집합 간선 그래프를 이용한 상호 작용적 연속 제어 캐릭터 생성

신현준, 오현석

아주대학교

{joony,ohs108}@ajou.ac.kr

Fat Graph: Constructing characters with continuous controls

Hyun Joon Shin and Hyun Seok Oh

Ajou University

요 약

본 연구는 실행 시(run-time)에 사용자의 연속적인 입력에 따라 움직이는 고품질의 애니메이션을 적은 계산 량으로 합성하는 방법을 제안한다. 본 연구의 목적은 Gleicher 등 [5] 의 연구에 기반하여 집합 간선 그래프(fat graphs)를 반 자동으로 생성하는 것이다. 여기서 집합 간선 그래프는 동작 데이터를 구성하는 프레임 중 특정한 자세를 노드로 하고 노드의 자세로부터 시작하거나 끝나는 동작 절편의 집합들(motion segments)을 간선으로 하는 그래프를 일컫는다. 입력 동작에 대하여 이러한 집합 간선 그래프를 생성하고 동작 합성 및 변환 기법을 통해 가공한 후, 실행 시에 사용자가 마우스, 키보드, 조이스틱과 같은 입력 장치를 통해 캐릭터를 원하는 대로 움직일 수 있도록 한다. 몇 가지 제약 조건이 존재하지만 비디오 게임을 비롯한 여러 응용 소프트웨어에서 본 연구의 결과를 사용할 수 있을 것으로 기대한다.

1. 서 론

최근 강력한 기능을 제공하는 3D 그래픽 하드웨어들이 등장하면서 사용자의 명령에 의해 움직이는 캐릭터들(interactively controllable characters)이 비디오 게임을 비롯한 영화, 방송, 가상 현실 응용 소프트웨어 등 다방면에서 널리 사용되고 있다. 이와 같은 캐릭터들이 사실적이기 위해서는 캐릭터들의 동작을 동작 포착 등을 통해 생성된 동작 정보를 이용하여 효율적으로 합성해야 한다.

지금까지 이러한 캐릭터를 구현하기 위하여 진행된 연구들 중 가장 성공적인 연구 결과로 그래프에 기반한 방법과 동작 혼합(motion blending)방법을 들 수 있다. 그래프에 기반한 방법은 동작 데이터들 간의 연결성을 고려하여 그래프를 생성하고, 이를 이용하여 필요한 애니메이션을 재구성하는 방법이다. 동작 혼합 기법은 기존에 주어진 데이터들을 혼합하여 동작의 특징을 나타낸 매개변수 공간 내에서 연속적으로 캐릭터의 움직임을 제어할 수 있도록 하는 방법이다.

본 논문에서 제안하고자 하는 방법은 사용자의 명령에 따라 빠르고 정확하게 움직이는 사실적인 애니메이션을 합성하는 방법이다. 이를 위해 집합 간선 그래프(fat graphs)라는 개념을 제안한다. 집합 간선 그래프란 동작 정보 중 몇 자세를 노드로 하고, 각 자세를 시작과 끝으로 하는 동작 절편(motion segment)을 진입 간선(incoming edges)과 진출 간선(outgoing edges)으로 하는 유향 그래프이다. 각 간선들은 유사성에 따라 간선 집합을 구성한다. 사용자는 각 노드에 연결된 여러 개의 진출 간선 집합 중 하나를 선택할 수 있고, 선택된 집합 간선 내의 동작 절편들을 혼합하여 연속된 매개변수 공간 내에서 동작을 제어할 수 있다.

그래프와 동작 혼합 기법을 이용한 캐릭터 동작 제어 방법은 각각 게임 업계 등에서 지난 수 십 년간 사용되어 온 방법이다. 하지만 대부분의 경우 그래프를 만들고 동작 절편들을 매개변수화 하는 작업이 수작업을 통해 이루어 졌다.

본 연구는 이를 개선하여 그래프의 생성과 동작 절편의 매개변수를 정하는 과정을 반자동화 하고자 한다.

본 연구는 크게 세 가지 측면에서 기여한다. 첫 번째, 사용자의 명령에 의해 움직이는 캐릭터를 만들고, 동작의 특징을 나타내는 연속적인 매개변수 공간에서 캐릭터를 제어하는 기존의 두 방법을 결합시키기 위한 새로운 자료구조를 제안한다. 두 번째, 기하적인 특징을 이용하여 동작 절편들을 매개변수화 하는 방법을 제안한다. 세 번째, 실행 시에 위의 내용들을 이용하여 조이스틱이나 마우스 등과 같은 입력장치로 캐릭터 모션을 제어하는 방법을 제시한다.

이 논문은 크게 다섯 부분으로 구성되어 있다. 두 번째 절에서 본 연구와 관련된 연구들을 살펴볼 것이다. 세 번째와 네 번째 절에서는 사용자의 명령에 의해 움직이는 캐릭터를 저작(authoring)하는 방법을 살펴본다. 다섯 번째 절에서는 연속적으로 제어할 수 있는 동작을 합성하는 방법을 제안하고, 마지막으로 여섯 번째 절에서는 본 연구의 실험 결과를 살펴본 후 마지막 절에서 본 연구에서 제안한 방법들에 대한 논의점과 한계점에 대하여 논한다.

