음악 속도에 따른 캐릭터의 춤동작 생성 및 제어

Motion Generation and Control of a Character Dancing with Music

김건우, Gunwoo Kim*, 왕연, Yan Wang*, 서혜원, Hyewon Seo*
*충남대학교 컴퓨터 공학과 컴퓨터 그래픽스 및 응용연구실

요약 본 논문에서는 음악 신호로부터 추출한 비트 정보를 사용하여 가상 캐릭터의 움직임을 제어하는 방법에 대해 논한다. 특히 주기를 가지는 반복적인 동작, 그 중에서도 춤동작에 대한 음악 신호와의 동기화 방법을 제안한다. 서버로 구현된 음악 비트 인식기는 입력 음악 신호에 대한 분석 정보를 규칙적으로 출력한다. 동작 클라이언트는 동작 캡쳐를 통해 얻은 동작 데이터를 여러 개의 기본 동작들로 나누고, 사용자가 선택한 새로운 순서대로 기본 동작들을 연결하여 부드럽게 재생한다. 또한 서버에 접속하여 전송 받은 음악의 템포에 맞게 동작데이터를 와핑(warping)하고 음악의 주요 비트 시각에 맞추어 기본 동작들의 재생시작 시간을 동기화한다. 음원에 의한, 즉 박자, 강약, 비트와 같은 기본적인 정보뿐만 아니라 분위기, 박자 변화와 같은 고급 정보에도 동적으로 반응하여 춤을 추는 가상 캐릭터를 개발하는 것이 본 연구의 궁극적인 목표이다.

핵심어: 음악 비트 정보 추출, 동작 연결, 비트 동기화, 가상 캐릭터, 컴퓨터 애니메이션

1. 서론

컴퓨터 그래픽스 분야에서 사람과 비슷한 캐릭터의 동작을 실시간에 제어하는 연구는 많은 관심을 받고 있다. 그러나 음악 신호에 반응하도록 가상 캐릭터의 움직임을 자동으로 제어하는 문제는 상대적으로 최근에 연구되고 있는 분야이다. 특히, 동적으로 음악과 동작을 동기화하는 연구는 컴퓨터 애니메이션과 음악신호 처리에 대한 전문적인 지식을 모두 필요로 한다.

이 논문에서는 '요술집'이라는 가상 환경 내에서[12] 춤을 추는 가상 캐릭터를 소개한다. 사용자들은 가상 환경 내의 물체와 상호작용하고, 음악을 선택하거나 바꾸어 캐릭터가 음악에 따라 춤을 추도록 한다. 본 논문에서는 동작을 캡쳐하고 해석하여, 사람이 음악에 맞춰 춤을 추는 것과 같이 가상 캐릭터를 생성하는 방법을 제안한다. 특히, 음악에 맞춰 춤을 추는 연기자의 동작을 캡쳐하여 획득한 동작 클립들로 동작 데이터베이스를 구축하였다. 데이터베이스를 구성하는 각각의 클립은 캡쳐 시에 사용된 배경 음악의 정보를 내포하고 있다. 사용자가 음악을 선택하면, 음악 비트 인식기(*music beat detector*) 혹은 음악 서버(*music server*)라 불리는 모듈은 실시간으로 오디오 장치의 음악 신호를 해석하고 비트시각을 탐색한다.

동작 클라이언트(*motion client*)는 음악 서버에 접속하여 전달받은 음악의 정보에 따라 데이터베이스에서 동작 클립(motion clip)들을 선택하고 화면에 재생한다. 음악과 동작의 동기화는 동작 클립을 음악의 템포에 맞도록 타임와핑(time warping)한 후, 음악의 비트 시각과 동작 클립의 시작 시간을 동기화하여 해결하였다.

본 논문에서 제안하는 시스템의 한 방식의 기술적인 핵심은 비트 동기화 방법을 통하여 독립적인 음악 비트 인식기와 동작 클라이언트를 성공적으로 연동한 것이다. 또한 각각의 동작 클립들로부터 전이 동작(transition motion)을 자동적으로 제거하는 새로운 방법을 제시하였다(4.2.2 참조).

본 논문의 구성은 다음과 같다. 2절에서는 선행 연구에 대해서 언급한다. 3절에서 음악 서버의 역할을 하는 음악 비트 인식기에 대해서 기술하고. 4절에서는 데이터베이스로부터 음악에 맞는 동작들을 선택하고 재생하는 동작 클라이언트에 대하여 설명한다. 5절에서는 3절과 4절에서 설명한 서버와 클라이언트를 연동하는 과정을 설명한다. 마지막으로 6장에서 결과를 기술한 뒤, 7절에서 결론과 향후 연구에 대해서 언급한다.

