공학석사학위논문

경험 이식 기법을 통한
제어기 간접 학습 알고리즘

Controller Indirect Learning Algorithm
Using Experimental Implantation Technique

2017년 8월

서울대학교 대학원

컴퓨터공학부

박 종 호

초록

물리 기반 애니매이션이란 가상의 캐릭터들이 물리 법칙의 지배 하에서 움직이도록 하는 것으로, 움직임에 현실성을 부여함으로써 보는 사람들로 하여금 자연스러운 느낌이 들게 해주는 기법이다. 현재 가상 캐릭터의 동작을 생성하기 위해 가장 보편적으로 이용되고 있는 방법은 모션 캡쳐 기법인데, 이 방법은 현실의 사람이나 동물이 배우가 되어 직접 촬영한다는 점에서 필연적으로 몇 가지 물리적 한계를 갖는다. 본 논문은 두 가지 알고리즘을 제안한다. 먼저 첫 번째는 원하는 물리 환경과 가상 캐릭터가 있을 때, 얻고자 하는 동작의 종류에 따라 캐릭터의 움직임에 대한 보상(reward) 시스템만 정해주면 강화학습을 통해 주어진 조건에 맞는 동작을 자동으로 생성할 수 있는 제어기를 학습시키는 방법이다. 두 번째 제안 알고리즘은 첫 번째에 이어지는 내용으로, 주어진 환경에서 잘 학습된 동작 제어기를 갖고 있을 때, 형태 및 구조는 동일하지만 다른 방식으로 환경을 인식하는 가상 캐릭터의 제어기를 빠르게 학습시킴으로써 환경 인식 센서를 일반화하는 방법이다. 실험으로는 장애물을 피해 목표물로 비행하는 가상 캐릭터를 이용하여 이미 학습된 제어기의 경험을 통해 간접적으로 학습된 제어기의 성능을 검증하였다.

주요어: 물리 시뮬레이션, 강화학습, 동작 제어기

학번: 2015-22899

목차

제 1장. 서론

물리 기반 애니매이션은 컴퓨터 애니매이션 및 그래픽스 분야에서 주요한 위치를 차지하고 있으며, 활발하게 연구가 진행되고 있는 분야이다. 물리 기반 애니매이션이란 가상의 캐릭터들이 물리 법칙의 지배 하에서 움직이도록 하는 것으로, 움직임에 현실성을 부여함으로써 보는 사람들로 하여금 자연스러운 느낌이 들게 해주는 기법이다. 실제로 물리 기반 애니매이션 기술은 영화의 특수효과 제작이나 컴퓨터 게임이나 의공학을 비롯한 다양한 분야에서의 시뮬레이션 작업에 적용되어 큰 효용을 가져오고 있다. 하지만 3차원 공간에서 높은 자유도로 제작된 가상 캐릭터의 물리 기반 동작을 생성하는 일은 여전히 어렵고 잘 풀리지 않은 문제로 남아있다. 본 논문에서는 3차원에서 임의의 구조를 가진 고차원 자유도의 가상 캐릭터가 환경에 맞춰 자동으로 움직임을 생성할 수 있도록 학습시키는 강화학습 알고리즘을 제안하고, 나아가 학습된 제어기의 경험 이식을 통해 보다 일반화된 제어기를 학습시키는 방법에 대해 다룬다.

현재 가상 캐릭터의 동작을 생성하기 위해 가장 보편적으로 이용되고 있는 방법은 모션 캡쳐 기법이다. 모션 캡쳐란 사람이나 동물이 배우가 되어 몸에 센서를 부착한 상태에서 원하는 동작대로 움직이고 특수한 카메라로 이를 촬영하여

동작 정보를 얻어내는 기법으로, 이렇게 얻어진 동작 정보를 가상 캐릭터에 입혀 애니매이션을 만들어낸다. 이때 배우들은 실제 물리 환경에서 동작을 하기 때문에 유사한 가상 캐릭터에 동작을 입히면 어느 정도 자연스러운 애니매이션을 제작할 수 있다. 하지만 모션 캡쳐 기법을 이용한 방법은 몇 가지 문제로 인해 한계를 갖는다. 먼저 애니매이션 안의 가상 캐릭터가 움직이는 환경이 모션 캡쳐를 수행하는 환경과 물리적으로 상이하면 적합한 동작을 만들기가 어렵게 된다. 예를 들어 수영을 하는 애니매이션을 만들기 위해서는 모션 캡쳐 배우가 움직이는 환경도 수중과 같이 부력이 작용하는 조건을 갖추어야 자연스러운 동작을 만들 수 있다. 두 번째로 모션 캡쳐는 배우가 직접 촬영하는 방식이기 때문에 제작한 가상 캐릭터의 형태가 모션 캡쳐를 수행하는 배우와 다른 경우 적합한 움직임을 생성하기에 어려움이 있다. 신체 구성 요소의 길이에 차이가 있는 경우에는 후보정을 통해 캐릭터 모델에 적합시킬 수도 있지만 이 방법도 차이가 나는 정도에 따라 한계를 갖는다. 이러한 문제는 용이나 공룡처럼 현실에 존재하지 않는 구조의 캐릭터를 다루는 경우 더욱 큰 문제가 된다. 마지막으로 모션 캡쳐를 이용하면 한 동작을 생성할 때 전문 배우나 기술자들의 많은 노력이 들며 이미 만들어진 동작의 재사용성도 좋지 않다.

본 논문은 두 가지 알고리즘을 제안한다. 먼저 첫 번째는 원하는 물리 환경과 가상 캐릭터가 있을 때, 얻고자 하는 동작의 종류에 따라 캐릭터의 움직임에 대한 보상(reward) 시스템만 정해주면 강화학습을 통해 주어진 조건에 맞는

동작을 자동으로 생성할 수 있는 제어기를 학습시키는 방법이다. 여기서 물리 환경을 선택할 때에는 현실적인 제약을 받지 않기 때문에 중력이 작용하는 일반적인 대기 중으로 설정할 수도 있고, 우주와 같은 무중력 상태, 또는 중력을 상쇄하는 부력이 존재하는 반중력 상태 등 여러 가지 조건으로 자유롭게 설정할 수 있다. 또한 가상 캐릭터도 그 생김새나 구조에 제약을 받지 않는다. 단, 물리 기반 애니매이션을 생성하는 것이 목적이기 때문에 물리적으로 유의미한 동작을 수행하기 힘든 모양으로 설정하는 것은 적절하지 않다. 예컨대 캐릭터가 앞으로 달려나가기를 의도한다면 합리적인 크기의 다리를 만들어주는 등 가속을 할 수 있을법한 부분을 만들어줘야 할 것이다. 보상 시스템이란 캐릭터가 시뮬레이션 상에서 취하는 매 동작마다 얼마나 좋은 동작이었는지에 대해 점수를 부여하는 일종의 함수로 생각할 수 있다. 다시 앞으로 달려나가는 동작을 만드는 상황의 예를 들면, 캐릭터의 현재 속도에서 전방의 성분값에 비례하는 크기의 점수를 부여해줄 수 있다. 본 논문의 실험에서는 실제 공중에서와 유사한 유체 환경으로 설정하였고, 날개를 2개, 4개, 6개 가진 가상의 캐릭터를 모델을 이용해서 장애물을 피하면서 주어진 목표점을 따라가는 동작을 학습시켰다.

