제목
인간 동작 데이터로 애니메이션되는 아바타의 학습
Training Avatars Animated with Human Motion Data

성명
이강훈, 이제희
Kang Hoon Lee, Jehee Lee

소속기관
서울대학교 컴퓨터공학부 운동연구실
Movement Research Lab., School of Computer Science, Seoul National University

연구 세부분야
3차원 컴퓨터 그래픽스, 애니메이션, 가상현실

주소: 서울시 관악구 신림동 산 56-1 서울대학교 302동 312-1호
우편번호: 151-744
전화번호: 02-880-1864
팩스번호: 02-871-4912
E-mail 주소: zoi@mrl.snu.ac.kr

인간 동작 데이터로 애니메이션되는 아바타의 학습
Training Avatars Animated with Human Motion Data

국문 요약
제어 가능하고 상황에 따라 반응하는 아바타의 제작은 컴퓨터 게임 및 가상현실 분야에서 중요한 연구 주제이다. 최근에는 아바타 애니메이션과 제어의 사실성을 높이기 위해 대규모 동작 캡처 데이터가 활용되고 있다. 방대한 양의 동작 데이터는 넓은 범위의 자연스러운 인간 동작을 수용할 수 있다는 장점을 갖는다. 하지만 동작 데이터가 많아지면 적절한 동작을 찾는데 필요한 계산량이 크게 증가하여 대화형 아바타 제어에 있어 병목으로 작용한다. 이 논문에서 우리는 레이블링(labeling)이 되어있지 않은 모션 데이터로부터 아바타의 행동을 학습시키는 새로운 방법을 제안한다. 이 방법을 사용하면 최소의 실시간 비용으로 아바타를 애니메이션하고 제어하는 것이 가능하다. 본 논문에서 제시하는 알고리즘은 Q-러닝이라는 기계 학습 기법에 기초하여 아바타가 동적인 환경과의 상호작용에 따른 시행착오를 통해 주어진 상황에 어떻게 반응할지 학습하도록 한다. 이 접근 방식의 유효성은 아바타가 서로 간에, 그리고 사용자에 대해 상호작용하는 예를 보임으로써 증명한다.

국문 키워드
휴먼 애니메이션, 모션 캡쳐, 강화 학습, 가상 환경, 대화형 제어

영문 요약
Creating controllable, responsive avatars is an important problem in computer games and virtual environments. Recently, large collections of motion capture data have been exploited for increased realism in avatar animation and control. Large motion sets have the advantage of accommodating a broad variety of natural human motion. However, when a motion set is large, the time required to identify an appropriate sequence of motions is the bottleneck for achieving interactive avatar control. In this paper, we present a novel method for training avatar behaviors from unlabelled motion data in order to animate and control avatars at minimal runtime cost. Based on machine learning technique, called Q-learning, our training method allows the avatar to learn how to act in any given situation through trial-and-error interactions with a dynamic environment. We demonstrate the effectiveness of our approach through examples that include avatars interacting with each other and with the user.

영문 키워드
human animation, motion capture, reinforcement learning, virtual environments, interactive control

1. 서론

컴퓨터 게임과 가상현실 분야에서 3차원 아바타의 실시간 애니메이션 및 제어는 중요한 문제이다. 최근에는 방대한 양의 동작 데이터를 대화형 응용 프로그램이나 오프라인 애니메이션에 활용하여 사실감을 높이는 연구가 활발히 진행되고 있다. 사실적인 아바타의 동작을 만들기 위해서는 몇 가지 요건이 충족되어야 한다. 우선 자연스러운 아바타의 동작을 대규모로 추출해야 하고, 둘째로는 그 동작들이 부드럽게 연결될 수 있도록 저장해야 하며, 끝으로 사용자가 원하는 방식으로 아바타의 움직임을 제어할 수 있어야 한다. 본 논문은 이 중 마지막 요건, 즉 아바타가 사용자의 명령에 적절하게 반응하여 필요한 동작들을 선택할 수 있게 하는 문제를 다룬다.

아바타의 동작 데이터는 동작 간의 연결 관계를 표현하는 유향 그래프(directed graph)로 표현하곤 한다. 예를 들어, 점프 동작 뒤에는 쪼그리고 앉는 동작이 뒤따라올 수 있다는 등의 연결 관계가 그래프에 저장된다. 이 그래프와 마르코프 프로세스(Markov process)나 히든 마르코프 모델(hidden Markov model) 같은 통계적 모델이 결합하면 아바타의 동작에 융통성과 제어 능력을 부여할 수 있다[1,2,3,4,5,6,7,8,9,10]. 하지만 충분한 양의 사실적인 동작을 저장하기 위해서는 그래프의 크기가 매우 커지고, 따라서 적절한 동작을 찾는데 걸리는 시간도 크게 증가하기 때문에 이를 대화형 아바타 제어에 활용하는 것은 쉽지 않다.

본 논문에서는 많은 양의 동작 데이터로부터 적절한 동작을 최소한의 비용으로 찾도록 하는 선행계산(pre-computation) 방법을 제안한다. 기본적인 접근방법은 아바타가 특정한 상태(state)에 놓여있을 때 취할 수 있는 각각의 행동(action)에 대한 효용성(utility)을 미리 계산하고 테이블로 저장하여 최적에 가까운 일련의 행동을 테이블 검색을 통해 효율적으로 찾을 수 있게 하는 것이다. 이 방법은 Q-러닝(Q-learning)이라는 기계 학습(machine learning) 알고리즘에 기초한다. Q-러닝을 이용하면 주어진 상태-행동 모델에 대한 제어 정책(control policy)을 학습시킬 수 있다. 이 논문에서는 아바타의 동작을 제어하는데 사용할 몇 가지 정책을 학습시킨다.

시작 상태와 목표 상태가 주어졌을 때 중간 경로를 찾는 경로 계획(path planning) 알고리즘이나 상태 공간 탐색(state-space search) 알고리즘은 많이 알려져 있다. 본 논문에서 제안하는 접근 방식은 이들 알고리즘과 큰 차이가 있다. 이 논문에서는 하나의 상태에서 다른 상태로 연결되는 경로를 찾는 대신, 아바타가 주어진 상황에서 어떻게 행동해야 하는지를 결정하는 제어 정책을 찾고자 한다. 아바타는 동적 환경에서 시행착오를 통해 이러한 정책을 학습해나간다. 일단 제어 정책을 학습하고 나면, 아바타는 목표 상태에 도달하기 위해 지나야 하는 일련의 상태를 실시간에 매우 효율적으로 찾아낼 수 있다. 이 논문의 방식은 주어진 상태 공간의 범위와 해상도에서 최적의 제어 정책을 찾아낼 수 있다는 점에서 해상도 완결성(resolution-complete)을 갖는다.

본 논문은 둘 이상의 아바타가 상호작용하는 예와 동적 환경에서 사용자가 아바타를 제어하는 예를 보임으로써 논문에서 제안한 방법의 실시간 효용성을 보여주고자 한다. 실험을 위해 가능한 물리적 상호작용이 자주 발생하는 행동 영역을 선택했고, 사용자는 마우스 등의 인터페이스를 이용해서 아바타의 동작을 직접적이고 즉각적으로 제어할 수 있다. 본 논문에서 제안하는 방식은 비교적 간단하게 구현할 수 있으며, 폭넓은 행동 영역에 적용 가능하다.