2. 선행 연구

2.1 동작과 음악

최근 몇 년 동안 음악과 동작을 동기화 하기 위해 많은 연구가 진행되어 왔다. 본 논문의 연구와 밀접한 관련이 있는 김태훈(T. H. Kim) 등의 연구에서는 캐릭터의 리듬감 있는 동작을 합성하기 위한 기술을 소개하였다[16]. 그들은 동작의 비트를 식별하고, 비슷한 동작들을 그룹들로 클러스터링(clustering) 하였다. 클러스터링된 전체 그룹들은 이동 전이 그래프(movement transition graph)로 나타내어 진다. 그룹들간의 전이 가능성을 예제 동작들로부터 추정하고, 실시간으로 이동 전이 그래프를 순회함으로써 새로운 동작을 생성한다.

로보틱스 분야에서 시노자키(Shinozaki) 등은 배경 음악에 반응하여 춤을 추는 로봇을 제안하였다[29]. 이 논문에서는 로봇의 각 보조전동기(servomotor)의 움직일 수 있는 한계영역을 설정한 후, 동작 캡쳐 데이터와 로봇간의 역학적인 차이를 없애고 로봇의 각도 표현법에 맞추기 위하여 동작 캡쳐 데이터를 교정하였다.

한편, 사용자의 조깅 속도에 맞추어 음악의 속도를 느리거나 빠르게 자동 조절해 주는 mp3 재생기가 상용화되는 등[30] 음악과 동작의 조화를 연구하고 이를 응용하는 노력들이 시도되고 있다.

2.2 동작 데이터베이스와 동작 그래프

가상 캐릭터의 동작을 생성하기 위해 캡쳐한 동작 데이터를 수정하거나 재사용하는 많은 방법들이 연구되어 왔다[2][5][14][28][31]. 따라서, 캐릭터 애니메이션에서 동작 데이터를 색인화 하기 위해 데이터베이스를 사용하는 것이 새로운 기술은 아니다. 동작 데이터의 라벨링을 위해 몇몇 연구자들은 텍스츄얼 라벨링(textual labeling)[1][4][27][28] 방법을 사용하였고, 다른 일부 연구자들은 수학적 계산을 통한 동작의 내부적인 표현 방법을 사용하였다[15][21].

라반노테이션(Labannotation)은 무용에서 춤을 표기하기 위해서 사용되었을 뿐만 아니라 동작 데이터베이스에서 데이터를 색인화 하기 위해서도 연구되어 왔다. 유(Yu) 등은 사용자가 라반노테이션을 입력하여 동작들을 데이터베이스에서 검색할 수 있도록 동작의 속성 중 하나를 라반노테이션 시퀀스로 설정하였다[32]. 본 논문에서는 춤동작들을 데이터베이스로 구축하고 춤동작을 캡쳐할 때 사용한 배경 음악의 정보를 레코드의 속성으로 설정하였다.

컴퓨터 애니메이션 분야에서 많은 연구자들은 동작들간의 전이를 표현하고 비슷한 동작들을 클러스터링하는 것과 같은 보다 높은 수준의 스키마(organization scheme)를 추구해왔다. 코바(Kovar) 등이 동작 그래프를 제안한 이후로[17], 다른 연구자들은 동작 혼합(motion blending)[19]과 동작 전이(motion transition)[26]를 위해 동작 그래프를 더욱 발전시켜왔다.

2.3 동작 데이터의 압축 표현방법

동작을 캡쳐하여 얻은 데이터는 용량이 크고 고차원이기 때문에 이를 직접 이용하는 것은 쉽지 않은 일이다. 이러한 대용량의 데이터에 대해서, 효율적으로 데이터의 크기를 줄일 수 있는 강력한 방법으로 주성분분석(PCA : Principal Component Analysis)[13] 기법이 있다. 이 기법은 컴퓨터 그래픽스의 많은 어플리케이션들[2][9][22] 뿐만 아니라 동작 전처리 과정에서도 많이 사용되어 왔다. 임(I. S. Lim)과 탈만(Thalmann)의 연구[18]와 글라동(Glardon) 등의 연구[10]에서는 PCA를 사용하여 동작 혼합에 사용되는 보간(Interpolation) 함수의 차원을 효과적으로 감소시켰다. 이와 유사하게, 바비치(Barbič) 등은 PCA에 기반한 방법들이 동작데이터를 별개의 동작을 갖는 데이터들로 자동으로 분할하는데 효과적임을 보였다[6]. 위 논문에서는 일정한 수의 주성분(Principal Component)들을 이용하여 재구성된 동작과 본래 동작을 비교하여 오차를 계산하였고, 그 오차의 미분계수가 급격한 변화를 보일 때 동작데이터를 분할하였다. 본 논문 또한 간결하게 표현된 동작 데이터를 획득하기 위해 PCA를 사용하였고, PCA를 통하여 기본 동작(basic motion)들의 전이 여부를 효과적으로 결정하였다.