2. 관련 연구

주어진 동작 절편들을 이용하여 사용자가 원하는 형태의 동작을 합성하는 동작 혼합 기술은 본 연구와 관련하여 가장 널리 연구되고 있는 주제이다. Wiley와 Hahn은 사용자가 정한 위치를 지나가는 동작을 합성하는 방법을 제안하였다 [17]. 이들은 몸이 닿는 공간상에 밀집 격자를 지정하고 격자상의 각 점에 도달하도록 만들어진 동작 절편을 이용하여 기하학적으로 매개변수화 하였다. Rose 등은 동작 절편의 특성을 파악하여 주어진 동작 절편을 매개변수 공간에 가져다 놓은 후, 사용자가 정한 매개변수 공간과 일치하는 동작 절편의 가중치를 계산하기 위해 원형 근간 함수(RBF)을 사용하였다 [15]. Park 등은 온라인 실시간 이동 동작 발생 방법(on-line real-time locomotion generation)이라는 동작 혼합 기술을 고안했다 [13]. 속도, 회전각, 스타일에 기반한 동작 절편의 다양성을 매개변수화 하였고, 동작 절편의 가중치를 계산하기 위하여 역시 원형 근간 함수를 사용하였다. Kovar와 Gleicher는 두 동작을 혼합하는 일반적인 방법을 제안하였다 [7]. 이들은 커다란 동작 데이터베이스로부터 혼합할 수 있는 조각들을 찾고 이것을 자동으로 매개변수화 하는 것으로 연구를 확장하였다. Mukai와 Kuriyama는 지정통계적인 혼합 방법(geostatistical blending method)으로 기존의 방법들 보다 좋은 품질의 동작을 생산하는 방법을 선보였다 [12]. 본 연구는 Wiley 와 Hahn [17], 그리고 Kovar와 Gleicher [7] 의 연구로부터 동기를 얻었다. 본 연구에서는 위 연구들을 토대로 보다 간단한 인터페이스를 통해 효과적으로 실행 시(run-time)동작을 제어 할 수 있도록 하였다.

그래프를 이용한 연구들 [1, 9, 10, 3, 2, 5, 6, 11] 은 주어진 동작 데이터를 이용하여 새로운 애니메이션을 만들어 내는 가장 성공적인 접근 방법중 하나이다. 이 연구들은 주어진 동작의 애니메이션 프레임들 간의 연결성을 저장하기 위해서 특별한 자료 구조를 만든다. 이 때 저장된 연결성은 새로운 애니메이션을 만들어 낼 때, 각 프레임 간의 연결 가능성을 판별하기 위하여 사용된다. 본 연구는 특히 Gleicher 등 [5] 의 연구에 영향을 받았다. 그들은 위의 자료 구조를 형성 하는 방법과 함께 사용자의 명령에 의해 움직이는 캐릭터 애니메이션을 효과적으로 제작할 수 있는 특별한 구조를 고안하였다. 본 연구에서는 [5] 에서 사용한 자료 구조와 매우 유사하나 보다 개선된 자료 구조를 사용 하였다.

단순한 사용자 인터페이스를 사용하여 사용자의 제어가 가능한 애니메이션을 만들어내는 것은 컴퓨터 애니메이션에서 중요한 논점으로 여겨져 왔다. Davis 등의 스케치 인터페이스는 애니메이터가 키 프레임에서 2차원 자세를 스케치하도록 하였고, 이 스케치로부터 3차원 자세를 얻고, 3차원 자세를 재구성하여 보간하여 3차원 캐릭터 애니메이션을 만들었다 [4]. Thorne 등은 [16] 사용자가 입력한 곡선들(a sequence of cursive strokes)로 부터 캐릭터 애니메이션으로 생성하는 스케치 기반의 사용자 인터페이스 시스템을 고안하였다. 본 논문에서 제안하고자 하는 방법은 캐릭터 애니메이션의 안무(choreographing)보다는 게임과 같은 사용자의 제어가 가능한 응용 소프트웨어에서 캐릭터의 동작을 제어하는 방법이다. 그 결과 본 연구에서는 애니메이션이 만들어지는 동안 어떻게 하면 동작을 보다 효과적으로 제어할 수 있을까에 초점을 맞추었다.

3. 집합 간선 그래프의 저작

집합 간선 그래프는 유사한 자세들로 구성된 노드와 유사한 동작 절편들로 만들어진 간선으로 구성된다. 사용자는 이 그래프의 간선들을 지나면서 해당되는 동작 절편들의 혼합하여 캐릭터의 애니메이션을 만들어낸다. 그래프 구조 내에서 사용자는 이전 애니메이션에 잘 연결되는 동작들로 만들어진 진출 간선 중 하나를 선택한다. 집합 간선 그래프의 간선은 여러 개의 유사한 동작 절편들을 포함하기 때문에 선택된 간선에 해당하는 동작 절편들을 혼합하여 사용자가 정의한 동작을 만든다(그림 1).