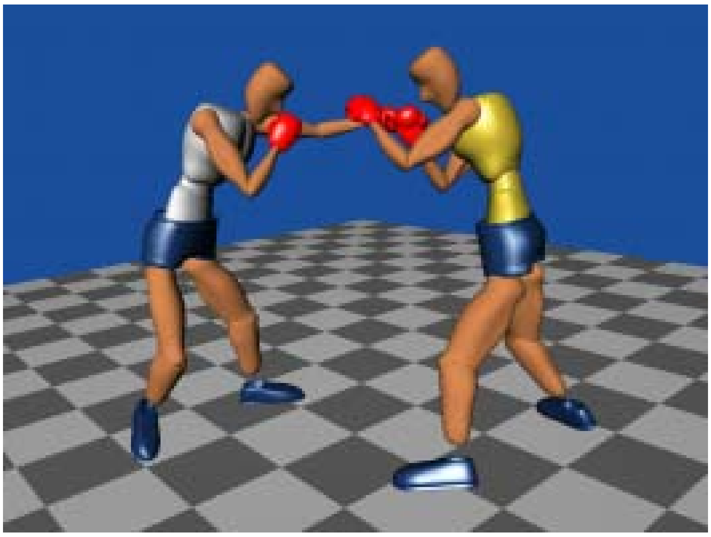

그림 1: 실시간 권투 시뮬레이션. 동작 데이터는 한 명의 권투 선수가 섀도우 복싱하는 모습을 동작 캡처하여 얻었다. 섀도우 복싱이란 훈련의 일환으로서 가상의 적과 스파링하는 것을 말한다. 권투 선수 아바타는 동작 데이터를 통해 상대, 그리고 사용자와 상호작용하기 위한 몇 가지 동작을 학습했다.

2. 배경

3차원 캐릭터의 애니메이션 및 제어는 수십 년간 컴퓨터 그래픽스 분야에서 활발히 연구되어 왔다[11,12,13,14,15]. 최근에는 특히 동작 캡쳐 데이터를 사용하는 데이터 기반 방식에 관심이 쏠리고 있다. 동작 캡쳐 시스템을 이용하면 연기자의 움직임을 실시간에 그대로 아바타에 옮김으로써 매우 쉽게 대화형 아바타 제어가 가능하다. 많은 연구자들이 연기자의 움직임을 크기와 비례가 다른 아바타에 적용하는 인형극(puppetry) 기법을 연구했다[16,17,18,19,20]. 하지만 이 논문은 동작 캡쳐 시스템 같은 별도의 사용자 인터페이스 장비를 갖추지 않고 아바타를 제어하는 방법에 초점을 맞추고 있다. 이 연구에서도 동작 캡쳐 장비가 이용되나 처음 데이터를 수집하는 단계에서만 필요하고 이후의 인터페이스는 마우스와 같은 일반적인 장비를 이용하도록 한다.

방대한 양의 동작 데이터에 내재한 구조를 파악하기 위한 방법으로 통계적 모델이 자주 사용되어왔다. 몇몇 연구자들은 동작 데이터에 통계적 변화를 도입하는 방법을 연구하기도 했다[21,22]. 보다 널리 사용되는 방식은, 먼저 데이터를 단순화하기 위해 PCA를 이용해서 차원(dimension)을 낮추고, 비슷한 동작들끼리 클러스터(cluster)로 묶은 후, 마르코프 프로세스 모델을 이용해서 클러스터 간의 전이(transition)를 가능케 하는 방식이다. 클러스터 간의 전이 관계는 유향 그래프로 나타낼 수 있는데 일반적으로 전이 테이블 형태로 표현한다. 브랜드(Brand)와 헤르쯔만(Herzmann)은 동작 데이터를 일반화하고 동작 스타일에 변화를 주기 위해서 변형된 히든 마르코프 모델을 사용했다[2]. 보덴(Bowden)은 동작 데이터를 압축시켜 표현하기 위해 구간별 비선형 PCA와 마르코프 체인을 함께 사용했다[23]. 몰리나-탄코(Molina-Tanco)와 힐튼(Hilton)은 사용자가 지정한 키프레임을 보간하는 동작을 찾기 위해 PCA 및 클러스터 기반 시스템을 개발하였다[10]. 리(Li) 등은 클러스터 내부의 동작을 보다 잘

근사하기 위해 선형 다이내믹 시스템을 채용했고, 이 시스템의 변수를 변화시킴으로써 동작에 작은 변화들을 줄 수 있게 했다[7]. 또한 김태훈 등이 사용한 통계적 모델은 k-평균 클러스터링(k-mean clustering)과 전이 그래프를 활용하여 음악에 맞춰 춤 동작을 생성하였다[5].

몇몇 연구 그룹은 기존 동작 데이터를 여러 조각으로 잘라낸 후 새로운 순서로 조합하여 새로운 동작을 합성하는 기법을 연구해왔다. 이 방식은 동작의 클러스터 수준에서 전이 그래프를 만드는 대신 프레임 수준에서 전이를 허용하는 것이 특징이다. 클러스터 수준의 동작 그래프를 만들 때는 PCA 차원 감소와 클러스터링 과정에서 동작의 세부 데이터가 손실되는데 반해 프레임 수준의 동작 그래프에서는 이런 현상을 피할 수 있다. 하지만 전이 그래프의 크기가 커지기 때문에 이를 이용해서 아바타를 제어하는 작업은 보다 어려워진다. 풀런(Pullen)과 브리글러(Bregler)는 동작 데이터를 잘게 자른 후에 사용자가 지정한 키프레임에 맞게 재배치하는 방식을 제시했다[8]. 코바(Kovar) 등은 동작 데이터로부터 그래프 구조를 생성한 뒤, 사용자가 그린 경로를 따라 아바타가 이동하도록 전역적 탐색 알고리즘을 도입했다[4]. 아리칸(Arikan)과 포사이스(Forsyth)는 계층적 그래프를 만들어서 동작이 특정한 시공간적 제약을 만족시키도록 하는 무작위 탐색(randomized search) 알고리즘을 사용했다[1]. 이 두 알고리즘은 모두 전체 공간에서 해를 찾기 때문에 대화형 제어에는 적합하지 않다. 이제희 등은 온라인 지역 탐색(local on-line search) 알고리즘을 적용했는데, 탐색 범위를 어느 정도 제한할 경우 초당 5~8프레임 정도의 애니메이션을 생성할 수 있었다[6]. 하지만 이 정도 성능조차 여러 캐릭터를 동시에 실시간으로 애니메이션하기에는 부족하다. 같은 논문에서 저자는 그래프가 특정한 환경을 상정하도록 하고 아바타 동작의 융통성에 제한을 가하면 그래프 탐색이 매우 효과적일 수 있음을 지적했다. 최민규 등은 특정한 환경(대개 비어있는)에서 캡쳐한 동작 데이터로부터 복잡한 환경에 적용 가능한 동작 그래프를 생성했다[24]. 환경에 특화된 그래프를 이용하면 장애물이 놓인 복잡한 환경을 가로질러 이동하는 동작을 쉽게 찾을 수 있다. 아리칸 등은 시공간적 제약 조건이 명시될 경우 그래프로부터의 동작 합성 과정을 동적 프로그래밍(dynamic programming)으로 해결할 수 있음을 보였다[25]. 본 논문은 이러한 바탕 위에 보다 효율적인 알고리즘을 제시한다. 즉, 가능한 모든 상황에 대한 그래프 탐색을 미리 계산한 후 그 결과를 테이블에 저장해둠으로써 최소한의 실시간 비용만으로 적절한 동작 경로를 찾도록 하는 것이다. 제안한 시스템은 앞에서 언급한 그래프 검색 알고리즘이나 동적 프로그래밍 알고리즘에 비해 수백 배에서 수천 배까지 빠른 실시간 성능을 보인다.

기계 학습 기법은 컴퓨터 애니메이션 분야에서 폭넓게 적용돼왔다. 노(Ngo)와 마크(Mark)는 유전 프로그래밍을 사용하여 단순한 생명체의 동적 제어기(dynamic controller)를 만들었다[26]. 또한 심즈(Sims)는 가상 생명체가 스스로 형태를 진화시켜나가는 시뮬레이션을 위해 유전 프로그래밍을 활용하기도 했다[27]. 그레쥬크(Grzeszczuk) 등은 물고기나 달 착륙기와 같은 다이내믹 모델의 애니메이션을 위해 자동 학습 기법을 연구했다[28,29]. 이들 기법은 컨트롤러 추상화와 신경망에 기반하고 있다. 팔룻소스(Faloutsos) 등은 다이내믹 컨트롤러가 적절히 동작할 수 있는 "선행 조건"을 식별하기 위해서 서포트 벡터 머신(SVM)을 이용했다[30]. 아리칸 등은 동작 데이터베이스에 직관적인 키워드를 지정하기 위해 SVM을 사용했고, 사용자가 이들 키워드를 이용하여 원하는 시나리오를 기술할 수 있도록 했다[25].