3. 음악 신호로부터 비트 인식

본 논문에서 소개하는 시스템의 주요 부분들 중 하나가 음악 데이터의 처리 부분이다. 음악 신호로부터 음악의 분위기, 박자 변화와 같은 높은 수준의 정보를 뽑아내는 연구가 활발히 진행되고 있다. 본 논문에서 소개하는 비트 인식기(Beat Detector)[3]는 주로 고토(Goto)의 연구[11]를 기반으로 하고 있다.

실시간으로 재생되는 사운드를 분석하여 비트(beat) 정보를 추출하기 위해서 음악의 박자는 4/4박자이고 템포는 60-120 BPM 이라고 가정하였다. 또한 안정되고 강한 비트로 구성된 음악일수록 비트 인식 성공률이 높아질 것이라고 가정하였고 실제 테스트한 결과 향상되는 것을 확인하였다.

음악신호를 16KHz, 1-channel, 16-bit 형식으로 캡쳐하였으며, 512 sample 단위로 FFT(Fast Fourier Transform)를 실행하여 그림 1에서 보는 바와 같이 7-band Onset-agents를 구성하였다.

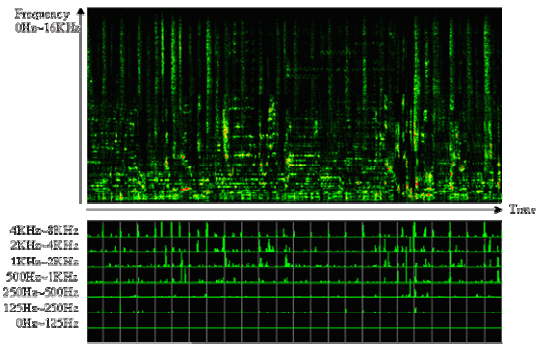

그림 1 Onset-time Vector의 흐름

각 Onset-agents 는 독립적으로 Onset-time을 추출하며, 독립적인 Onset-values 가 시간의 흐름에 따라 관리된다.

상기 과정에서 도출된 7-band Onset-Time Vector중에서 가장 규칙적으로 비트가 반복되는 것을 선택하고 그 결과를 다시 7-band Onset-time Vector 와 Cross-correlation 을 적용하여 현재 입력된 음악신호에 비트가 포함되었는지 여부를 도출한다.

4. 동작 데이터베이스와 동작의 재생

동작과 음악이 서로 밀접하게 연관되어 있는 영역이 바로 춤이다. 또한 춤이라는 것은 본 논문의 어플리케이션에 잘 부합된다. 앞서 언급했듯이, 본 논문에서는 음악에 따라 춤을 추는 댄서의 동작을 캡쳐하여 동작 데이터를 획득하였다. 동작 캡쳐 방식은 역기구학(IK) 등과 같이 계산을 통한 방법에 비해 춤동작과 같은 자유로운 움직임을 보다 쉽게 얻을 수 있는 장점이 있다.

4.1 동작 데이터의 획득

본 논문에서는 광학식 모션캡쳐 장비[24]를 사용하여 댄서의 춤동작을 캡쳐하였다. 비트 성분이 뚜렷하여 에러 없이 비트가 잘 인식되는 음악을 3~4곡 선정하여 모션 캡쳐시 배경음악으로 사용하였다. 모션 캡쳐를 시작하기 전에 댄서가 선정된 곡을 2~3회 정도 듣고 춤동작을 구상할 수 있도록 하였다. 본 논문의 데모 비디오에서 보는 것과 같이, 이러한 동작들은 다소 리드미컬한 느낌으로 음악과 동기화 되어 있다.