두 번째 제안 알고리즘은 첫 번째에 이어지는 내용으로, 주어진 환경에서 잘 학습된 동작 제어기를 갖고 있을 때, 형태 및 구조는 동일하지만 다른 방식으로 환경을 인식하는 가상 캐릭터의 제어기를 빠르게 학습시킴으로써 환경 인식 센서를 일반화하는 방법이다. 가상 캐릭터는 주어진 센서를 이용하여

시뮬레이션 상에서 주변 환경에 대한 정보를 인식하는데, 이러한 인식 방식을 상황에 따라 다르게 부여해줄 수 있다. 본 논문의 실험과 같이 장애물을 피해 목표물을 따라다니는 동작의 경우에는 가상 캐릭터가 장애물에 대한 정보와 목표물에 대한 정보를 인식할 수 있어야 하는데, 이때 인식 방법을 3차원 상에서 캐릭터에 대한 장애물이나 목표물의 상대 좌표를 제공하는 좌표 방식을 이용할 수도 있지만 캐릭터가 바라보는 방향의 시각 정보를 이미지로 제공하는 시각 방식 등 다른 다양한 방식을 이용할 수도 있다. 본 논문의 실험에서는 위에서 언급한 좌표 방식의 제어기가 미리 학습된 상태로 주어졌을 때, 이 제어기에 시각 방식의 센서를 동시에 탑재하고 동작을 진행하면서 시간에 따른 시각 정보와 보상 정보를 수집하였고, 이 정보를 이용하여 초기화 상태의 시각 정보 제어기를 간접적으로 학습시켰고, 그 결과 간접적으로 학습된 시각 방식의 제어기가 직접 학습할 때에 비해 훨씬 적은 시간 동안에 의도한 움직임을 잘 만들어낼 수 있는 상태가 되었고, 이후에는 미리 주어진 좌표 방식의 제어기보다 더 높은 성능을 보이는 것도 확인할 수 있었다.

제 2장. 관련 연구

애니매이션에서 가상 캐릭터의 물리 기반 제어기를 만드는 문제는 컴퓨터 애니매이션 분야에서 굉장히 오래 연구되어왔다. 뿐만 아니라 이는 기계학습, 로봇공학 그리고 생물기계학에서도 많은 관심을 받아온 문제이기도 하다. 특히 최근에는 기계학습 분야에서 딥러닝 기술의 발전에 힘입어 강화학습을 이용해 제어기를 학습시키는 방법이 괄목할만한 성과를 보였다. 따라서 컴퓨터 애니매이션 분야에서의 물리 기반 애니매이션에 대한 연구와 강화학습 분야에서의 제어기 학습 관련 논문을 정리하였다.

2.1 물리 기반 애니매이션

물리 기반 애니매이션의 기법은 몇 가지 주요한 갈래로 나눌 수 있다. 먼저 관절 공간 추적(Joint-space tracking)은 가장 잘 연구되어있는 방법으로 매우 많은 종류의 세부 알고리즘이 개발되어있는 방법이다. 운동학적 목표 궤적(Kinematic target trajectory)으로부터 시간에 따라 주어지는 참조 자세(Reference pose)를 지역적 피드백 제어기(Local feedback controller)를 이용하여 따라간다는 것이 공통이 되는 원리이다. 지역적 피드백 제어기는 로봇공학 분야에서도 가장 보편적으로 이용되는 방법으로 운동학적 목표 궤적과 현재 상태와의 차이를 인식하고 이를 최소화하는 방향으로

관절에 힘을 발생시키는 식으로 작동하며 PD/PID 제어기가 대표적이다 [1, 2, 3]. 목표 궤적으로는 인간이 직접 제작한 동작 데이터를 이용하기도 하고 [4], 모션 캡쳐를 통해 얻은 동작 데이터를 이용하기도 한다 [5, 6].

다음으로 자극-반응 네트워크 제어(Stimulus-response network control) 방법이 있다. 이 방법은 생체 감각-운동 시스템의 작동 방식에 착안하여 고안된 것으로 감각계와 운동계를 뇌신경계가 연결하고 있는 것과 같이 센서 정보(자극)를 받아 행동 정보(반응)로 연결시키는 네트워크를 만들어 이용한다. 이 네트워크는 수많은 파라미터들로 구성되어 있는데, 이 파라미터들을 조정하기 위해 적합 함수(Fitness function)를 설정하고 다양한 최적화 방법을 이용해서 이를 최적화한다. 이때 네트워크는 인공신경망으로 구성할 수도 있고 [7, 8, 9, 10, 11, 12, 13], 유전 프로그래밍(Genetic programming)이나 [14, 15], 또는 순환 패턴 생성기를 이용하기도 한다 [16, 17, 18, 19].

마지막으로 제약적 역학 최적화 제어(Constrained dynamics optimization control) 방법이 있다. 이는 최적 제어 이론(Optimal control theory)와 최적화 이론(Optimization theory)를 기반으로 하여 발달한 방법으로 의도하는 동작을 만들어내기 위해 취해야하는 행동(Action)에 대해 온라인(On-line)으로 최적화를 수행한다. 이때 캐릭터나 환경의 역학적 속성은 최적화

문제에서의 제약 조건이, 그리고 의도하는 캐릭터의 동작은 목표 함수가 되어 최적화 문제로 옮겨지게 된다. 이 방법을 이용할 때 가장 어려움이 되는 부분은 최적화가 실시간으로 진행되어야 하며 캐릭터의 동작이 끝나기 전까지는 궤적 전체에 대한 참조가 불가능하다는 점이다. 이를 해결하기 위해서는 최적화 문제의 목적 함수의 설계에서 현시점 이전의 정보만을 이용하는 접근 방법이 있고 [20] 실시간이 아닌 오프라인(Off-line)으로 최적화를 수행한 뒤에 그 결과로 실시간 최적화를 보조하는 방법이 있다 [21, 22].