강화 학습(reinforcement learning)은 동적 환경과의 상호작용에 따른 시행착오를 통해 특정

상황에 대한 대응 방식을 학습하는 문제를 말한다. [31,32]에서 강화 학습에 대한 소개를 찾아 볼 수 있다. 강화 학습은 로봇 제어와 경로 계획분야에서 다양하게 변화된 형태로 사용되어왔다. 앳커슨(Atkeson) 등은 두 팔을 가진 로봇이 양쪽 손의 막대기로 곤봉을 번갈아 때림으로써 저글링(juggling)하도록 학습시켰다[33]. 마타릭(Mataric)은 강화 학습을 이용해서 일군의 바퀴 달린 로봇들이 집게를 이용해 작은 원반을 모아서 정해진 위치까지 옮기도록 학습시켰다[34]. 컴퓨터 그래픽 분야에서는 강화 학습이 그다지 많이 사용되지는 않았다. 블룸버그(Blumberg) 등은 음성 명령에 반응하도록 훈련된 가상의 강아지 캐릭터를 만들었다[35]. 그는 청각 신호와 행동이 시간적으로 가깝게 연결될 때 이 둘을 하나의 쌍으로 묶도록 훈련시켰는데, 이는 생태학적 관점에서 보면 적절한 방법이라고 할 수 있다. 본 연구에서는 블룸버그의 접근 방식과 달리 아바타가 훨씬 폭넓은 상태 공간과 다양한 행동을 바탕으로 보다 깊은 탐색을 수행하도록 훈련시키고자 한다.

본 연구는 전역 조명(global illumination)과 동적 형상 변형 (dynamic deformation) 분야에서 사용된 데이터 기반 선행 계산 기법과 연관이 있다. 많은 연구자들이 물체의 표면에서 이뤄지는 글로벌 레디언스(global radiance)의 전달 작용을 미리 계산한 후 테이블 형태로 저장해놓았다가 시점이나 조명의 변화에 따라 빠르게 렌더링을 수행하는 방식을 연구해왔다[36,37,38]. 제임스(James)와 파타할리안(Fatahalian)은 변형 가능한 모델의 동역학을 미리 계산해서 실시간 시뮬레이션에 활용했다[39]. 이들의 경우 상태-행동 공간의 차원을 감소시키기 위해 제한된 형태의 상호작용만을 허용했다.

3. 상태-행동 모델

이 논문에서 사용하는 강화 학습 모델은 아바타가 이산적인(discrete) 상태 집합 S와 행동 집합 A를 가지며, 동적인 환경 역시 상태 집합 E를 가진다고 가정한다. 따라서 전체 시스템은 상태-행동 순서쌍 $\{(S, E), A\}$로 나타낼 수 있다. 각 시뮬레이션 단계마다 하나의 특정한 상태에 놓여 있는 아바타는 선택 가능한 행동 중 하나를 선택하고, 그 행동은 다시 아바타 및 환경의 상태를 변화시킨다. 아바타는 단지 현재상태에만 의존해서 결정을 내리므로 이 과정은 마르코프 전이 과정으로 볼 수 있다. 학습의 효율성을 높이기 위해서는 상태-행동공간의 차원을 낮추는 동시에 공간을 효과적으로 분할(discretization)하는 것이 중요하다. 사람의 움직임이나 환경의 상태는 모두 매우 높은 차원을 가지기 때문에 학습을 적용하기 위해선 한층 단순화시켜야만 한다. 이 절에서는 전처리 과정으로서 이산적인 상태-행동 공간을 구성하는 방법에 관해 설명한다.

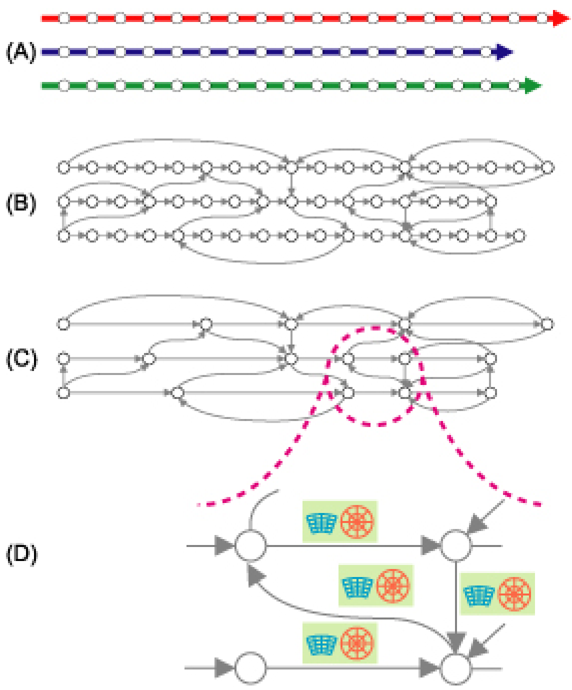

그림 2: 아바타 행동의 상태-행동 모델. (A) 동작 데이터는 초기에 여러 개의 프레임을 포함하는 많은 수의 동작 클립으로 구성되어있다. 각 프레임은 아바타의 자세에 대응된다. (B) 프레임간 전이 에지를 연결시킴으로써 유향 그래프를 만든다. (C) 나가는 에지가 하나인 연속적인 프레임들은 하나로 묶인다. 이 같은 방법으로 단순화된 그래프는 아바타 동작의 상태-행동 모델 (S, A)를 나타낸다. 이 때 그래프의 노드는 상태 S에, 에지는 행동 A에 대응된다. (D) 환경 상태는 격자 형태로 이산 분할되며 각각의 행동에 연결된다. 강화 학습을 통해 생성된 Q-값들은 이 격자 테이블에 저장된다.

사람의 움직임을 표현하는 방식은 전반적으로 이제희 등의 논문을 따른다[6]. 각 노드(node)는 하나의 애니메이션 프레임에 대응되고, 각 에지(edge)는 프레임간 전이에 대응되는 유향 그래프 형태로 표현한 것이다. 이 그래프는 전혀 가공하지 않은 동작 데이터로부터 비슷한 프레임을 찾아 연결시킴으로써 생성할 수 있다. 이제희 등의 생성 방식을 따르면 비교적 적은 수의 프레

임만이 여러 개의 나가는 에지(out-going transition)를 갖게 되며 대부분의 프레임은 한 개의 나가는 에지만을 갖는다. 이 그래프가 완성되면 아바타의 상태 집합 S와 행동 집합 A를 구성할 수 있다. 여러 개의 나가는 에지를 갖는 모든 프레임은 상태 집합 S의 원소가 되고, 한 개의 출력 전이를 갖는 연속적인 프레임들은 하나로 뭉쳐서 행동 집합 A의 원소가 된다. 결과적으로 S의 원소는 아바타의 고정된 자세를 나타내고, A의 원소는 하나의 자세를 다른 자세로 옮기는 전이라고 할 수 있다(그림 2(c)).

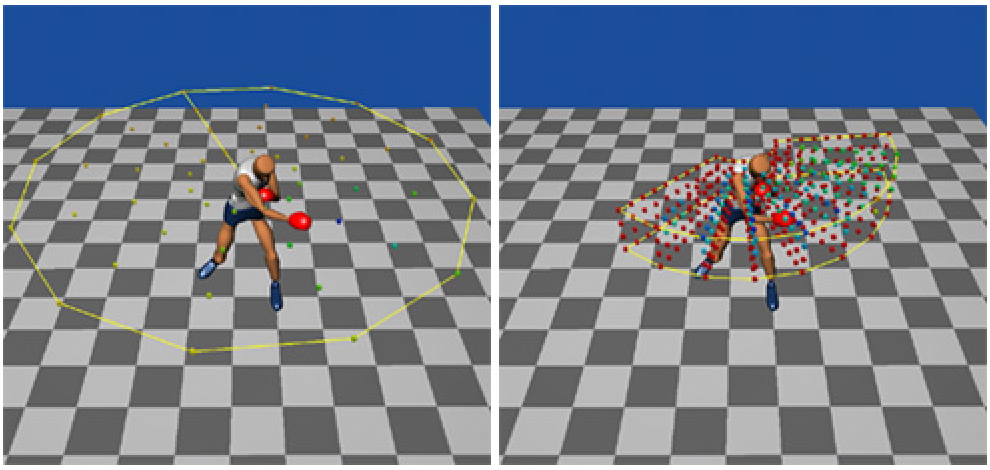

그림 3: 이동하는 목표물의 상태 공간은 행동에 따라 다르게 분할된다: (왼쪽) “목표물에 접근하기” 행동을 학습시키기 위한 상태 공간은 모든 방향에 걸친 2차원 공간이고 비교적 성기다. (오른쪽) “목표물을 가격하기” 행동을 학습시키기 위한 상태 공간은 3차원 공간이고 조밀하다. 이 행동의 경우 상태 공간이 주먹이 닿을 수 있는 범위만을 포괄하면 되기 때문에 격자는 비교적 좁고 작게 형성되어 있다.