집합 간선 그래프에서 특정 노드의 진출 간선들은 그 노

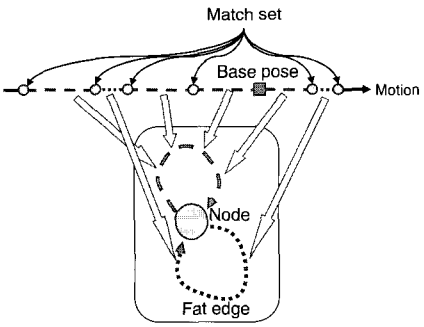


그림 1: 집합 간선 그래프의 구성

드에 해당되는 자세에서 시작하는 동작들에 해당한다. 따라서 적은 수의 노드가 많은 수의 진출 간선을 갖으면 매 번 사용자는 수많은 행동 중 하나를 선택할 수 있게 된다. 또한 혼합할 수 있는 동작 절편들이 많으면 보다 다양한 종류의 동작을 만들어낼 수 있다. 결과적으로 동작 합성에서 다양성은 그래프의 각 노드가 얼마나 많은 진출 간선을 가지고 있으며, 같은 노드의 진출 간선들이 얼마나 다양한 혼합 가능한 동작을 포함하고 있는가에 따라 결정된다.

3.1 그래프 구조 생성

인위적으로 구조를 조정하여 생성한 그래프는 Gleicher 등 [5] 이 *Snap-together motion*에서 제안한 중추 노드(hub-node)의 개념을 이용하여 만들 수 있다. 중추 노드는 여러 진입 간선과 진출 간선을 포함하고 있는 그래프 노드이다. 각 노드의 진입 간선과 진출 간선의 시작과 끝 자세가 거의 일치해야 하기 때문에, 중추 노드에 해당하는 자세는 많은 동작과 연결될 수 있는 자세이어야 한다. 예를 들어 무술 동작들의 데이터 집합에서 중추 노드는 준비 동작이 될 것이고, 걸어가는 동작의 경우는 어느 프레임에서 시작하든지 계속 반복되기 때문에 어느 동작이라도 중추 노드가 될 수 있다.

중추 노드는 자동 혹은 반자동으로 만들 수 있다. 그래프에 연결성을 높이기 위해 유사한 자세가 가장 빈번히 나타나는 자세를 찾아 자동적으로 중추노드로 결정할 수 있다. 이를 위해 먼저 주어진 모든 자세들 사이의 거리를 구해야 한다. 본 연구에서는 Kovar 등 [9] 이 제안한 방법을 사용하여 각 자세 간의 차이를 계산하였다. 이 방법을 이용하여 특정한 자세와 다른 모든 자세간의 차이를 계산하고 이들 중 차이가 사용자가 지정한 임계값보다 작고 주변 프레임에 비해

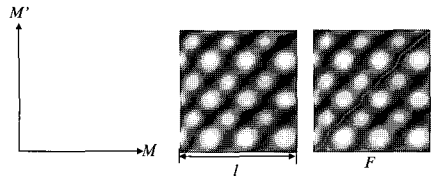


그림 2: 두 동작 절편 사이의 거리

주어진 자세와 더 유사한 극소값에 해당하는 프레임들을 선택하여 이들을 유사 집합(match set)으로 지정한다. 본 연구에서 제안한 시스템은 가장 큰 유사 집합을 가지는 기본 동작을 중추 노드로 결정하여 사용자에게 제안한다. 여기서 임계값은 그래프의 연결성과 결과 동작의 품질을 조절하는 역할을 한다.

시스템이 중추 노드와 유사 집합을 제시하면, 사용자는 시스템이 정한 결과를 사용하거나 혹은 사용자 임의로 보다 좋은 자세라고 여겨지는 것을 직접 기본 자세로 결정 할 수 있다. 중추 노드가 결정되고 나면 시스템은 사용자에게 그에 해당하는 유사 집합을 제공한다. 본 연구를 통해 개발된 시스템에서는 사용자가 좀 더 나은 동작 분할을 위해 유사 집합의 원소들을 추가하거나 삭제, 변경할 수 있게 하였다. 실험 결과를 살펴보면, 시스템은 거의 예외 없이 사용자의 수정이 거의 필요 없는 기본 자세와 유사 집합을 제공하였다. 또한 사용자가 원하는 그래프 구조를 얻을 때까지 계속 중추 노드를 추가 할 수 있다.

3.2 집합 간선

중추 노드(hub node)가 결정된 후, 같은 노드로 부터 시작하는 간선들 중 유사한 동작을 가지고 있는 간선들을 묶어 집합 간선을 구성한다. 동일한 노드로 연결되면서 서로 유사한 동작 절편을 포함하는 간선들은 서로 혼합되어 사용자가 원하는 동작을 생성할 수 있다. 이를 위해 동작 절편 사이의 거리를 측정하고 유사한 동작을 분류하는 작업이 이루어져야 한다.