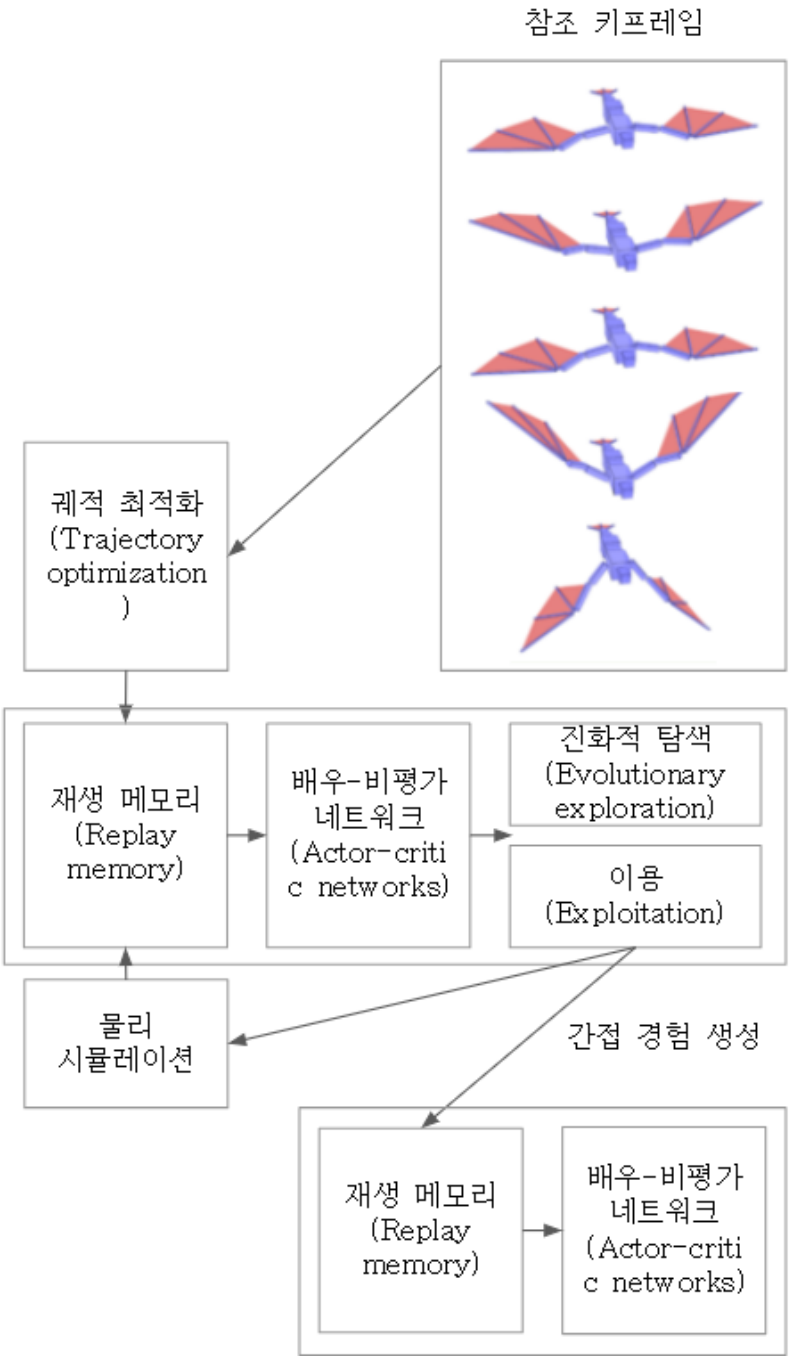
2.2 강화학습을 이용한 제어기 학습

컴퓨터 비전 분야에서 큰 발전을 가져온 딥러닝은 이후 강화학습 알고리즘과 결합하여 제어기를 학습하는 문제에서도 뛰어난 성능을 보여주었다. 지도학습(Supervised learning)에서와 같이 인공신경망을 구성하고 이를 학습시키는 방법에 따라 다양한 알고리즘이 존재한다. 직접적 정책 근사(Direct policy approximation) 방법은 현재 상태(State)를 입력받아 취해야 할 행동(Action)을 출력값으로 내보내는 정책(Policy)을 신경망을 이용하여 직접적으로 근사한다. 궤적 최적화(Trajectory optimization)를 이용해 얻은 궤적을 학습 데이터로 삼아 지도학습을 하는 방식이 대표적인데 이러한 방법은 캐릭터가 제어 중에 학습 데이터에 없는 상태로 빠지는 경우 학습된 바가 없기 때문에 부적절한 행동을 취하게 된다는 문제점이 있다 [23, 24]. 딥 큐 학습(Deep Q Learning)은 컴퓨터

게임을 하는 데에 있어 사람의 실력을 뛰어넘는 제어기를 학습하는 결과를 보여주어 제어기 학습의 수준을 한 단계 높이는데 기여하였으며 [25], 이어서 탐색한 경험 정보를 보다 효율적으로 이용하는 방법 [26], 그리고 딥 큐 학습의 과추산(Overestimation) 문제를 보완하는 방법도 등장하였다 [27]. 이후 연속적인 공간에서도 제어기를 학습할 수 있는 CACLA(Continuous actor-critic learning automation) [28]를 이용하여 몇 가지 형태의 가상의 동물 캐릭터가 2차원 환경에서 지형에 맞는 동작을 자동으로 발생시켜 앞으로 달려가도록 하는 연구도 진행되었다 [29]. 정책 기울기(Policy gradient)를 추정하여 기울기 하강(Gradient descent) 방법을 수행하는 정책 기울기 방법의 고안은 연속적인 행동을 취하는 문제에서 캐릭터 제어기 학습에 새로운 알고리즘군으로 등장하였고 현재까지 최고 성능의 알고리즘들은 모두 이 알고리즘군에 속한다 [10, 12, 13].

제 3장. 알고리즘 개요

시스템의 전체적인 개요는 [그림 1]과 같다. 입력으로는 의도하는 동작의 형태를 대략적인 키프레임으로 제작하여 넣어준다. 시스템은 제공된 키프레임을 기반으로 궤적 최적화를 수행하여 물리적으로 발생이 가능하면서 앞으로 최대한 빠르게 날아가는 동작을 생성하고 이를 훈련시키고자 하는 제어기의 재생 메모리에 입력해준다. 이때 재생 메모리는 축적된 데이터가 없는 초기화 상태로, 이렇게 최적화된 궤적 데이터가 입력된 상태로 학습을 시작함으로써 임의 탐색으로는 찾아내는 데에 많은 시간이 걸리는 동작을 빠르게 학습할 수 있게 된다. 이어서 기존 강화학습 알고리즘에서 일반적인 임의 탐색(Random exploration)에 CMA-ES(Covariance matrix adaptation evolution strategy)최적화 기법을 이용한 탐색을 추가한 진화적 탐색(Evolutionary exploration)을 이용한 CACLA를 수행하여 직진성 동작에서 보다 일반화된 방향의 비행을 학습하도록 한다. 이에 알고리즘을 Evo-CACLA로 칭한다.


