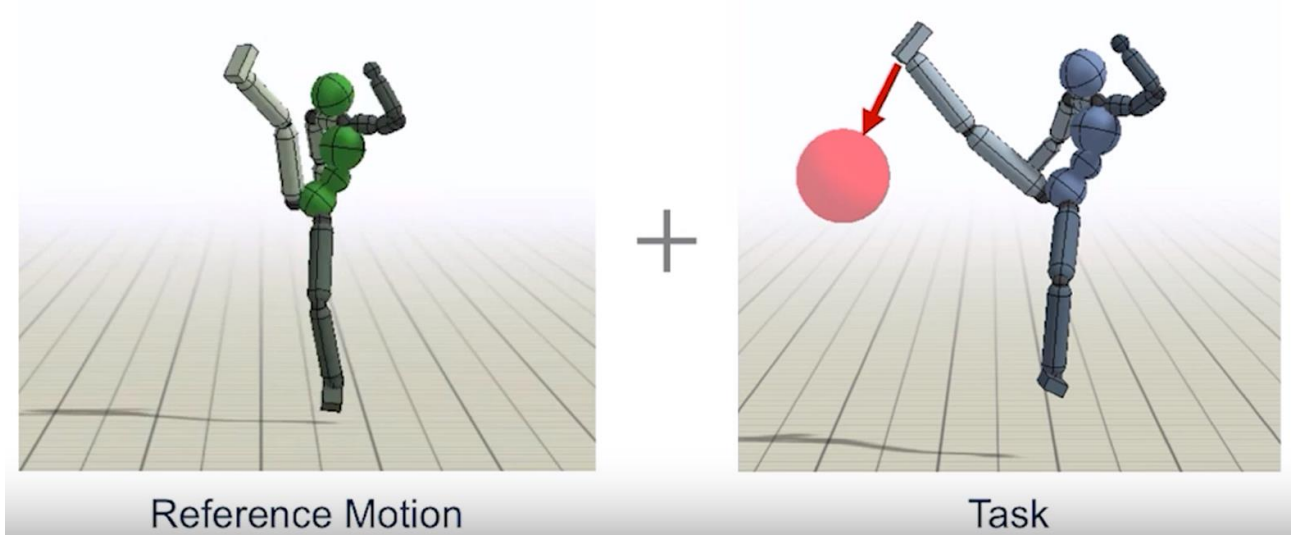
정해진 목표는 잘 수행하도록 학습되지만

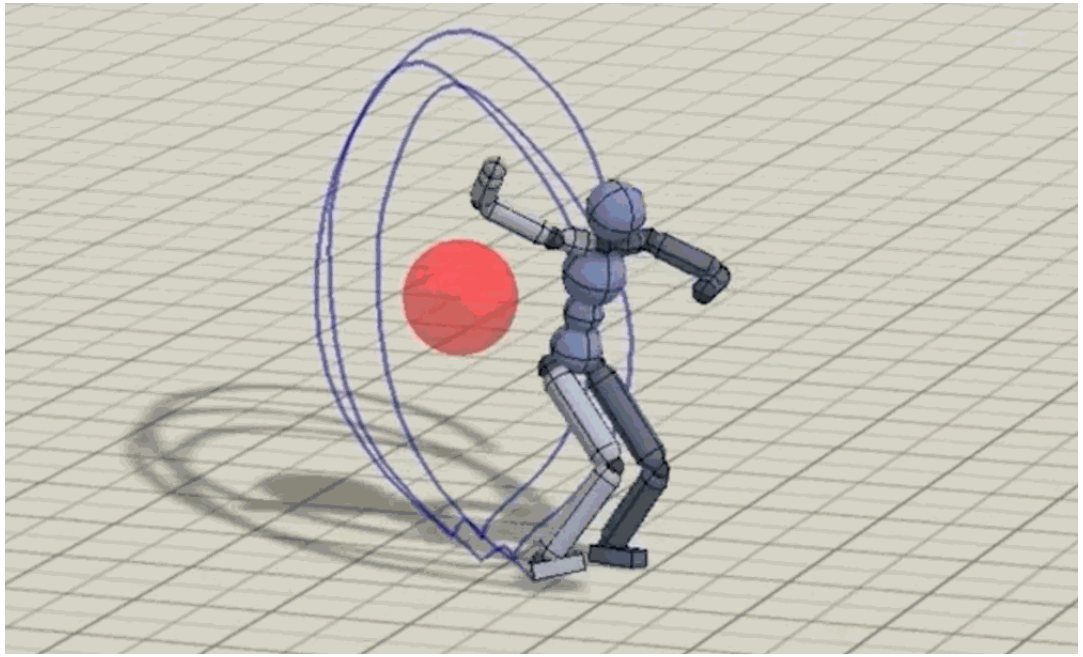
모션이 자연스럽다고 할 수 없음.