4.2 동작 데이터베이스의 구축

본 논문에서는 대용량의 동작 데이터를 효율적으로 관리하기 위해서, DBMS를 사용하였다. 또한 데이터의 재사용성을 극대화 하고 모션 캡쳐의 비용을 줄이기 위해서 기존의 춤동작 데이터를 수십 개의 클립으로 분할하였다. 또한 각각의 클립들에 존재하는 전이 부분을 제거하여 기본 동작으로 재구성하였고, 이들을 데이터베이스로 구축하였다.

4.2.1 동작 데이터의 분할

동작 데이터는 캡쳐시 사용된 배경 음악에 동기화된 비트들로 구성되어 있다. 본 논문에서는 음악의 한 마디에 대응하는 연속된 동작을 기본 동작으로 정의한다. 캡쳐하여 얻은 춤동작은 음악의 각 마디에 동기화된 기본 동작들의 연속이라고 볼 수 있다. 기본동작을 획득하기 위해, 캡쳐시에 사용된 배경 음악의 템포와 동작 파일의 프레임 비율(frame rate)을 이용하여, 음악의 마디 단위로 원형의 동작 데이터를 분할하였다.

그림 2에서, 동작 클립의 각각의 비트는 하나의 타원으로 표현하였다. 그림 2의 (a)는 각각 4개씩의 비트를 포함하고 있는 기본 동작이 세 개로 구성된 본래의 동작 데이터를 의미한다. 그림 2의 (b)는 기존의 모션 데이터를 4개의 비트 단위, 즉 기본 동작으로 자른 것을 나타낸다. 이 기본 동작들을 다른 순서로 연결한 그림이 그림 2의 (c)이다. 이러한 방법으로 동작 클립들을 여러 순서로 결합하여 새로운 동작들을 생성할 수 있다.

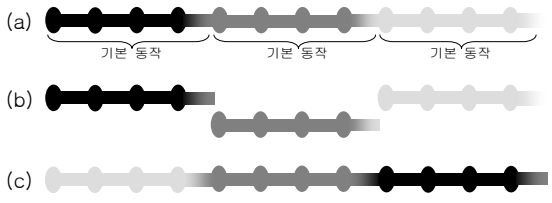

그림 2 (a)는 세 마디로 이루어진 원형의 동작 데이터로써, 각 타원은 배경음악을 통해 얻은 동작의 비트를 나타낸다. (b)는 (a)를 각 마디의 경계에 따라 자른 기본 동작들을 나타낸다. (c)는 (b)의 기본 동작들을 재정렬하고 부드럽게 보간한 동작 데이터를 나타낸다.

4.2.2 전이 동작의 제거

동작 클립들을 재배열 하는 것은 대부분의 클립들이 자신의 앞이나 뒤 부분에 전이 부분(transiting part)을 가지고 있기 때문에 문제가 발생할 수 있다. 전이 동작이란, 한 동작이 다른 동작과 연결되는 부분에 존재하는 중간 동작이다. 본래의 동작과는 다른 순서로 동작클립을 연결할 때에는, 각 클립의 본래의 전이 동작 부분이 새로운 순서의 앞 뒤 동작에 맞도록 바뀌어야 한다. 단적인 예로, 하나의 동작 클립을 연속해서 재생하면 전이 동작에 대해 확인할 수 있다. 이와 같이 반복되는 동작은 전이 부분이 남아있기 때문에 slerp과 같은 보간 기법을 사용하더라도, 실제 댄서가 반복해서 추는 것과 같은 자연스러운 동작을 볼 수 없다. 따라서 본 논문에서는 위와 같은 동작 클립의 연결 부분을 제거하는 전처리 과정을 제시하였다.

본래의 동작 데이터는 차원의 수가 백을 넘기 때문에 직접적으로 전이 부분을 찾는 것은 어려운 일이다. 이를 해결하기 위해 본 논문에서는 PCA기법을 제안한다. PCA기법은 데이터셋(dataset)을 간결화하고 데이터의 패턴을 식별하며[7], 데이터들간의 비교 및 대조 등에 널리 사용되어 왔다[9]. 동작을 연속적인 자세들의 집합이라고 가정하면, 각각의 자세는 n차원의 벡터 $v(\in R^n)$로 표현할 수 있다(차원의 수를 의미하는 n은 캐릭터의 관절 수인 j와 각 관절의 자유도(DOF : Degree of Freedom)를 나타내는 상수 6(translation, rotation)의 곱으로 표현할 수 있다).