참조 키프레임
궤적 최적화
(Trajectory
optimization
)
재생 메모리
(Replay
memory)
배우-비평가
네트워크
(Actor-criti
c networks)
진화적 탐색
(Evolutionary
exploration)
이용
(Exploitation)
물리
시뮬레이션
간접 경험 생성
재생 메모리
(Replay
memory)
배우-비평가
네트워크
(Actor-criti
c networks)

[그림1] 학습 시스템 개요.

제어기 학습이 충분히 수렴하면 다른 종류의 센서를 탑재한 제어기를 간접적으로 학습시킨다. 학습된 제어기에서 이용(Exploitation)만을 수행하여 궤적을 생성하면 주어진 목적을 최대한 잘 수행하는 동작을 생성한다. 예를 들어 본 논문의 실험에서는 물리 환경에 장애물과 목표물을 임의로 생성하고 캐릭터가 목표물을 추적하며 따라다니되 장애물과의 충돌을 피하는 것이 우선시되도록 목적을 설정하였는데, 잘 학습된 제어기는 장애물의 배치가 극단적이지 않은 일반적인 상황에서 대부분의 경우 목적을 결함 없이 잘 수행하는 상태에 이르렀다. 이종 센서를 탑재한 새로운 제어기는 기존에 학습된 제어기와 환경을 인식하는 방법이 다르기 때문에 얻어진 궤적 데이터를 간접적인 학습에 바로 이용할 수는 없다. 때문에 궤적을 생성하면서 동시에 이종 센서가 인식하는 방식의 데이터를 저장해두어야 한다. 이를 위해서 학습된 제어기는 궤적을 생성할 때에 자신의 센서와 함께 이종 센서를 이중으로 탑재한다. 자신의 센서는 정책(Policy)의 입력으로 넣어 적절한 행동(Action)을 얻어내어 궤적을 만들어나가는 데에 이용하고, 동시에 이종 센서로부터 얻은 데이터와 보상 데이터를 저장한다. 이렇게 저장된 경험 데이터는 간접적으로 얻어진 것이기 때문에 간접 경험(Indirect experience)이라고 할 수 있으며 이종 센서 제어기의 학습에 이용할 재생 메모리에는 초기화 상태에서 간접 경험 데이터를 채워넣어준다. 이종 센서 제어기는 추가적인 탐색이 없이 재생 메모리에 있는 간접 경험으로만 학습을 시켜도 충분한 성능으로 동작을 생성할 수 있게 된다.

이렇게 간접 경험을 통해 학습된 제어기는 상황에 따라 기존의 제어기를 뛰어넘는 성능을 보이기도 한다.

제 4장. 초기 최적화 궤적 생성

본 논문의 학습 시스템은 이용자가 제공한 키프레임 애니매이션으로부터 시작된다고 할 수 있다. 이용자가 제공하는 키프레임 애니매이션은 최종적으로 생성하고자 하는 애니매이션 동작의 대략적인 모습을 안내해주는 역할을 한다. 그렇기 때문에 키프레임 애니매이션은 물리 법칙에 대한 고려가 되지 않은 운동학적(Kinematic) 동작으로, 이를 그대로 모방하여 움직이면 제어기가 조종하는 가상 캐릭터는 이용자의 기대와는 달리 균형을 잡지 못하고 추락하는 등의 실패적인 결과를 보일 것이다. CMA-ES 최적화 기법을 이용한 궤적 최적화는 이 키프레임 애니매이션을 본래의 물리적으로 가능한 동작이면서 본래의 목적을 잘 수행하는 동작으로 변경해주는 역할을 한다. 이러한 메카니즘을 따름으로써 본 알고리즘은 이용자가 운동학적으로 원하는 애니매이션의 모습을 어느 정도 반영을 하면서도 물리적으로 가능한 물리 기반 애니매이션을 생성할 수 있다.

이용자가 제공한 키프레임 애니매이션은 $\{q_1, \ldots, q_n\}$으로 나타낼 수 있다. 여기서 q는 한 프레임에 캐릭터가 취하고 있는 자세(Pose)로 캐릭터의 운동 자유도 N_{dof} 만큼의 관절 각(Joint angle) 값으로 구성된다. 즉, 키프레임 애니매이션은 $n \times N_{dof}$차원의 벡터가 되어 CMA-ES의

최적화 대상이 된다. CMA-ES에서 해에 대한 평가 함수는 다음과 같다.

$$r = r_{target} + r_{collision} + r_{effort} + r_{balance} + r_{regularization}$$

여기서 각 항들은 다시 아래와 같이 표현된다.

$$r_{target} = -w_{target} \times \left|p\right|^2$$

$$r_{collision} = w_{collision} \times d^2 \ (d > 0) \ or \ 0 \ (d \le 0)$$

$$r_{effort} = -w_{effort} \times |\tau|^2$$

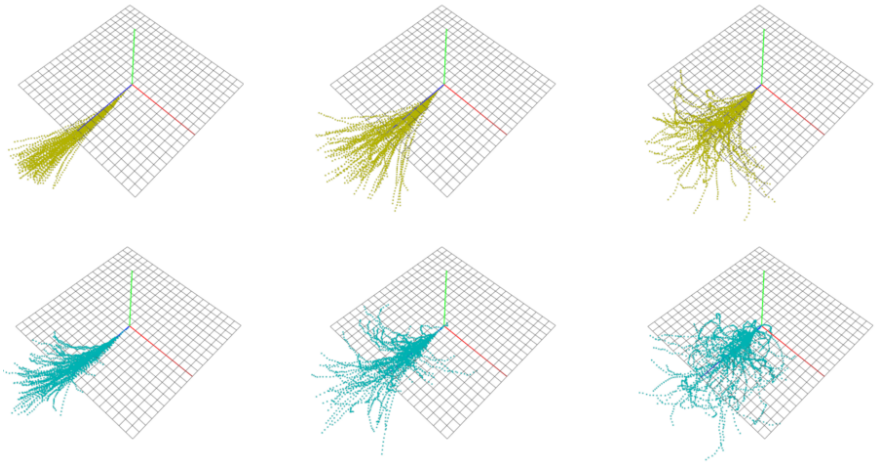
$$r_{balance} = -w_{balance1} \times |\omega|^2 - w_{balance2} \times |1 - v \cdot u|^2$$