이 논문은 동적인 환경에서 아바타를 제어하는데 특히 관심을 갖는다. 예를 들면 움직이는 목표물에 접근하여 가격하는 등의 동작이 여기에 포함된다. 동적인 객체의 위치를 환경 상태에 포함시키기 위해서 우리는 공간을 이산적으로 분할하여 격자(grid) 형태로 표현한다. 이 때 아바타를 중심으로 한 극좌표(polar coordinate)를 사용하면 일반적으로 보다 효과적이다. 격자의 차원(dimension)과 범위(range)는 환경의 기하학적 구성뿐 아니라 학습하고자 하는 행동의 종류도 고려하여 결정한다. 권투 예제에서는 두 가지 행동을 학습시키는데, "목표물에 접근하기"와 "목표물을 가격하기"가 그것이다(그림 3). 후자의 경우 격자는 팔이 닿는 거리까지의 비교적 적은 범위에 대응되고, x, y, z 좌표 모두를 고려해야 하므로 3차원이어야 한다. 반면 목표물에 접근하는 행동의 경우에는 목표물의 높이를 고려할 필요가 없으므로 2차원 격자만으로 충분하고, 목표물이 임의의 위치로 이동할 수 있기 때문에 상대적으로 넓은 범위를 가져야 한다. 목표물이 얼마든지 멀리 있어도 최적의 경로를 따라 접근할 수 있게 학습시키려면 이론적으로는 격자의 너비가 전체 상태 공간의 너비와 일치하도록 해야 한다. 하지만 실제로는 그보다 훨씬 적은 영역에 대해서 지역 탐색만 수행해도 적정 수준의 경로를 찾아갈 수 있다. 격자의 해상도(resolution)는 얼마나 세부적으로 제어하고자 하느냐에 따라 달라진다. 예를 들어 목표물을 가격하는 행동은 정밀하게 제어해야 하므로 조밀한 격자 상에서 학습시키는 것이 좋다. 반대로 목

표물에 접근하는 행동은 대략적인 방향만으로 제어할 수 있기 때문에 보다 성긴 격자에서 학습시켜도 무방하다.

4. 학습

동작 데이터로부터 그래프를 생성하고 나면 아바타에게는 행동을 선택할 수 있는 폭이 매우 넓어진다. 아바타는 매 시뮬레이션 단계마다 현재 상태 $(s, e) \in S \times E$에 따라 하나의 행동 $a \in A$를 선택해야 한다. 최적의 행동을 찾기 위해 그래프를 탐색한다면, 탐색 깊이에 따라 탐색 범위는 지수 함수적으로 증가한다. 이와 같은 많은 탐색 시간이 대화형 아바타를 만드는데 걸림돌이 된다. 이 논문의 목표는 모든 상황에 대해 최적의 행동을 미리 계산해놓음으로써 실행시에는 매우 효율적으로 적절한 행동을 찾도록 하는 것이다. 가장 간단한 방법으로 특정 상태 (s, e)에서 행동 a를 취했을 때의 효용을 계산하기 위해 모든 상태-행동 순서쌍에 대해 탐색 트리를 확장해볼 수 있을 것이다. 하지만 이 방법은 실제 응용에서 지나치게 많은 계산량을 필요로 하는 문제가 있다. (심지어는 전처리 과정으로서도 너무 많은 계산 시간을 요한다.) 이처럼 성능이 크게 떨어지는 가장 큰 원인은 상태-행동공간의 많은 부분이 서로 다른 상태로부터 뻗어 나온 수많은 탐색 트리에 의해 중복 탐색된다는데 있다. 강화 학습은 동적 프로그래밍에 기초하여 이 문제에 대한 해결책을 제공한다.

4.1 강화 학습

강화 학습은 자동화된 에이전트가 반복적인 시행착오를 통해 특정한 행동을 학습하도록 한다. 매번 행동을 할 때마다 학습자는 행동에 대한 보상을 받는다. 이러한 시행이 반복됨에 따라 학습자는 어떤 행동을 취하는 것이 미래의 총 보상을 증가시키는데 유리한지 찾아나간다. 권투의 예를 들면, 권투 선수는 언제나 자신의 현재 상태(자세) s와 목표물 e의 위치를 알고 있으며 그 상태에서 취할 수 있는 모든 행동 중 하나를 즉각적으로 선택한다. 그런데 행동을 취하자마자 그에 대한 보상이 항상 바로 돌아오는 것은 아니다. 즉, 몇 단계 지난 후 목표물이 주먹에 맞았을 때가 돼서야 보상이 돌아올 수 있다. 이처럼 보상이 지연되는 경우를 고려하여 권투 선수는 미래의 총 보상을 최대화하는 일련의 행동을 선택하도록 학습할 수 있을 것이다.

$$G = \sum_{t=0}^{\infty} \gamma^t r_t$$

$0 < \gamma < 1$는 동적 환경의 불확실성을 모델링하기 위해 오래 지연된 보상에 대해 불이익을 주는 상수이다.

특히 이 논문은 Q-러닝(Q-learning)의 아이디어를 차용한다. 왓킨스(Watkins)가 제안한 Q-러닝은 강화 학습기법 중에서 가장 대표적인 기법이다[40]. 기본 아이디어는 상태-행동 순서쌍으로 검색 가능한 테이블을 만드는 것이다. 테이블의 각 항목은 해당 상태-행동 순서쌍을 적용했을 때의 효용 값(Q-값)을 담고 있다. 각 항목에 들어갈 최적의 값은 임의로 상태-행동 순서쌍을 선택한 후 그에 대해 지역 갱신 규칙(local update rule)을 적용함으로써 학습시킬 수 있음이 알려져 있다. 지역 갱신 규칙을 적용하면 행동을 취했을 때 즉시 받는 보상이 효용에 반영되는 동시에 그 값이 점차 이전 상태로 퍼져나간다.

이제 아바타를 학습시키는데 사용할 보상 함수를 정의하자. 보상 r은 환경 상태 e와 행동 a에 대한 스칼라(scalara) 값 함수이다. 모든 행동 a가 특정한 상태 s에서만 선택될 수 있으므로 이후의 식에서는 s를 생략한다. 보상 함수는 학습하고자 하는 행동에 따라 결정된다. 일반적으로 원하는 조건이 만족되는 순간에 매우 짧은 시간 동안 보상 신호가 주어지며, 보상 함수 값은 행동 a가 진행되는 동안의 보상 신호 중 최대값으로 정의된다.

$$r(e,a) = \max{}_t(\gamma^t I(t) w(t))$$

시간 t에 기대 요건이 만족된다면(예를 들어 주먹이 목표물에 맞았다면) $I(t) = 1$이고, 그렇지 않으면 $I(t) = 0$이다. $w(t)$는 가중치 항이다. 어떤 행동은 보상 신호를 연속적으로 전달할 수도 있고, 다음과 같이 수식화된다.

$$r(e,a) = \max{}_t(\gamma^t w(t) \exp(-\frac{|\ e(t) - e_d\ |}{\sigma}))$$

e_d는 도달했을 때 최대의 보상을 받는 상태이다. 이런 연속적 보상 함수를 사용함으로써 학습 과정을 보다 빨리 수렴시킬 수 있다.

테이블의 각 항목 $Q(e,a)$는 상태 e와 행동 a를 가지고 검색할 수 있다. 상태 e는 앞 절에서 설명했듯이 고차원의 격자 구조를 가진다. 우리는 하나의 큰 테이블 대신 여러 개의 작은 Q-값 테이블들을 갖도록 구현하였다. 각각의 작은 테이블은 특정 행동 a에 연결되며 상태 e에 의해 인덱싱된다.