두 동작 절편 사이의 거리는 대응 되는 자세들의 차이의 제곱 합(squared sum)을 동작 길이로 정규화(normalize)한 값으로 정의 될 수 있다. 좀더 자세하게 살펴보면, 먼저 동작 절편 사이의 각 프레임 간의 대응관계(dense temporal correspondence)를 생성한다. 간선들의 동작 절편은 비교적 짧고, 서로 유사하지 않은 동작 절편의 쌍 사이에 합리적인 대응관계를 만들어낼 필요가 없기 때문에, 본 시스템에서는 단순하

게 일반적인 동적 시간 왜곡 기술(dynamic time warping technique)을 채택하였다. 동작 M으로부터 M'까지 일치 맵 F가 주어지면, M 과 M' 사이의 거리는 다음과 같이 정의된다.

$$\sum_{t}^{l} \frac{d\left(M(t), M'\left(F(t)\right)\right)^2}{l}, \quad (1)$$

이때 $M(t)$는 동작 M의 시간 t에서의 자세고, $d(\cdot)$은 두 자세 사이의 거리이다. l은 M의 길이 이다(그림 2).

유사한 동작을 가지는 간선들을 분류하기 위해 우선 모든 동작 절편에 각각의 집합을 할당한다. 그 후 각각의 동작 절편을 비교하여 유사한 경우에는 두 집합을 병합한다. 각 집합간에 병합이 이루어지기 위해서는 집합 내의 동작 절편 각각의 시작 노드와 끝 노드가 일치해야 한다. 이렇게 양 끝 노드가 일치하면 두 집합의 내에서 각각 추출한 임의의 동작 쌍 간의 거리를 비교하여 거리 값이 사용자가 정한 경계범위 이내에 포함될 경우 두 집합의 동작 절편은 서로 유사하다고 판단하고, 두 집합을 서로 병합한다. 이러한 과정을 계속 반복하게 되면 서로 혼합이 가능한 간선의 집합이 만들어 지게 된다. 본 연구의 시스템에서는 사용자 임의로 집합 간선들을 구성할 수 있도록 하기 위해 경계 값을 변경하거나, 집단 내의 동작 절편들을 추가 또는 삭제할 수 있도록 하였다.

4. 동작의 매개변수화

모든 집합 간선들은 실행 시에 서로 혼합 할 수 있는 다수의 동작 절편 들로 구성되어 있다. 사용자가 혼합 가중치를 결정하는 것은 매우 어려운 일이기 때문에 자료 구성 단계에서 사용자들이 편리하게 값을 지정 할 수 있도록 해주어야 한다. 본 연구는 많은 사용자 들이 어느 한 시점에 특정 관절이 지나가는 지점을 제어할 수 있기를 요구한다는 점에 착안하였다. 예를 들어 걸어가는 동작을 제어하기 위해서는 매 시점 마다 몸의 중심 위치가 특정한 값을 가질 수 있도록 하는 것이 합리적이다. 한편 발차기 동작에서는 발을 닿는 순간 특정 위치를 타격하는 동작을 필요로 할 것이다. 그 결과 본 시스템은 사용자가 원하는 시점에 선택한 관절이 지나갈 수 있는 영역을 제시하고 그 위치를 결정할 수 있도록 하였다.

동작 절편들의 혼합을 통해 얻을 수 있는 관절들의 위치는 가중치와 항상 선형적 관계를 가지고 있는 것이 아니기 때문에, 본 시스템에서는 다양한 가중치에 따른 관절의 여러 위치들을 추출한 Kovar 등 [8] 의 방법을 활용하였다. 각 동작 절편의 가중치를 임의로 얻고, 5.1에서 살펴 볼 동작 혼합 과정을 통해 사용자가 지정한 시각에 관절 위치를 얻어낸다. 실험을 통해 수백 개 정도의 점들을 추출함으로써 주어진 동작 절편들 간의 혼합에 의해 얻어진 관절의 허용 공간을 형성할 수 있었다.

한 편, 실행 시에 동작을 제어할 2차원 입력 장치들을 이용하기 위해서 동작을 2차원적으로 매개변수화해야 한다. 이 작업은 평면 결정 및 투영, 변수들의 축 결정, 공간 분할, 정규 추출의 네 단계로 이루어진다(그림 3). 첫 번째 단계에서는 전 단계에서 임의로 얻은 점들을 2차원 평면에 투영시켜 2차원 매개변수 공간을 구성한다. 실행 시에 좀더 직관적인 제어를 위해 공간의 축을 새로 지정하여 제어장치의 축을 배치한다. 또한, 사용자가 지정한 점에 해당하는 가중치를 계산할 때 주변의 세 투영점의 가중치 값을 보간하여 사용할 수 있도록 투영점들이 이루는 공간을 삼각분할 한다. 그 후 실행 시 계산 량을 줄이기 위해 공간에 정규 격자점에 대하여 가중치를 계산한다.

평면 결정 및 투영: 연속된 값들을 발생하는 일반적인 입력 장치들은 두 축을 가지고 있기 때문에, 앞서 임의로 얻은 관절의 위치를 2차원 공간에 투영해야 하여 2차원 매개변수 공간을 생성한다. 관절의 위치들을 기반으로 주축 분석(principal component analysis)를 수행하여 적절한 평면을 얻고 관절 위치들을 평면에 투영하여 매개변수 공간을 구성한다.