$$r_{regularization} = -w_{regularization} \times \sum_i \left|a_i - q_i\right|^2$$

r_{target}은 캐릭터가 목표물로부터 얼마나 멀리 떨어져있는지를 나타내는 항으로 상대 위치 벡터의 크기 제곱 $\left|p\right|^2$에 비례하는 음수의 값으로 설정하여 거리가 가까워지는 방향으로 유도하는 역할을 한다. $r_{collision}$은 충돌에 대한 항으로 장애물과 충돌이 발생하면 투과한 깊이 d가 0 이상의 값을 갖게 되어 장애물에 접촉하는 것을 억제하는 역할을 한다. r_{effort}는 각 관절에 걸리는 돌림힘 τ의 제곱에 비례하는 음수 값으로 되어 각 관절에 과도한 돌림힘을 발생시키지 않는 쪽으로 유도한다. 다음으로 $r_{balance}$는 균형을 잡도록 하기 위한 항으로 ω는 캐릭터의 가장 중심이 되는 구성체의 각속도이고, v와 u는 각각 캐릭터의 윗 방향 벡터와 물리 환경의 윗 방향 벡터를 나타내어 캐릭터가

심하게 흔들리거나 기울어지는 것을 방지한다. 마지막으로 $r_{regularization}$은 캐릭터의 자세 a와 사용자가 제공한 키프레임 애니매이션에서의 자세 q의 차이에 패널티를 적용하여 주어진 키프레임 애니매이션과 지나치게 달라지지 않도록 유도하는 역할을 한다.

위와 같은 함수를 이용하여 해를 평가할 때에 주의해야할 것이 바로 임의의 흔들림(Random perturbation)에 대한 강건함을 반영해야 한다는 점이다. 만일 단순한 접근으로 평가하고자 하는 해를 따라 궤적을 생성하면서 위에서 설정한 함수 값을 통해 해에 대한 평가를 실시한다면, 최적화 결과로 특정 해가 도출되겠지만 이렇게 얻어진 해는 노이즈가 조금만 섞여도 쉽게 망가지는 동작일 가능성이 있다. 하지만 학습 시스템은 궤적 최적화로 얻은 동작을 기반으로 하여 그 근처로 조금씩 임의의 탐색을 수행하면서 추가적인 학습을 수행하여야 한다. 이때 기반이 되는 동작이 노이즈에 취약한 동작이라면 임의 탐색을 수행할 때 행동 공간(Action space) 상에서 기준 행동 근처의 공간을 탐색하여 향상된 제어기를 얻어내기가 힘들어질 것이다. 때문에 본 알고리즘에서는 CMA-ES 해 평가를 할 때에도 임의로 노이즈를 섞어서 궤적을 생성하며 이렇게 여러 번 반복하였을 때의 평균값을 해에 대한 점수로 책정하는 방법을 이용하여 노이즈가 발생하는 상황에서 성능 변화가 크게 없는 안정적인 동작을 얻을 수 있도록 하였다 [그림 2].



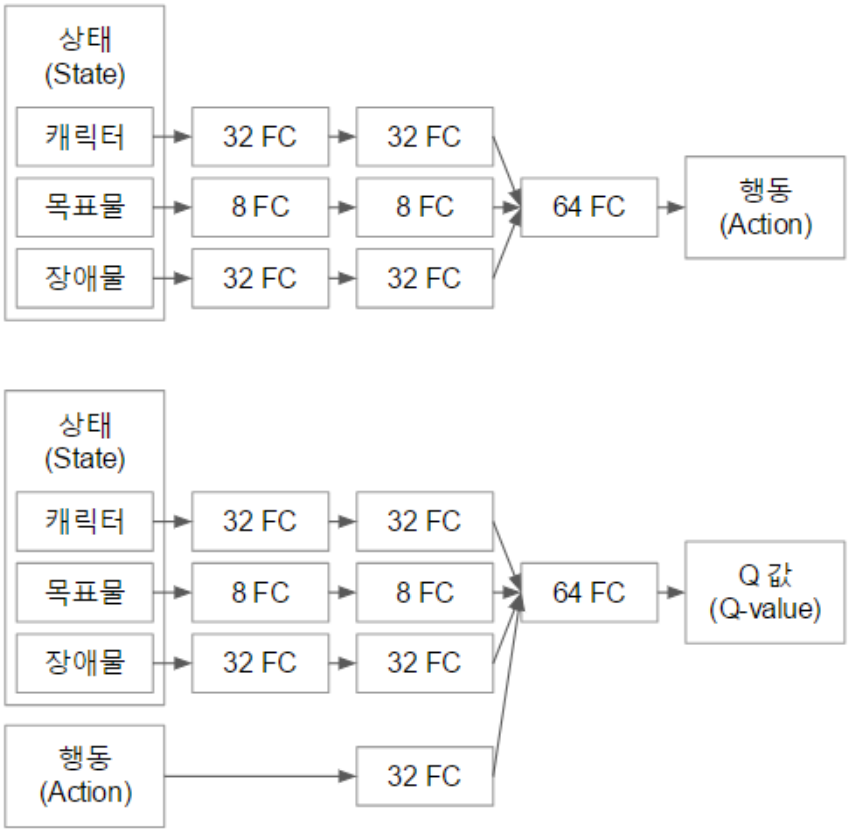
[그림 2] 평균값 평가를 적용하여 궤적 최적화를 수행한 경우(위)와 적용하지 않은 경우(아래) 노이즈에 대한 궤적 변경 범위. 왼쪽부터 오른쪽으로 가면서 노이즈의 강도가 높아질 때, 평균값 평가를 적용한 경우 비교적 궤적의 변경 폭이 작다.

제 5장. 진화적 CACLA

궤적 최적화를 통해 얻어진 정보를 통해 제어기의 초기 학습을 수행한다. 이를 충분히 마치고 나면 캐릭터는 어느 정도 외부로부터 임의 흔들림이 발생하더라도 중심을 잃지 않으면서 앞으로 나아가는 정도의 성능을 갖추게 된다. 하지만 이 단계에서는 항상 전방으로만 나아가고 목표물의 위치를 전방이 아닌 다른 곳으로 변화시키거나 경로에 장애물을 배치하더라도 그에 반응해서 움직이지는 않는다. 초기 학습 단계에 이어 수행하는 진화적 CACLA 알고리즘은 전방 비행만 하는 상태의 제어기를 보다 일반적인 상황에 대해 학습시켜서 장애물을 피해면서 변화하는 목표물의 위치를 따라 움직일 수 있도록 만들어주는 역할을 한다.