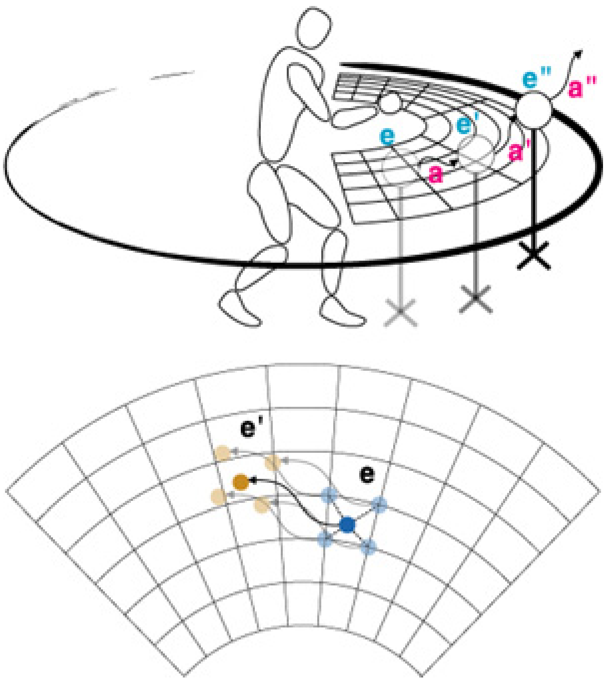

그림 4: Q-러닝의 기본 아이디어는 경험으로부터 배우는 것이다. (위) 학습 단계에서 아바타는 임의의 초기 상태로부터 무작위적으로 연속적인 행동을 취해나간다. (아래) 하나의 상태에서 다음 상태로 전이할 때는 지역 갱신 규칙을 사용하여 처음 상태에 인접한 지점들의 Q-값을 갱신한다.

4.2 학습 과정

일련의 행동을 무작위로 추출하고 그에 따라 지역 갱신 규칙을 적용해서 점차 Q-값 테이블을 수정해나가는 단순한 반복 절차를 통해 아바타의 행동을 학습시킬 수 있다. 우선 임의의 초기 상태로부터 시작한다. 즉 임의의 자세 s와 환경 상태 e(격자 영역 내부에 놓인 목표물의 위치)를 무작위로 선택하는 것이다. 그리고 나서 매 단계마다 현재 아바타 상태 s에서 가능한 모든 행동 중 임의의 행동 a를 선택한다. 행동 a를 취하면 아바타의 상태는 s'으로 옮겨가고 목표물의 상대적인 위치 역시 새로운 e'으로 이동한다. 이제 지역 갱신 규칙을 적용하여 Q-값 테이블의 (e, a) 항목을 수정한다. 이 과정은 모든 상태-행동 순서쌍을 수십 번 방문할 때까지 반복된다. e'이 격자 영역을 벗어날 경우에는 새로운 초기 상태를 다시 선택한 후에 같은 과정을 반복하여 진행한다(그림 4(위)).

Q-값을 갱신하는 표준 규칙은 다음과 같다.

$$Q(e,a) := Q(e,a) + \alpha\,(r + \gamma^t \max{}_{a'} Q(e',a') - Q(e,a))$$

이 식은 상태 e에서 행동 a를 취했을 때 상태는 e'으로 옮겨가고 즉각적인 보상 r이 뒤따름을 뜻한다. 즉각적인 보상 r과 다음 상태의 보상을 감쇠시켜 합한 값을 이용해 $Q(e, a)$의 값을 갱신한다. 감쇠 상수 $0 < \gamma < 1$은 지연된 보상이 역으로 전달될 때 그 값이 시간에 따라 감쇠되도록

하는 역할을 한다. t는 행동 a가 지속되는 시간이다. 학습률 α는 처음에 1로 설정됐다가 학습이 진행되면서 선형적으로 감소함으로써 학습 과정이 수렴하도록 한다.

갱신 규칙에서 e와 e'은 모두 연속적인 변수이고 일반적으로 격자 상의 점 위에 정확히 놓이지 않는다. 때문에 이 논문에서는 불연속적인 격자 상에 이 갱신 규칙을 적용하기 위해 다음과 같은 방식을 사용한다. e에서 e'으로 전이하는 행동 a가 주어졌다고 하면 우리는 e에 근접한 점들에 해당하는 모든 Q-값을 수정한다. e를 모든 인접한 격자 상의 점으로 이동시킨 후 그 상태에서 행동 a를 취했다고 가정하는 것이다(그림 4(아래)). 이 때 e를 각각의 격자 상의 점으로 이동시키는 동시에 e'도 함께 이동시키며, e'에서의 Q-값은 그와 인접한 점들의 Q-값들을 선형 보간하여 얻는다.

5. 실시간 합성

일단 Q-값 테이블이 적절한 값으로 채워졌다면 실시간에 최적의 행동 경로를 찾는 것은 간단하다. 아바타가 현재 취할 수 있는 행동이 두 가지 이상일 경우 그 중 가장 높은 Q-값을 갖는 행동을 선택하면 되는 것이다. 이처럼 단순한 전략만으로 아바타는 주어진 상태 공간의 범위에서 최적의 행동을 수행한다. 아바타가 취할 수 있는 행동의 종류가 여러 개라면(예를 들어, 목표물에 접근하는 행동과 목표물을 가격하는 행동) 각각의 행동에 대한 Q-값을 어떤 식으로든 융합시켜야 한다. 이 논문에서는 이 경우 모든 Q-값의 가중합(weighted sum)을 최대화하는 행동을 선택함으로써 문제를 해결한다.

아바타가 항상 최대의 보상을 받을 수 있는 경로를 선택하도록 하는 것도 중요하지만, 보다 높은 사실성을 위해서는 무작위성(randomness)을 도입할 필요가 있다. 실제 사람의 행동을 보면 그것이 항상 최적의 경로를 따르지는 않음을 알 수 있다. 또한 운동 선수는 때로 상대방을 교란시키기 위해 똑같은 상황에서 의도적으로 다른 행동을 취하기도 한다. 이 논문에서는 아바타의 행동에 무작위성을 부여하기 위해서, 일단 높은 Q-값을 갖는 후보 행동들(candidate actions) 몇 가지를 골라낸 후 임의로 그 중 하나의 행동을 선택하도록 했다.

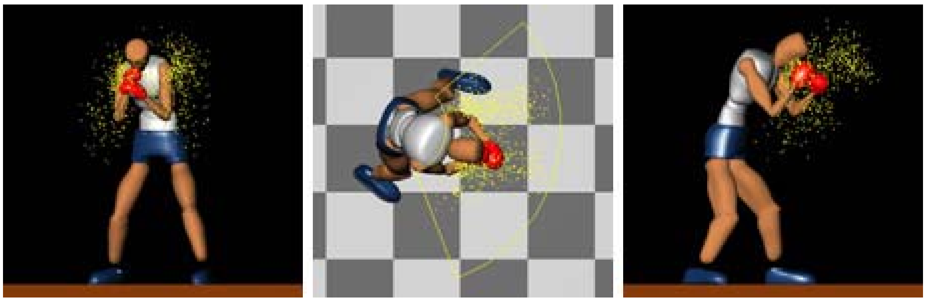

그림 5: 아바타 주위로 효과적인 가격 지점이 분산되어 있는 모습. 차례로 정면, 위, 측면에서 바라본 모습.

6. 실험

실험에 사용된 모든 동작 데이터는 바이콘(Vicon) 광학 시스템을 이용해 초당 120프레임으로 캡쳐한 다음 실시간 출력을 위해 초당 15프레임으로 다운샘플링(down-sampling)하였다. 동작 데이터에는 중심(골반)의 위치와 방향에 대한 경로 및 몸을 구성하는 각 부분에 대한 관절 각도로 이루어져 있다. 시간 측정은 1Gbyte 메모리가 장착된 인텔 펜티엄4 2.4GHz 컴퓨터에서 이루어졌다. 권투 선수 아바타는 다음과 같은 과정을 통해 만들어졌다.

데이터 획득.