Reference

DeepMimic: example-guided deep reinforcement learning of physics-based character skills
(2018, ACM Transactions on Graphics)





- 단순히 강화학습을 사용해 Attack animation을 생성한다면, 학습의 대상인 Agent는 정해진 목표만을 이루기 위해 그 목적에 만 효율적인 기괴한 모션을 생성할 수 있고, 학습이 되기까지 매우 많은 시도를 거쳐야 한다.
- 이 논문에서는, Reference motion을 얼마나 잘 따라하는지도 고려하게 하면서 잘 맞출 수 있게 학습을 시켜, 자연스럽고도 각 상황에 맞는 Physics-based animation을 생성할 수 있게 된다.

States, Action

States

- 캐릭터의 모든 관절의 Orientation, Linear velocity, Angular velocity

Action

- 각 관절이 도달해야 할 목표 각도, 관절이 낼 수 있는 힘

Reward

Reward

- Task reward
  - 맞추려는 캐릭터의 부위와 타겟과의 거리, 한 번 맞추면 애니메이션의 남은 프레임동안 최대 보상으로 고정

$$r_t^G = \begin{cases} 1, & \text{target has been hit} \\ \exp\left[-4||p_t^{tar} - p_t^e||^2\right], & \text{otherwise} \end{cases}$$

- Imitation reward
  - Orientataion matching reward

$$r_t^p = \exp\left[-2\left(\sum_j ||\hat{q}_t^j \ominus q_t^j||^2\right)\right].$$

  - Angular velocity matching reward

$$r_t^v = \exp\left[-0.1\left(\sum_j ||\hat{\dot{q}}_t^j - \dot{q}_t^j||^2\right)\right].$$

  - End-effector matching reward

$$r_t^e = \exp\left[-40\left(\sum_e ||\hat{p}_t^e - p_t^e||^2\right)\right].$$

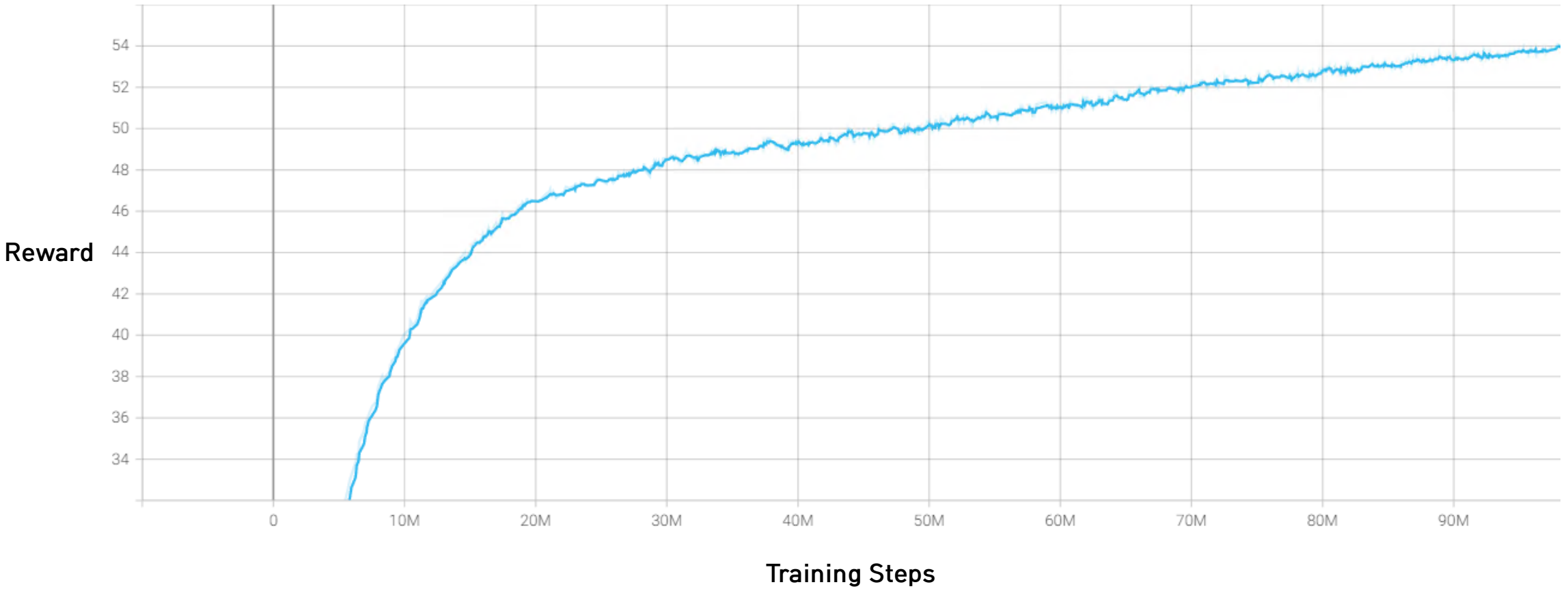
  - Center-of-mass matching reward

$$r_t^c = \exp\left[-10\left(||\hat{p}_t^c - p_t^c||^2\right)\right].$$

Training

하나의 모션을 학습하는데 총 59시간 소요

Training result



Training result

InGame