학습은 배우-비평가(Actor-critic) 학습 방법으로 진행된다. 배우와 비평가 네트워크는 인공신경망으로 구성되어있고 각각의 구조는 [그림 3]과 같다. 상태(State)란 캐릭터가 행동에 대한 판단을 내리는 데에 이용하는 모든 정보를 나타내는 것으로 현재 캐릭터의 자세, 목표물의 위치 그리고 장애물의 위치 등의 정보가 모두 포함되어 있다. 캐릭터의 자세에 대한 정보는 캐릭터의 동작을 만들어내는 자유도 만큼의 관절 각 값들로 구성되어 있고, 목표물에 대한 정보는 캐릭터의 지역 좌표계에서 본 목표물의 상대 위치로 주어진다. 마지막으로 장애물에 대한 인식은 두 가지

방법으로 나누었는데, 목표물에 대한 정보와 같이 상대 좌표를 이용하는 단순한 방법과 캐릭터의 전방 시야를 이미지 그대로 인식하는 방법이다. 전자는 상대적으로 네트워크를 구성하는 파라미터의 수가 적고 학습도 빠르게 되는 편인 데에 비해 후자는 내부적으로 이미지에서 반응해야할 장애물의 상대적 위치를 유추해야하는 상황인만큼 학습이 보다 까다로운 문제가 된다. 행동(Action)이란 다음 시뮬레이션 시각(Simulation time)에 캐릭터가 취할 자세가 된다. 행동으로 캐릭터의 관절에서 발생시킬 돌림힘 값을 이용하는 방법이 더 직관적으로 다가올 수 있는데, 일반적으로 캐릭터가 물리 기반 애니매이션 동작을 취하며 움직일 때, 관절 각에 비해 돌림힘의 크기는 급격하게 변하며 연속성이 떨어지는 경향을 갖기 때문에 학습을 통해 근사하기가 더 까다롭다는 문제가 있다. 실제로 돌림힘을 행동으로 설정했을 때에는 제어기 학습이 원활하게 되지 않는 것으로 보아 이는 학습에 결정적인 영향을 주는 것으로 보인다. 이에 본 논문에서는 관절 각의 집합인 자세를 행동으로 설정하였다. 내부적으로는 제어기가 특정 자세를 행동으로 생성하면 하층부에 내장된 PD 제어기를 통해 그 자세를 취하기 위한 돌림힘 값이 계산되어 캐릭터의 관절에 적용시키는 방식으로 작동한다. 배우-비평가 네트워크의 업데이트 과정은 [31]를 따른다.



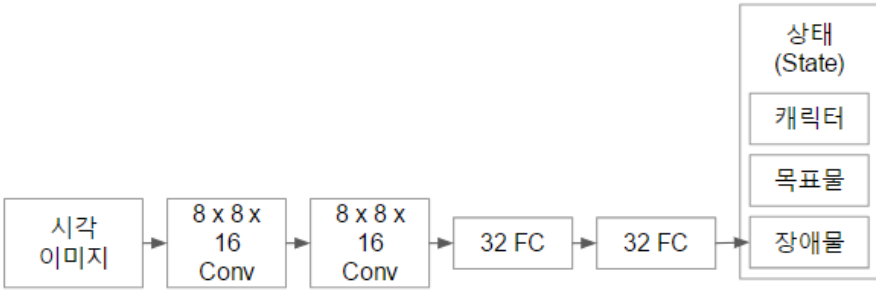
[그림 3] 배우 비평가 네트워크의 구조.

제 6장. 간접 경험 학습

진화적 CACLA를 통해 충분히 학습된 제어기는 임의의 상태(State)에서 주어진 일반적인 형태의 목적을 잘 수행한다. 본 논문의 실험에서는 학습시킬 제어기의 종류로 두 가지를 설계하였다. 두 제어기의 차이점은 장애물을 인식하는 센서의 방식이 다르다는 것으로 먼저 좌표 방식의 센서를 탑재한 제어기는 캐릭터의 지역 좌표계를 기준으로 한 장애물의 좌표를 직접 알아낼 수 있도록 설계되었다. 두 번째로는 시각 센서 방식의 제어기이다. 시각 센서는 같이 캐릭터가 향하고 있는 방향을 중심으로 시야각 270도 반경의 물체들을 원근 카메라(Perspective camera)로 촬영한 이미지를 가공되지 않은 상태 그대로 이용하는 방식이다. 카메라의 한계 거리 이내에 아무 것도 없을 때를 픽셀값 0으로 하고, 장애물이 카메라에 가까울 수록 255의 값을 가지며 따라서 모든 픽셀은 0 이상 255 이하의 정수값을 갖는다. 이미지는 가로 64, 세로 64 픽셀의 정밀도로 구성되어있으며 실제 촬영된 모습은 [그림 4]과 같다. 전자의 제어기의 경우 배우 네트워크가 약 10K의 파라미터로 구성되어 있는 데에 비해 후자의 제어기의 경우 50K로 약 5배 정도 많은 파라미터로 구성되어 있다. 따라서 직접 학습을 하는 데에 훨씬 더 많은 시간이 소요되며 실제로 실험에서는 학습 과정에서 안정적으로 수렴하지 못하는 결과를 확인할 수 있었다. 전자의 경우 네트워크의 구성은

[그림 3]에서 제시되었고 후자의 경우 네트워크의 구성은 [그림 5]과 같다.

[그림 4] 시각 센서에서 촬영한 이미지. 가로 64, 세로 64 픽셀로 구성되어있으며 각 픽셀은 0~255 구간의 정수 값을 갖는다.



[그림 5] 시각 센서를 이용하는 제어기의 네트워크 구조.

본 논문에서는 이에 대한 대안으로 좌표 방식의 제어기를 먼저 학습시킨 뒤에 이미 학습된 상태의 좌표 방식 제어기의 동작 경험을 이용하여 시각 방식의 제어기를 간접적으로 학습시키는 방법을 고안하였고 이렇게 학습시킨 결과 장애물을 회피하며 목표물을 향해 비행하는 문제에서 간접 학습된 시각 방식의 제어기가 기존의 좌표 방식 제어기를 뛰어넘는 성능을 보이는 것을 확인하기도 하였다. 간접

학습을 하기 위해서는 먼저 이미 학습되어있는 기존의 제어기에 간접적으로 학습시키고자 하는 제어기의 센서를 중첩해서 탑재한다. 그리고 두 가지 종류의 센서가 모두 탑재된 제어기는 임의의 탐색(Exploration)이 없이 이용(Exploitation)만을 하여 궤적을 생성하도록 한다. 이때 충분히 다양한 상황에서의 데이터가 수집되어야 하기 때문에 장애물과 목표물 모두 완전한 임의의 상태(State)에서 시작하여 데이터가 일반성을 확보할 수 있도록 하는 것이 중요하다. 실험에서는 약 500K의 시뮬레이션 시각에 대한 데이터를 발생시켰다. 궤적을 생성하는 동안 탑재되어있는 두 가지 센서의 역할은 각각 다른데, 먼저 기존의 센서는 학습된 제어기가 갖고 있는 정책(Policy)에서 다음 행동(Action)을 계산하기 위해 입력값으로 넣어줄 상태(State)를 얻는 데에 이용된다. 그리고 추가로 탑재된 센서의 데이터는 궤적 생성 중에는 이용되지 않으며, 센서에서 인식한 환경에 대한 정보와 그 시뮬레이션 시각에 발생시킨 행동 및 보상에 대한 정보와 함께 저장을 한다. 이는 이후에 학습시킬 제어기의 입장에서 직접 경험한 것이 아니라 학습된 다른 제어기가 발생시킨 경험이기 때문에 간접 경험 데이터라고 말한다.