동작 데이터를 얻기 위해 프로 권투 선수로부터 8분 가량의 데이터를 캡쳐했다. 연기자는 빈 공간에서 혼자 섀도우 복싱을 하도록 했다. 섀도우 복싱이란 권투 선수가 혼자서 훈련할 때 흔히 사용하는 방식인데, 펀치(punch)와 풋워(footwork)의 다양한 조합을 연습하기 위해 가상의 적을 앞에 두고 대전하는 형식을 말한다. 연기자는 대략 20여 가지의 서로 다른 펀치조합을 수행했다. 오른손 스트레이트(straight) 펀치와 왼손 잽(jab)이 연속되는 동작, 왼손 훅(hook) 다음에 오른손 어퍼컷(uppercut)이 따라오는 동작 등이 여기에 포함된다. 대부분의 조합은 2~4개의 펀치로 이루어졌다. 또한 연기자는 더킹(ducking), 다징(dodging), 블로킹(blocking) 등 방어동작역시 수행했다. 행동 간에 부자연스러운 연결이 일어나는 것을 방지하기 위해 연기자에게는 아무런 시나리오도 제공하지 않았고 특정한 연속 펀치를 요구하지도 않았다. 단지 캡쳐 영역(가로, 세로 5미터) 안에서 동작을 수행하도록 하는 것이 유일한 제약 사항이었다. 연기자의 동작이 자연스럽게 연결되도록 하기 위해서 우리는 한 번에 비교적 긴 시간(대략 90초 정도)씩 캡쳐했다.

데이터 주석 삽입.

동작 간의 부드러운 전이를 생성하고 보상 함수를 계산하려면 환경과 접촉이 일어나는 순간을 식별할 수 있어야 한다. 동작 데이터에는 언제 발이 땅과 접촉하는지, 언제 주먹이 가상의 적을 효과적으로 가격하는지 등에 관한 정보가 명시적으로 포함되어 있지 않다. 구현된 시스템은 동작 데이터에 이러한 정보를 자동으로 첨가해준다. 이제희 등의 논문에서 제시했듯이, 발 근처의 관절 중 하나(발목 혹은 발가락)의 위치가 충분히 바닥에 가깝고 그 속도가 임계 속도보다 느리다면 발이 땅에 접촉한 것으로 간주한다[6]. 펀치의 경우 가격시에 주먹의 진행 방향과 팔뚝의 축이 일치해야 힘을 효과적으로 전달할 수 있다. 이 시스템에서는 팔뚝의 축에 주먹의 속도 벡터를 사영한 결과의 크기가 임계 값 이상이고 두 벡터의 방향이 반대가 아닐 경우에 그 펀치가

효과적이라고 판단한다. 그리고 그 순간의 주먹 끝 지점을 효과적인 가격 지점(effective hitting point)이라고 부른다. 수행 결과 주어진 동작 데이터로부터788개의 효과적인 가격 지점을 찾아냈다(그림 5).

그래프 생성.
프레임 간의 전이 에지를 연결시키는 방식으로 동작 데이터로부터 유향 그래프를 생성했다. 프레임 i에서 프레임 j로 전이 에지가 연결되는 경우는, 프레임 i에서의 자세와 프레임 $j-1$에서의 자세가 유사한 동시에 양쪽 프레임 모두에서 왼쪽 발이 지면으로부터 막 떨어져 나오고 있는 경우이다. 자세 간의 유사성은 관절 각도와 속도의 차이를 고려하여 측정된다[6]. 그래프의 특정 노드로부터 전이할 수 있는 노드가 전혀 없는 상황을 방지하기 위해서 우리는 강하게 연결된 최대 부분 그래프(strongly connected component)를 찾는다. 강하게 연결된 최대 부분 그래프 내에서는 모든 프레임으로부터 다른 모든 프레임까지 전이 에지를 거쳐 도달할 수 있다[4,6]. 실험 데이터에서 찾아낸 강하게 연결된 최대 부분 그래프는 6453개의 프레임을 포함하고 있는데, 이는 대략 7분 17초 정도의 동작에 해당된다. 하나의 나가는 에지를 갖는 프레임들을 뭉치고 나면 총 437개의 프레임과 27072개의 전이가 남는다. 이들 프레임 및 전이가 바로 권투 선수 아바타의 상태 및 행동에 대한 이산적인 집합을 형성한다.

보상 함수.
아바타는 "목표물에 접근하기"와 "목표물을 가격하기"라는 두 종류의 행동을 학습했다. "목표물에 접근하기" 행동에 대한 보상 함수는 다음과 같다.

$$r = \max{}_t(\gamma^t \exp(-\frac{|\mathbf{p}(t) - \mathbf{p_d}|}{10}))$$

$\mathbf{p_d}$는 목표물의 2차원 상의 위치이다. 아바타가 목표물에 접근하고 있을 뿐 아직 가격하지는 않았기 때문에 $p(t)$는 효과적인 가격 지점들의 중심에 해당하는 위치의 2차원 이동 궤적이다. 감쇠 상수 r는 0.97을 사용했다. 모든 좌표는 아바타의 몸에 대해 지역적인 극좌표 시스템을 바탕으로 표현된다. 아바타를 둘러싼 2미터 반경의 영역은 거리에 따라 5등분, 각도에 따라 13등분된 격자 형태로 쪼개져있다(그림 3(왼쪽)).

"목표물을 가격하기" 행동에 대한 보상 함수는 다음과 같다.

$$r = max_t(\gamma^t I(t) w(t))$$

주먹의 위치가 목표물에 충분히 가깝고, 해당 프레임에서 효과적인 가격이 있는 경우 $I(t)=1$이다. 그렇지 않을 경우 $I(t)=0$이다. 함수 $w(t)<1$는 가격 순간에 주먹의 속도를 표준화한 값이며, 가장 빠른 가격 속도일 때 1의 값을 갖는다. 이 함수는 효과적인 가격 지점의 바운딩 볼륨(bounding volume)에 중첩된 10×10×4 격자에 연결되어 있다. 이 바운딩 볼륨은 -1.114~1.317 레디언(radian)의 각도 범위와 29.7~129.3mm의 거리 범위, 그리고 126.6~162.4mm의 높이 범위를 갖는다. Q-값 테이블은 50MB의 저장 공간을 필요로 한다.

학습.

학습은 정적인 목표물에 대해 이루어졌다. 목표물의 초기 위치는 격자 지점 중에서 임의로 선택되고, 아바타는 그 때부터 목표물이 격자를 벗어나기 전까지 동작 그래프 상에서 임의의 행동을 밟아나간다. "목표물에 접근하기" 행동을 학습하는 데는 5000만 번의 무작위 행동을 수행할 때까지 이 과정을 반복했고, 이에 걸린 시간은 대략 2시간 정도였다. 이와 유사하게 "목표물을 가격하기" 행동을 학습하는 데는 총 4억 번의 무작위 행동을 수행했고 15시간이 소요되었다.

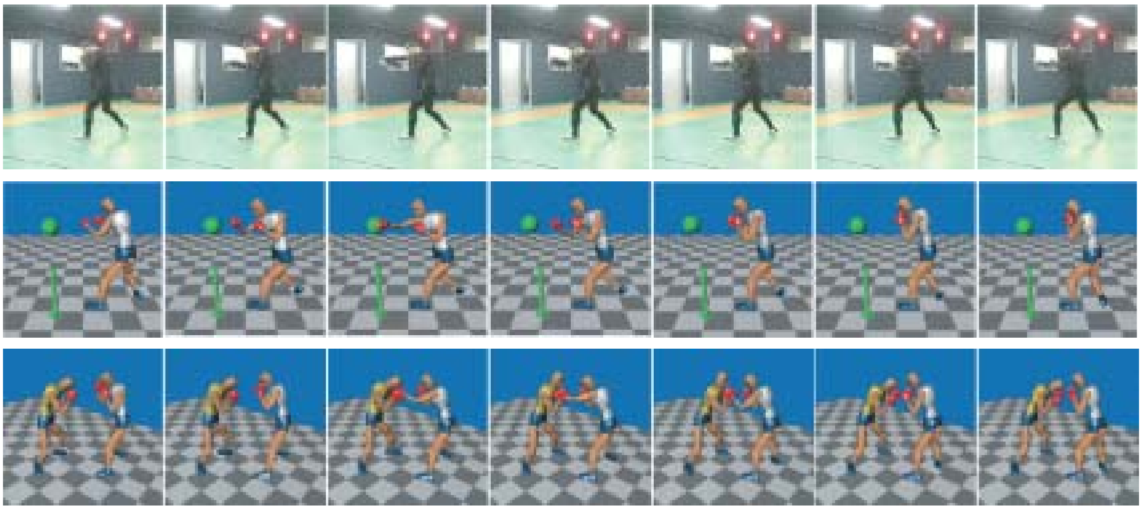

그림 6: 가격하고 그에 반응하는 제어 가능한 권투 선수 아바타들. (위) 동작 데이터는 빈 환경에서 캡쳐되었다. (중간) 권투 선수 아바타는 목표물에 접근해서 가격하는 행동을 학습했다. (아래) 두 권투 선수 아바타가 스파링하고 있다.