매개변수 축 결정: 주축분석에서 얻은 축은 일반적으로 사용자가 동작을 직관적으로 제어하기 위한 축과 일치하지 않는다. 따라서 입력 장치의 축을 이용하여 동작을 직관적으로 제어할 수 있도록 평면상의 두 직교 축을 결정할 필요가 있다. 많은 경우에 사용자는 장치의 y축이 매개변수 공간의 y축에 할당하는 것이 적합하다. 혹은 매개변수 공간이 평면적이라면 장치의 y축은 매개변수 공간에서 캐릭터의 정면 방향과 일치하는 것이 직관적인 제어를 가능하게 한다. 따라서 먼저 y축과 매개변수 공간을 나타내는 평면의 법선 벡터 사이의 각을 측정하고, 만약 각이 45도 보다 적을 경우, 평면 위의 y축을 투영하여 장치의 y축에 해당하는 축을 얻는다. 각이 45도 보다 클 경우, 일반적으로 캐릭터의 정면을 나타내는 z축을 캐릭터의 중심 방향에 따라서 회전하여 정면 방향을 계산하고 매개변수 평면에 투영하여 장치의 y축에 해당하는 축을 얻는다. 장치의 x축은 평면상에서 결정된 y축과 직교하는 축으로 결정할 수 있다.

공간 분할: 평면 위에 점이 주어지면, 그 점을 지나가는 동작을 생성하기 위한 가중치의 조합을 알아내야 한다. 이미 평면 위의 수 많은 투영점들이 존재하고 각각의 가중치 조합을 알고 있기 때문에, 주어진 점에서의 가중치 조합은 주변에 존재하는 투영점들의 가중치 조합을 이용하여 계산

그림 3: 발차기 동작의 매개변수화

할 수 있다. 효과적으로 주변의 투영점들을 찾기 위해서, 투영점들을 정점으로 하여 매개변수 공간을 Delaunay 삼각형화 방법을 이용하여 삼각 분할한다. 매개변수 공간이 삼각분할 되면 주어진 점의 가중치 조합은 주어진 점을 포함하는 삼각형의 세 꼭지점의 가중치 조합을 선형 보간하여 계산할 수 있다.

격자점 구성: 실행 시 사용자가 지정한 점을 포함하는 삼각형을 삼각 분할 된 매개변수 공간에서 검색하는 것은 게임과 같은 상호작용 응용 소프트웨어에서는 특히 비용이 많이 소모되는 작업이다. 따라서, 이 작업을 보다 효율적으로 수행하기 위해, 2차원 격자 형태의 규칙적으로 분포하는 점들에 대하여 가중치 조합을 미리 계산한다. 각 격자 점들에 대하여 이 점들을 포함하는 삼각형을 찾아내고 삼각형의 꼭지점들을 선형 보간하여 가중치 조합을 계산하여 격자점의 가중치 조합을 얻는다. 실행 시 주어진 점의 가중치 조합은 주변 격자 점들에서 그것들을 쌍 일차 보간(bilinear linear interpolation)하여 효율적으로 계산 할 수 있다.

5. 실행 시 동작 생성

실행 시(run time), 그래프의 간선을 방문하며 각 간선에 해당하는 동작 절편을 연결하여 애니메이션을 만든다. 기존의 그래프를 이용한 동작 합성 방법에서와 같이, 간선에 해당하는 동작 절편들을 자연스럽게 연결하기 위해 노드 주변을 부드럽게 혼합한다. 또한 선택된 집합 간선의 동작 절편들은 사용자의 명령에 의하여 결정되는 가중치들의 조합에 의해 혼합된다. 이 절에서는 주어진 가중치를 이용하여 집합 간선의 동작들을 어떻게 혼합 할 수 있는가에 대해 살펴본다. 그리고 본 연구에서 제안하는 시스템에서 사용자가 그래프를 탐색 이동하며 그 가중치를 제어하는 방법을 설명할 것이다.

5.1 동작 구성

집합 간선의 동작 절편들은 사용자에 의해 결정된 가중치를 적용하여 혼합된다. 이 때 매 순간 마다 시각 정보, 평면 이동, 관절 각 정보를 각각 보간하여 자세를 생성한다. 동작 절편의 길이는 각기 다르고 동작 절편들의 시간적 대응관계가 선형적이지 않기 때문에, 좋은 품질의 애니메이션을 생성하기 위해서는 시간 정보를 고려하여 동작을 혼합해야 한다. 동작을 비교하기 위해 앞서 계산한 동작간의 시간적 대응관계를 이용하여 혼합될 자세를 선택한다. 이렇게 얻어진 대응 자세들은 각각 다른 시작 위치와 방향을 가지고 있기 때문에, 동작 데이터의 포함된 캐릭터 절대 위치와 방향 대신, 캐릭터의 수평 변위와 회전을 혼합해야 한다. 마지막으로 각각의 관절 각을 혼합하여 애니메이션 프레임의 캐릭터 자세를 생성한다.