간접적으로 학습시킬 제어기의 학습 메커니즘은 이전 제어기를 학습시킬 때와 같이 재생 메모리를 초기화하는 것으로부터 시작된다. 단, 기존의 학습 방법에서는 이용자가 제공한 키프레임 애니매이션을 CMA-ES로 최적화한 궤적 데이터를 이용해 초기화를 한 것과 달리, 간접 학습 방법에서는 저장해둔 간접 경험 데이터를 이용하여 재생

메모리를 초기화한다. 이후 간접 학습되는 제어기는 학습 도중에 임의 탐색 및 진화적 탐색을 수행하지 않는다. 다시 말해 학습을 수행하는 중에 재생 메모리에 새로 저장되는 데이터가 없이 주입되었던 간접 경험 데이터만을 이용해서 배우와 비평가 네트워크를 업데이트 하는 것이다. 이때 업데이트 방식 자체는 기존의 제어기를 학습할 때와 같이 [31]의 알고리즘을 따른다.

제 7장. 실험 및 결과

본 실험에서 캐릭터에게 주어진 목적은 목표물 추적이다. 단, 목표물을 추적하는 과정에서 임의로 장애물을 도입할 수 있으며 날아가는 경로에 장애물이 있으면 목표물을 향한 최단 시간 경로에서 우회하여 회피하는 경로로 추적을 계속해야 한다. 이는 보상 함수를 구성하는 항에서 아래의 두 항에 곱해지는 상수를 조정함으로써 구현하였다. 먼저 목표물과의 거리를 좁히도록 유도하는 항은 아래와 같다.

$$r_{target} = -w_{target} \times |p|^2$$

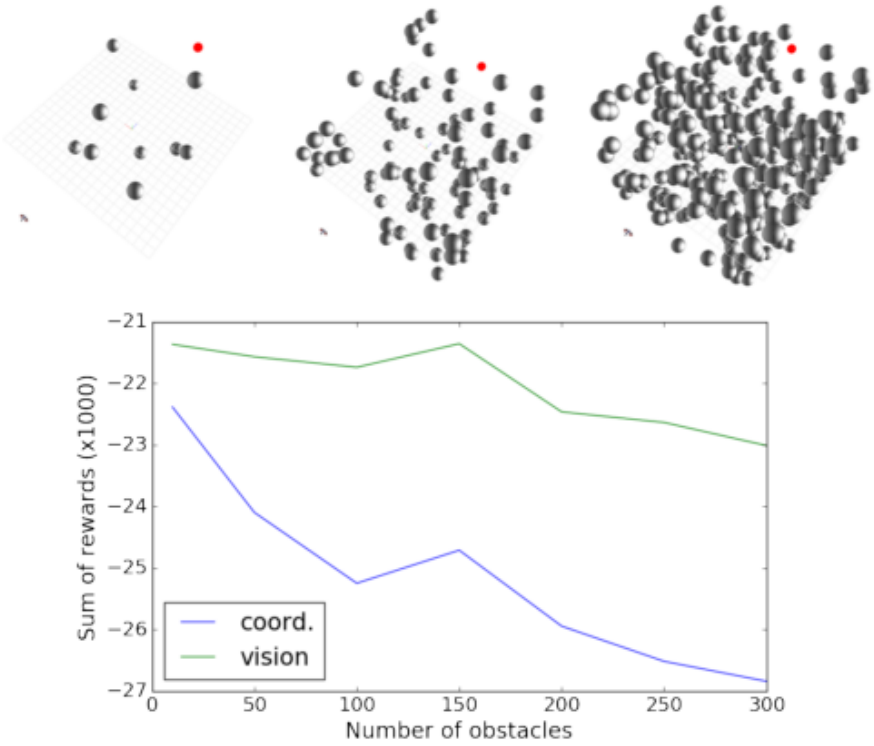
그리고 장애물과의 충돌을 피하도록 하는 항은 아래와 같다.

$$r_{collision} = w_{collision} \times d^2 \ (d > 0) \ or \ 0 \ (d \leq 0)$$

이 두 항에서의 계수 w_{target}과 $w_{collision}$의 균형을 맞추는 것이 장애물을 얼마나 잘 피해가는지를 결정하는 결정적인 요소로 작용한다. 본 실험에서는 $w_{collision} / w_{target} > 1.0e05$ 정도 범위에서 성공적인 결과를 보였으나 이는 캐릭터의 구조나 물리 환경으 특징 등 기타 요소들의 영향을 받기 때문에 초기 학습 과정에 대한 관찰을 통해 조정해줄 필요가 있다.

간접 경험을 통해 시각 센서 제어기를 학습하는 실험에서는 데이터 분포의 일반성 및 임의성을 충분히 확보하기 위하여

500K 이상의 간접 경험 데이터를 발생시켜서 매번 임의 추출로 배우 및 비평가 네트워크 업데이트를 수행했다. 이때 한 번에 32개 데이터로 구성된 배치 데이터에 대하여 10K 이상 학습이 진행되면 기존의 제어기와 비슷한 수준의 제어기가 학습되는 것을 확인하였다. 이후 충분히 수렴할 정도로 학습한 뒤 제어기의 성능을 비교 시험해보기 위해 장애물의 수를 증가시켜가며 목표물을 찾아가는 실험을 수행하였다. [그림 6]를 보면 “coord”로 표기된 좌표 센서 방식의 제어기와 “vision”으로 표기된 시각 센서 방식의 제어기의 성능 차이를 확인할 수 있다. 세로축은 보상 합을 나타내며 두 제어기 모두에 대해 같은 초기 상태를 주고 주어진 상태에서 목표물로 충분히 근접하기까지의 평균적인 보상 합을 비교하였다. 두 제어기 모두 장애물의 개수가 많아질 수록 충돌이 잦아지고 많이 우회해서 목표물로 이동하기 때문에 평균적인 보상 합이 낮아지는 것을 볼 수 있는데, 장애물이 많아질 수록 좌표 방식의 제어기의 성능이 시각 방식의 제어기에 비해 크게 감소하는 것을 알 수 있다.