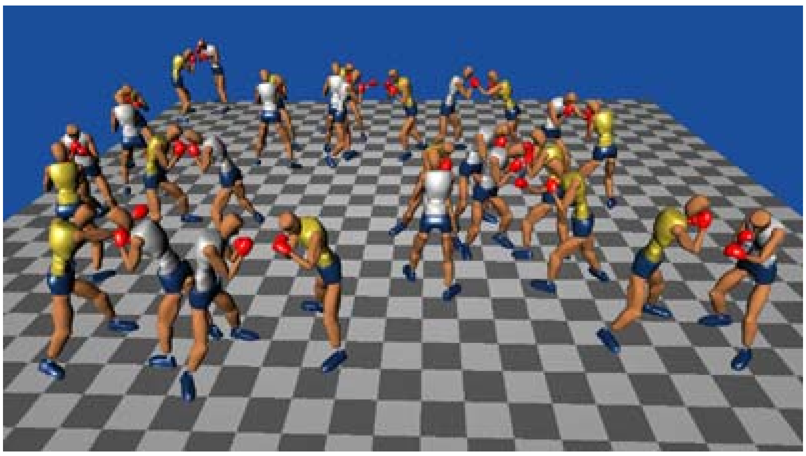

그림 7: 30개의 아바타가 스파링하는 모습.

애니메이션과 제어.

그림 6(중간)의 첫 번째 예제는 권투 선수 아바타가 서있는 목표물을 향해 다가가고 가격하는

모습을 보여준다. 사용자는 언제든 마우스를 이용해서 목표물을 끌어 옮길 수 있다. '접근 행동'은 항상 활성화되어 있으며 '가격 행동'은 목표물이 효과적인 가격 지점의 바운딩 볼륨 내부에 들어왔을 때만 활성화된다. 손과 목표물 간의 충돌 검사는 효과적인 펀치를 갖는 프레임에 대해서만 수행된다. 충돌이 확인되면 목표물은 가격 방향과 반대로 속도에 비례하는 만큼 구부러졌다가 천천히 원래 위치로 되돌아온다. 그림 6(아래)의 두 번째 예제는 두 명의 권투 선수 아바타가 스파링하는 모습을 보여준다. 두 아바타는 똑같은 동작 데이터와 제어 정책을 통해 움직이며 상대의 머리를 목표 지점으로 간주한다. 충돌 검사는 손과 상대방의 상체 간에 이루어진다. 충돌이 일어나면 가격 당한 몸의 부위를 가격 방향에 따라 역운동학(inverse kinematic solver)을 이용해서 이동시킨다.

성능.
시스템의 성능을 측정하기 위해 30개의 아바타가 스파링하도록 만들었다(그림 7). 1000프레임의 비디오 이미지를 만드는 데는 대략 251초가 걸렸는데, 여기에는 렌더링 시간이 상당 부분 점유하고 있다. 비디오와 사운드 생성을 제외하면 같은 애니메이션을 9초 만에 생성할 수 있었다. 이것은 30개의 아바타 움직임을 계산하고 제어하는데 초당 100프레임 이상의 성능을 보인다는 것을 의미한다.

7. 토의
이 논문에서는 아바타가 동작 데이터로부터 제어 정책을 학습함으로써 대화형 애니메이션을 수행하도록 하는 학습 방법을 제시했다. 이 방법은 주어진 상태 공간의 범위와 해상도에 따라 최적화된 제어 정책을 찾아낸다. 상태 공간의 해상도를 충분히 높인다면 이에 따라 학습된 제어 정책은 최적의 연속적인 행동을 찾아낼 수 있고, 이는 [1,4]에서 제시한 전역 탐색알고리즘을 이용해 찾을 수 있는 결과와 일치한다. 이제희 등[6]의 지역 온라인 탐색 알고리즘은 최적의 결과를 찾는 목표와 대화형 응용을 위한 성능목표를 절충한 방법이다. 이 논문은 주어진 격자의 해상도와 범위 내에서 일련의 행동을 찾아내도록 함으로써 [6]과 유사하게 최적해와 성능 목표를 절충했고, 그 결과 탐색 공간을 적절하게 제약할 수 있었다.

실시간 대화형 시스템에 있어서 제어 정책을 학습하는 방식은 지역 온라인 탐색 알고리즘에 비해 월등한 성능 향상을 보여준다. 일단 정책 학습이 끝나고 나면 실시간 동작 합성은 상당한 성능을 가진다. 하지만 제어 정책 학습 방식에는 결점도 존재한다. 한 번 정책이 학습되고 나면 실시간에 최적화의 목표와 변수들을 바꾸는 것이 불가능하고, 결과적으로 여러 개의 목표를 조율하는 것이 어려워지기 때문이다.

이 논문이 제시한 방법은 많은 저장 공간을 필요로 한다. 각각의 행동을 학습하기 위해서는 $O(NM)$ 저장 공간이 필요하다. 여기에서 N은 행동의 개수를, M은 격자에 놓인 점의 개수이다. 본 실험에서는 정적인 목표물에 대해 아바타를 학습시켰다. 이는 실험에 사용한 컴퓨터의 메인 메모리가 각각의 행동에 대해 단지 3차원 격자만을 수용할 수 있는 정도였기 때문이다. 기계 학습 분야의 최근 결과는 근사 함수(function approximator)와 적응형 해상도 모델(adaptive resolution model)을 사용함으로써 방대한 상태 공간을 간결하게 저장할 수 있음을 보여준다[31,32]. 무어와 앳커슨[41]은 자신들이 제시한 적응형 해상도 모델로 최대 9차원까지의 상태 공간에 대한 정책을 학습할 수 있다고 보고했다. 고차원 상태 공간에서 학습을 수행

한다면 정적인 목표물이 아닌 실제 상대방과의 스파링을 통한 정책 학습과 같이 보다 설득력 있는 학습 시나리오를 적용할 수 있을 것이다.

저자들은 이 논문에서 제시한 방법이 권투 이외에도 폭넓은 범위의 인간 행동에 일반화될 수 있다고 기대한다. 일반화를 할 때 중요하게 고려해야 할 것 하나는 레이블링(labelling)이 되어있지 않은 동작 데이터로부터 아바타와 환경 간의 상호작용을 파악하는 것이다. 현재의 동작 캡쳐 시스템으로는 물리적 상호작용과 접촉 등을 쉽게 찾아내기는 어렵다. 권투 예제에서는 어떤 물리적 목표물도 없이 캡쳐를 수행한 후에 데이터를 분석함으로써 효과적인 가격 지점을 식별해냈다. 일반적인 인간 행동으로의 확장을 위해선 이 밖에 여러 종류의 물리적 상호작용이나 시선의 방향과 같은 간접적 상호작용도 캡쳐할 수 있어야 할 것이다.

여러 개의 아바타를 동시에 애니메이션할 때는 충돌 검사 및 반응이 중요한 시각적 신호로 작용한다. 우리는 충돌시 발생하는 물리적 상호작용은 배제했고, 단지 충돌이 일어났음을 보여주기 위해서 접촉이 발생한 몸의 부위를 다소 임의적으로 이동시키는 방식을 택했다. 조단(Zordan)과 호진스(Hodgins)는 시뮬레이션되는 인간의 충돌 반응을 계산하기 위해 물리적 충돌 모델을 사용했다[42]. 이와 같은 물리 기반 방식을 실시간 대화형 시스템에 통합하는 작업은 앞으로 연구할 만한 흥미로운 영역의 하나이다.