시각 정보: 각 애니메이션 프레임에서 동작 절편으로부터 대응되는 자세들을 수집해야 한다. 3.2에 언급한바와 같이 기존 연구들 [15, 13, 7] 에서와 같은 특정한 동작절편을 참조 절편으로 선택하고 주어진 동작 절편의 애니메이션 프레임들 간의 대응 관계를 만든다. 참조 절편의 애니메이션 시간을 참조 시간으로 삼고, 참조 시간 T가 주어졌을 때, i번째 동작의 대응은 $M_i\left(F_i(T)\right)$ 이다. 반면에 $M_i(t)$는 시간 t에서 i번째 동작의 자세로 얻을 수 있다. 여기서 $F_i(t)$는 참조 절편으로 부터 i번째 동작 절편으로의 시간 대응 맵이다. 다음 애니메이션 프레임에 해당하는 참조 시간을 얻기 위한 참조 시간의 증가량 ΔT는 동작 절편들 각각이 다음 프레임으로 진행할때 대응되는 T의 증가량을 혼합 함으로써 계산한다.

$$\Delta T = \sum_i w_i(T) \frac{dT}{dF_i} \Delta t, \tag{2}$$

$w_i(T)$'는 현재 가중치 조합이고 Δt는 실제 애니메이션의 프레임 률(frame rate)로 일반적으로 1/30 이다.

위치와 방향성: 캐릭터의 절대 위치와 방향 대신 변위와 회전량을 혼합하면 초기 출발 위치나 방향이 다르지만 유사한 내용을 가진 동작들을 혼합 할 수 있다는 것은 잘 알려

져있다 [15, 14, 7]. 따라서 동작 절편의 각 프레임에 캐릭터의 절대 좌표와 방향 대신 이전 애니메이션 프레임의 수평 위치와 y축으로의 회전량을 저장한다. 즉, 두 연속적인 애니메이션 프레임에서 최상위 절편의 방향 $\mathbf{q}(t)$와 $\mathbf{q}(t+\Delta t)$이 주어지면, 가장 먼저 z축(또는 수평 평면에 놓여지는 임의의 단위 벡터)를 $\mathbf{q}(t)$와 $\mathbf{q}(t+\Delta t)$로 회전시켜 $\mathbf{v}(t)$와 $\mathbf{v}(t+\Delta t)$을 얻는다. θ는 $\mathbf{v}(t)$와 $\mathbf{v}(t+\Delta t)$를 $x-z$ 평면에 투영하고 정규화 한 벡터 사이의 각으로 정의한다. y축에 대한 $\mathbf{q}(t)$에서 $\mathbf{q}(t+\Delta t)$로의 회전을 나타내는 회전 백터는 $\hat{y}$ 로 계산될 수 있다. 이와 같은 회전 벡터들을 혼합하고 이전 프레임의 중심 위치를 $\hat{y}\theta/2$로 회전 시켜 해당 프레임의 방향을 얻을 수 있다.

관절 각: 특정한 관절의 각을 혼합하는 데에, 단위 사원수 공간의 대척점 일치 속성(antipodal equivalence property) 때문에 발생하는 문제를 제거하기 위해 Park 등이 제안한 방법 [14] 을 채택하였다. 이 방법은 주어진 각과 혼합된 값 사이의 차이를 각의 코사인으로 정의하고 차이의 가중합이 최소화되는 단위 사원수를 찾는 방법이다. 이 방법을 이용하면 관절 각을 혼합하는 계산을 매우 효율적으로 수행할 수 있다. 이런 방법으로 얻은 각 관절각들을 캐릭터에 대입하여 최종적인 자세를 계산한다.

5.2 실행 시 제어

실행 시에 사용자는 캐릭터의 행동을 제어하기 위해서 연결된 그래프 노드와 간선을 방문한다. 중추 노드로부터 출발하는 다수의 집합 간선이 있기 때문에, 사용자는 중추 노드로부터 어떠한 간선을 따라가도록 할지에 대한 몇 가지 선택권 갖게 된다. 그 결과, 이를 사용하는 응용 소프트웨어는 키보드나 조이스틱 버튼을 사용하여 사용자가 캐릭터의 행동을 선택 할 수 있도록 해야 한다. 중추 노드로부터 시작하는 집합 간선들은 각각에 해당하는 키와 버튼으로 할당 되고, 키나 버튼을 누름을 통해 원하는 행동을 생성하게 된다.

집합 간선이 선택 되면 간선의 동작 절편들의 가중치 조합은 출력 동작을 생성하는데 이용된다. 이미 두 매개변수 축에 집합 간선의 동작 절편들을 변수화하여 가지고 있기 때문에 마우스나 조이스틱과 같은 두 제어 축을 사용하는 입력 장치들 역시 매개변수화 하여 직접 사용하면 된다. 만약 매개변수 공간이 z축과 거의 수직인 경우에는 매개변수 평면 위에 y축의 투영에 해당하는 첫 번째 매개변수 축은 입력장치의 y축에 대응된다. 다른 매개변수 축은 장치의 x축에 대응된다. 장치로부터 x와 y값이 주어지면, 주변의 네 점에 저장되어 있는 가중치 조합을 2차 선형 보간하여 가중치 조합들을 찾아낸다.

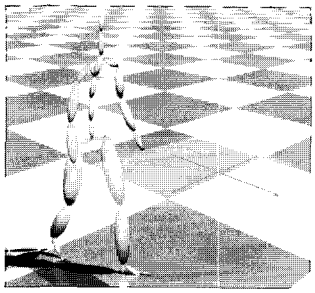
6. 결과

본 연구에서 제안한 방법을 실험하기 위하여 걸어가기, 살금살금 걸어가기, 야구 방망이 휘두르기, 발차기의 네 종류 입력 동작 데이터를 이용하여 저작 시스템을 시험하였다. 모든 입력 동작 데이터는 광학 동작 포착 시스템 Vicon을 사용하여 초당 120 프레임의 비율로 포착하고 실제 애니메이션을 위해 초당 30 프레임의 시간 간격으로 재추출(resampling)하였다.