[그림 6] 시각 센서와 좌표 센서를 이용하는 제어기의 성능 비교. 가로 축은 장애물의 개수, 세로 축은 평균적인 보상의 총합을 나타낸다. 시각 센서를 이용하는 제어기는 좌표 센서를 이용하는 제어기의 경험을 통해 간접적으로 학습되었음에도 장애물의 개수가 많아짐에 따라 보상 총합의 감소가 좌표 센서 제어기에 비해 적다.

참고문헌

[1] Tomei, Patrizio. "A simple PD controller for robots with elastic joints." IEEE Transactions on automatic control 36.10 (1991): 1208-1213.

[2] Tomei, Patrizio. "Adaptive PD controller for robot manipulators." IEEE Transactions on Robotics and Automation 7.4 (1991): 565-570.

[3] Rivera, Daniel E., Manfred Morari, and Sigurd Skogestad. "Internal model control: PID controller design." Industrial & engineering chemistry process design and development 25.1 (1986): 252-265.

[4] Hodgins, Jessica K., and Wayne L. Wooten. "Animating human athletes." Robotics Research. Springer London, 1998. 356-367.

[5] Yin, KangKang, Kevin Loken, and Michiel van de Panne. "Simbicon: Simple biped locomotion control." ACM Transactions on Graphics (TOG). Vol. 26. No. 3. ACM, 2007.

[6] Zordan, Victor Brian, and Jessica K. Hodgins. "Motion capture-driven simulations that hit and react." Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation. ACM, 2002.

[7] Lewis, F. W., Suresh Jagannathan, and A. Yesildirak. Neural network control of robot manipulators and non-linear systems. CRC Press, 1998.

[8] Psaltis, Demetri, Athanasios Sideris, and Alan A. Yamamura. "A multilayered neural network controller." IEEE control systems magazine 8.2 (1988): 17-21.

[9] Levine, Sergey, et al. "End-to-end training of deep visuomotor policies." arXiv preprint arXiv:1504.00702 (2015).

[10] Schulman, John, et al. "High-dimensional continuous control using generalized advantage estimation." arXiv preprint arXiv:1506.02438 (2015).

[11] Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." International Conference on Machine Learning. 2016.

[12] Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971 (2015).

[13] Schulman, John, et al. "Trust Region Policy Optimization." ICML. 2015.

[14] Wolff, Krister, and Peter Nordin. "Learning biped locomotion from first principles on a simulated humanoid robot using linear genetic programming." Genetic and

Evolutionary Computation Conference. Springer Berlin Heidelberg, 2003.

[15] Lewis, M. Anthony, Andrew H. Fagg, and Alan Solidum. "Genetic programming approach to the construction of a neural network for control of a walking robot." Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on. IEEE, 1992.

[16] Taga, Gentaro. "A model of the neuro-musculo-skeletal system for human locomotion." Biological cybernetics 73.2 (1995): 97-111.

[17] Taga, Gentaro, Yoko Yamaguchi, and Hiroshi Shimizu. "Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment." Biological cybernetics 65.3 (1991): 147-159.

[18] Miyakoshi, Seiichi, et al. "Three dimensional bipedal stepping motion using neural oscillators-towards humanoid motion in the real world." Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on. Vol. 1. IEEE, 1998.

[19] Taga, Gentaro. "A model of the neuro-musculo-skeletal system for anticipatory adjustment of human locomotion during obstacle avoidance." Biological cybernetics 78.1 (1998): 9-17.

[20] Macchietto, Adriano, Victor Zordan, and Christian R. Shelton. "Momentum control for balance." ACM Transactions on graphics (TOG) 28.3 (2009): 80.

[21] Muico, Uldarico, et al. "Contact-aware nonlinear control of dynamic characters." ACM Transactions on Graphics (TOG). Vol. 28. No. 3. ACM, 2009.

[22] Muico, Uldarico, Jovan Popović, and Zoran Popović. "Composite control of physically simulated characters." ACM Transactions on Graphics (TOG) 30.3 (2011): 16.

[23] Levine, Sergey, and Pieter Abbeel. "Learning neural network policies with guided policy search under unknown dynamics." Advances in Neural Information Processing Systems. 2014.

[24] Levine, Sergey, and Vladlen Koltun. "Learning Complex Neural Network Policies with Trajectory Optimization." ICML. 2014.

[25] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." arXiv preprint arXiv:1312.5602 (2013).

[26] Schaul, Tom, et al. "Prioritized experience replay." arXiv preprint arXiv:1511.05952 (2015).

[27] Van Hasselt, Hado, Arthur Guez, and David Silver. "Deep Reinforcement Learning with Double Q-Learning." AAAI. 2016.

[28] Van Hasselt, Hado, and Marco A. Wiering. "Reinforcement learning in continuous action spaces." Approximate Dynamic Programming and Reinforcement Learning, 2007. ADPRL 2007. IEEE International Symposium on. IEEE, 2007.

[29] Peng, Xue Bin, Glen Berseth, and Michiel van de Panne. "Terrain-adaptive locomotion skills using deep reinforcement learning." ACM Transactions on Graphics (TOG) 35.4 (2016): 81.

[30] Hansen, Nikolaus, and Andreas Ostermeier. "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation." Evolutionary Computation, 1996., Proceedings of IEEE International Conference on. IEEE, 1996.

Abstract

Controller Indirect Learning Algorithm Using Experimental Implantation Technique

Jongho Park
Computer Science and Engineering
The Graduate School
Seoul National University

Physics-based animation is a technique that allows virtual characters to move under the rule of physics, which makes people feel natural by giving reality to motion. Currently, the most commonly used method for generating the motion of a virtual character is the motion capture technique. However, this method necessarily has some physical limitations in that a human being or an animal is photographed directly as an actor. This paper proposes two algorithms. First, if there is a desired physical environment and a virtual character, and if only the reward system for the movement of the character is determined according to

the kind of the motion to be obtained, the motion corresponding to the given condition can be automatically generated through the reinforcement learning This is a method of learning the controller. The second proposal algorithm follows the first one. When we have a well-learned motion controller in a given environment, we quickly learn the controller of a virtual character that has the same type and structure but recognizes the environment in a different way. It is a generalization method. In the experiment, the performance of the controller learned indirectly through the experience of the already learned controller was verified by using the virtual character flying from the obstacle to the target.