감사의 글

이 논문은 2005년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구입니다. (KRF-2005-205-R08-2003-000-10167-0)

참고문헌

[1] Arikan, O., and Forsyth, D. A., "Interactive motion generation from examples," Proceedings of SIGGRAPH 2002, pp. 483-490, 2002.
[2] Brand, M., and Hertzmann, A., "Style machines," Proceedings of SIGGRAPH 2000, pp. 183-192, 2000.
[3] Galata, A, Johnson, N., and Hogg, D., "Learning variable length markov models of behaviour," Computer Vision and Image Understanding (CVIU) Journal, Vol.81, No.3 (March), pp. 398-413, 2001.
[4] Kovar, L., Gleicher, M., and Pighin, F., "Motion graphs," Proceedings of SIGGRAPH 2002, pp. 473-482, 2002.
[5] Kim, T., Park, S. I., and Shin, S. Y., "Rhythmic-motion synthesis based on motion-beat analysis," ACM Transactions on Graphics (SIGGRAPH 2003), Vol.22, No.3, pp. 392-401, 2003.
[6] Lee J., Chai, J., Reitsma, P. S. A., Hodgins, J. K., and Pollard, N. S., "Interactive control of avatars animated with human motion data," Proceedings of SIGGRAPH 2002, pp. 491-500, 2002.
[7] Li Y., Wang T., and Shum, H.-Y., "Motion texture: a two-level statistical model for character motion synthesis," Proceedings of SIGGRAPH 2002, pp. 456-472, 2002.
[8] Pullen , K., and Bregler, C., "Motion capture assisted animation: Textureing and synthesis," Proceedings of SIGGRAPH 2002, pp. 501-508, 2002.
[9] Sidenbladh, H., Black, M. J., and Sigal, L., "Implicit probabilistic models of human motion for synthesis and tracking," European Conference on Computer Vision (ECCV), pp. 784-800, 2002.
[10] Molina Tanco, L., and Hilton, A., "Realistic synthesis of novel human movements from a database of motion capture examples," Proceedings of the Workshop on Human Motion, pp. 137-142, 2000.
[11] Blumberg, B. M., and Galyean, T. A., "Multi-level direction of autonomous creatures for real-time virtual environments," Proceedings of SIGGRAPH 95, pp. 47-54, 1995.
[12] Blumberg, B., "Swamped! Using plush toys to direct autonomous animated characters," SIGGRAPH 98 Conference Abstracts and Applications, p.109, 1998.
[13] Bruderlin, A., and Calvert, T. W., "Goal-directed, dynamic animation of human walking," Computer Graphics (Proceedings of SIGGRAPH 89), Vol.23, pp. 233-242, 1989.
[14] Noma, T., Zhao, L., and Badler, N. I., "Design of a virtual human presenter," IEEE Computer Graphics & Applications, Vol.20, No.4 (July/August), 2000.
[15] Perlin, K., and Goldberg, A., "Improv: A system for scripting interactive actors in virtual worlds," Proceedings of SIGGRAPH 96, pp. 205-216, 1996.
[16] Badler, N. I., Hollick, M., and Granieri, J., "Real-time control of a virtual human using minimal sensors," Presence 2, pp. 82-86, 1993.

[17] Dontcheva, M., Yngve, G., and Popovic, Z., "Layered acting for character animation," ACM Transaction of Graphics (SIGGRAPH 2003), Vol.22, No.3, pp. 409-416, 2003.
[18] Molet, T., Boulic, R., and Thalmann, D., "A real-time anatomical converter for human motion capture," EGCAS '96: Seventh International Workshop on Computer Animation and Simulation, Eurographics, 1996.
[19] Semwal, S., Hightower, R., and Stansfield, S., "Mapping algorithms for real-time control of an avatar using eight sensors," Presence 7, No.1, pp. 1-21, 1998.
[20] Shin, H. J., Lee, J., Shin, S. Y., and Gleicher, M., "Computer puppetry: An importance-based approach," ACM Transactions on Graphics, Vol.20, No.2, pp. 67-94, 2001.
[21] Bradley, E., and Stuart, J., "Using chaos to generate choreographic variations," Proceedings of the Experimental Chaos Conference, 1997.
[22] Pullen, K., and Bregler, C., "Animating by multi-level sampling," Computer Animation 2000, IEEE CS Press, pp. 36-42, 2000.
[23] Bowden, R., "Learning statistical models of human motion," IEEE Workshop on Human Modelling, Analysis and Synthesis, CVPR2000, 2000.
[24] Choi, M. G., Lee, J., and Shin, S. Y., "Planning beped locomotion using motion capture data and probabilistic roadmaps," ACM Transactions on Graphics, Vol.22, No.2, pp. 182-203, 2003.
[25] Arikan, O., and Forsyth, D. A., "Interactive motion generation from examples," Proceedings of SIGGRAPH 2002, pp. 483-490, 2002.
[26] Ngo, J. T., and Marks, J., "Spacetime constraints revisited," Proceedings of SIGGRAPH 93, pp. 343-350, 1993.
[27] Sims, K., "Evolving virtual creatures," Proceedings of SIGGRAPH 94, pp. 15-22, 1994.
[28] Grzeszczuk, R., and Terzopoulos, D., "Automated learning of muscle-actuated locomotion through control abstraction," Proceedings of SIGGRAPH 95, pp. 63-70, 1995.
[29] Grzeszczuk, R., Terzopoulos, D., and Hinton, G., "Neuroanimator: fast neural network emulation and control of physics-based models," Proceedings of SIGGRAPH 98, pp. 9-20, 1998.
[30] Faloutsos, P., Van De Panne, M., and Terzopoulos, D., "Composable controllers for physics-based character animation," Proceedings of SIGGRAPH 2002, pp. 251-260, 2001.
[31] Kaelbling, L. P., Littman, M. L., and Moore, A. W., "Reinforcement learning: A survey," Journal of Artificial Intelligence Research 4, pp. 237-285, 1996.
[32] Sutton, R. S., and Barto, A. G., "Reinforcement Learning: An Introduction," MIT Press, 1998.
[33] Atkeson, C., Moore, A., and Schaal, S., "Locally weighted learning for control,"

AI Review 11, pp. 75-113, 1997.
[34] Mataric, M. J., "Reward functions for accelerated learning," Proceedings of the Eleventh International Conference on Machine Learning, 1994.
[35] Blumberg, B., Downie, M., Ivanov, Y., Berlin, M., Johnson, M. P., and Tomlinson, B., "Integrated learning for interactive synthetic characters," Proceedings of SIGGRAPH 2002, pp. 417-426, 2002.
[36] Ng, R., Ramamoorthi, R., and Hanrahan, P., "All-frequency shadows using non-linear wavelet lighting approximation," ACM Transactions on Graphics (SIGGRAPH 2003), Vol.22, No.3, pp. 376-381, 2003.
[37] Sloan, P.-P., Hall, J., Hart, J., and Snyder, J., "Clustered principal components for precomputed radiance transfer," ACM Transactions on Graphics (SIGGRAPH 2003), Vol.22, No.3, pp. 382-391, 2003.
[38] Sloan, P.-P., Liu, X., Shum, H.-Y., and Snyder, J., "Bi-scale radiance transfer," ACM Transactions on Graphics (SIGGRAPH 2003), Vol.22, No.3, pp. 370-375, 2003.
[39] James, D. L., and Fatahalian, K., "Precomputing interactive dynamic deformable scenes," ACM Transactions on Graphics (SIGGRAPH 2003), Vol.22, No.3, pp. 879-887, 2003.
[40] Watkins, C. J. C. H., and Dayan, P., "Q-learning," Machine Learning, Vol.8, No.3, pp. 279-292, 1992.
[41] Moore, A., and Atkeson, C., "The parti-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces," Machine Learning, Vol.21, 1995.
[42] Zordan, V. B., and Hodgins, J. K., "Motion capture-driven simulations that hit and react," Proceedings of ACM SIGGRAPH Symposium on Computer Animation, pp. 89-96, 2002.