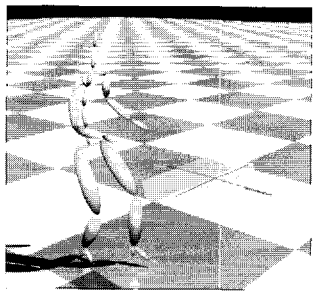
걸어가기: 임의로 걸어가기는 연기자가 임의의 방향으로 걸어가는 동작을 포착하였다. 동작 포착 데이터는 22 걸음이 포함되었다. 저작 시스템은 기본 자세로 하나의 자세를 자동으로 찾아주었고, 다양한 방향으로의 걸음에 해당하는 하나의 집합 간선을 찾아내었다. 그 결과 하나의 중추 노드(hub node)와 하나의 집합 간선을 가지는 그래프를 만들어 내었다. 집합 간선은 21개의 동작 절편을 가지고 있다. 집합 간선은 마지막 프레임의 캐릭터의 위치를 기반으로 간선을 매개변수화(parameterization)하였다. 매개변수 공간은 100개 점을 임의 추출하여 만들어 졌고, 매개변수 축은 앞뒤 방향과 왼쪽/오른쪽 방향과 일치했다(그림 4(a)). 본 연구의 실행 능력을 평가하는 하는 시험 프로그램을 구현하였다. 이 시스템은 사용자가 매개변수 공간의 회전각과 걸음의 폭을 마우스를 사용하여 제어하도록 하였다.

살금살금 걸어가기: 여러 방향으로 살금살금 걸어가는 예제에서는 입력 동작 포착 데이터에 32걸음이 포함되어 있었다. 저작 시스템은 정상적인 걸음을 제외한 22개의 프레임들을 유사 집합으로 하는 중추 노드로 자동으로 제공하였다. 걸어가기와 동일한 방법을 적용하여 매개변수화 작업을 진행하였다(그림 4(b)). 걸어가기에서 사용한 시험 프로그램을 동작의 품질을 확인하기 위하여 동일하게 사용하였다.

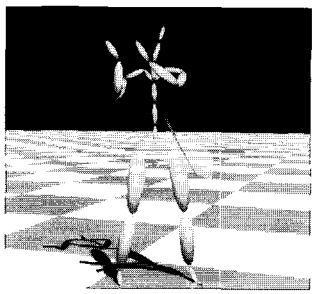
야구 방망이 휘두르기: 세 번째 예제로 야구 방망이를 여러 번 휘두르는 동작을 입력으로 사용하였다. 동작 포착 단계에서 연기자에게 아홉 개의 임의의 목표 지점을 지나가도록 방망이를 휘둘러 달라고 요청하였다. 이 예제에서는 세 개의 집합 간선과 하나의 중추 노드를 갖는 그래프를 만들어 낼 수 있었다. 중추 노드는 방망이를 휘두르기 전 준비 자세 이었고, 간선들은 두 종류의 준비 동작과 휘두르는 동작이었다. 휘두르는 동작은 타격 지점의 오른손을 기반으로 매개변수화 하였다. 만들어진 변수 공간은 스트라이크 존과 유사하였다(그림 4(c)). 실행 단계에서는 준비 동작을



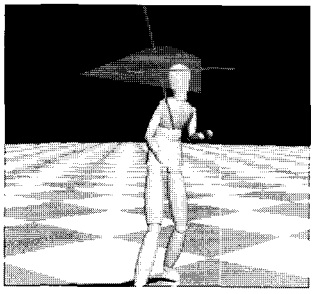
(a) 걸어가기



(b) 살금살금 걸어가기



(c) 야구 방망이 휘두르기



(d) 발차기

그림 4: 매개변수 공간

되풀이하여 재생하고 있다가 마우스 포인터가 위치한 지점을 클릭하면 그 방향을 지나가면서 휘두르는 동작을 만들 수 있었다.

발차기: 마지막 예제에서는 무술 동작 데이터를 사용하여 그래프를 만들어 보았다. 여러 발차기 동작들을 이용하여 왼쪽과 오른쪽 준비 동작에 해당하는 두 개의 중추 노드로 그래프를 만들었다. 매우 다양한 발차기 동작들이 있어서 11개의 집합 간선이 만들어 졌다. 간선은 두 개 정도의 살짝 점프하는 짧은 동작 외에는 모두 왼쪽 오른쪽 준비 자세에서 시작하여 발차기를 하는 동작이었다. 혼합할 동작의 품질을 희생하는 간선 분류 과정을 진행하는 동안 경계값을 증가시켜 집합 간선의 수를 줄일 수 있었다. 가장 큰 집합 간선의 매개변수공간을 그림 4(d)에서 보여주고 있다. 공간상의 선택된 점을 선택하면, 그 지점을 발로 차는 동작을 만들어 낼 수 있다.

7. 결론 및 토의

이 논문에서는 기존 애니메이션을 기반으로 새로운 캐릭터의 애니메이션을 합성하는 방법을 제시했다. Gleicher등 [5]이 제안한 상호 작용적 캐릭터 시스템과는 다르게 본 논문에서는 동작을 부드럽게 연결하기 위하여 동작 절편들을 실행시에 보간한다. 이러한 접근 방법은 기존의 방법에 비해 실행 시 보다 많은 계산을 요구한다. 그러나 이러한 계산 량은 매개변수 제어를 통한 동작 혼합 방법 자체가 요구하는 계산량에 비해 많지 않으며 동작을 연결하는 과정에서의 품질 손실을 최소화 할 수 있다.

이 방법의 문제점은 적용할 수 있는 동작의 범위가 제한적이라는 점이다. 이 방법 그래프 안에 소수의 중추 노드를 만들어야 하기 때문에 유사한 자세를 시작과 끝으로 하는 동작이 많이 있어야 효과적이다. 또한 주어진 동작은 매개변수화 되기 위해 많은 유사한 동작을 가지고 있어야 한다. 그러나 이러한 제약이 비디오 게임과 같은 응용 프로그램을 개발하는데 경우에는 큰 문제가 되지 않을 것이다. 특히 게임에서 캐릭터들은 한정된 동작만을 수행하고, 그 중 대부분의 경우는 시작과 끝 동작이 매우 유사하다. 예를 들어 여러 무술 게임들에서 많은 동작들이 무술의 준비 자세에서 시작해서 준비 자세로 끝나고 대부분의 동작을 몇 개의 묶음으로 구분 할 수 있다. 따라서 게임 등에서와 같이 제한된 동작을 사용하는 응용 분야에는 유용하게 사용될 수 있을 것이다.

이 방법에서 다른 문제점은 매개변수화 과정에서 3차원 관절 위치를 2차원 평면으로 근사 한다는 점이다. 또한 가중치 조합들과 관절 위치의 대응이 국소적으로 선형성을

가진다고 근사 했다. 이와 같은 근사 과정을 통해 만들어진 동작이 사용자가 정해준 지점을 정확하게 보간하지 않는 다는 문제를 야기한다. 실험에서는 많은 점들을 추출하고 촘촘한 격자를 사용하여 오차를 줄일 수 있었다.

참고 문헌

[1] Okan Arikan and D. A. Forsyth. Interactive motion generation from examples. *ACM Transactions on Graphics*, 21(3):483–490, 2002.

[2] Okan Arikan, David A. Forsyth, and James F. O'Brien. Motion synthesis from annotations. *ACM Transactions on Graphics*, 22(3):402–408, 2003.

[3] Min Gyu Choi, Jehee Lee, and Sung Yong Shin. Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Transactions on Graphics*, 22(2):182–203, 2003.

[4] James Davis, Maneesh Agrawala, Erika Chuang, Zoran Popović, and David Salesin. A sketching interface for articulated figure animation. In *Proceedings of the 2003 ACM SIGGRAPH / Eurographics Symposium on Computer animation*, pages 320–328, 2003.

[5] Michael Gleicher, Hyun Joon Shin, Lucas Kovar, and Andrew Jepsen. Snap-together motion: assembling run-time animations. In *Proceedings of 2003 ACM Symposium on Interactive 3D Graphics*, pages 181–188, April 2003.

[6] Tae-hoon Kim, Sang Il Park, and Sung Yong Shin. Rhythmic-motion synthesis based on motion-beat analysis. *ACM Transactions on Graphics*, 22(3):392–401, 2003.

[7] Lucas Kovar and Michael Gleicher. Flexible automatic motion blending with registration curves. In *Proceedings of the 2003 ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 214–224, August 2003.

[8] Lucas Kovar and Michael Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Transactions on Graphics*, 23(3):559–568, 2004.

[9] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. *ACM Transactions on Graphics*, 21(3):473–482, 2002.

[10] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics*, 21(3):491–500, July 2002.

[11] Jehee Lee and Kang Hoon Lee. Precomputing avatar behavior from human motion data. In *Proceedings of the 2004 ACM SIGGRAPH / Eurographics symposium on Computer animation*, pages 79–87, 2004.

[12] Tomohiko Mukai and Shigeru Kuriyama. Geostatistical motion interpolation. *ACM Transactions on Graphics*, 24(3):1062–1070, August 2005.

[13] Sang Il Park, Hyun Joon Shin, Tae-hoon Kim, and Sung Yong Shin. On-line motion blending for real-time locomotion generation. *Computer Animation and Virtual Worlds*, 15(3):125–138, 2004.

[14] Sang Il Park, Hyun Joon Shin, and Sung Yong Shin. On-line locomotion generation based on motion blending. In *Proceedings of ACM SIGGRAPH Symposium on Computer Animation*, pages 105–112, July 2002.

[15] Charles Rose, Michael F. Cohen, and Bobby Bodenheimer. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics & Applications*, 18(5):32–40, September - October 1998.

[16] Matthew Thorne, David Burke, and Michiel van de Panne. Motion doodles: an interface for sketching character motion. *ACM Transactions on Graphics*, 23(3):424–431, 2004.

[17] D.J. Wiley and J. K. Hahn. Interpolation synthesis of articulated figure motion. *IEEE Computer Graphics and Applications*, 17(6):39 –45, November 1997.