$$v = (T_{x1}, T_{y1}, T_{z1}, R_{x1}, R_{y1}, R_{z1}, \cdots, T_{xj}, T_{yj}, T_{zj}, R_{xj}, R_{yj}, R_{zj}) \quad (1)$$

하나의 동작 파일은 $n \times m$ 차원의 행렬로 표현할 수 있다(m : 프레임의 개수). 댄스 동작에서 사람의 팔다리는 대게 동시에 움직이려는 경향이 있고, 각 프레임 사이의 움직임의 변화는 매우 적기 때문에, 행렬 내에는 중복되는 값들이 많다. PCA변환에서 처음 r ($r \ll n$)개의 주성분을 선택하여 동작 데이터를 재구성하면 동작의 세부 정보를 많이 잃지 않으면서 행렬의 크기를 충분히 줄일 수 있다. 세부 절차는 아래와 같다.

1. 동작 파일로부터 $M(n \times m)$의 행렬을 생성한다.
2. 공분산 행렬 $C = MM^T$ 의 고유벡터(eigenvector)a_i를 포함하는 고유공간(eigenspace) A를 계산한다. 그리고 각 고유 벡터의 계수 β_i 로 표현되는 행렬 B를 계산한다.

3. 주성분들 중에서 처음 $r(r<n)$개를 선택한다(처음 r개의 고유벡터들이 가장 큰 고유값들을 갖는다). 동작 데이터는 아래의 식으로 근사하여 표현한다.

$$M=AB=\sum_{i=0}^{n}\alpha_i\beta_i\approx\sum_{i=1}^{r}\alpha_i\beta_i \qquad (2)$$

4. 새로 표현되는 $(\beta_1, \beta_2,\cdots,\beta_r)$로 동작을 분석한다.

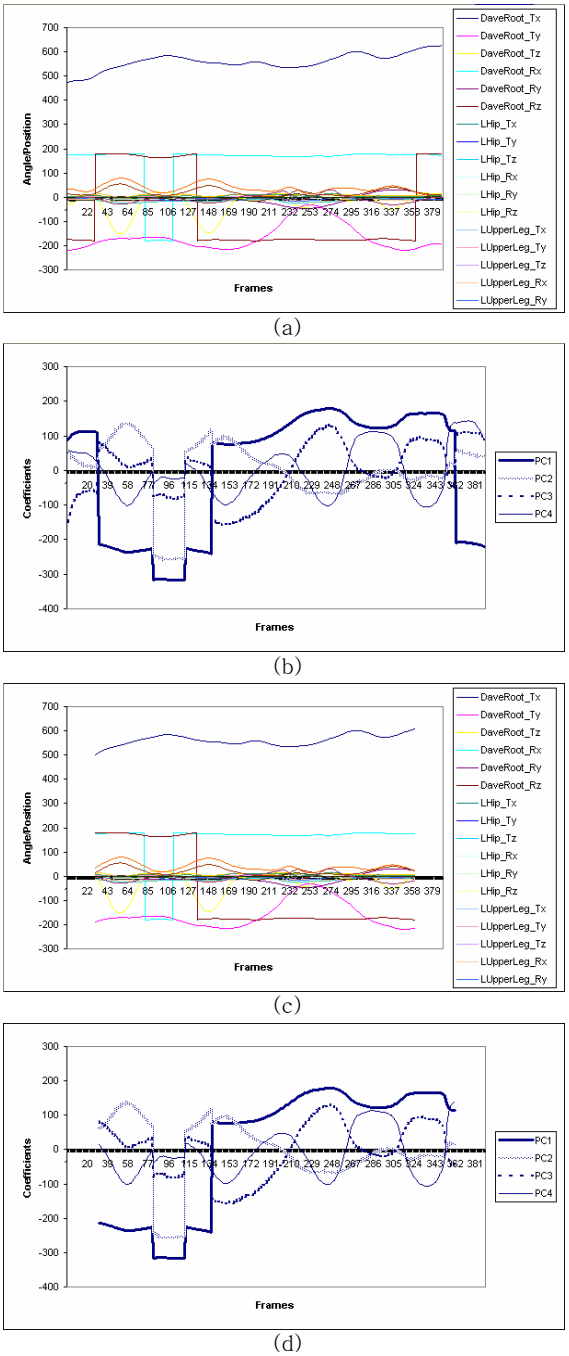

그림 3 (a) 원형 동작 데이터의 루트, 왼쪽 힙, 왼쪽 허벅지의 곡선들, (b) 첫 4개의 주성분(PC)에 대한 4개의 곡선들, (c) (a)의 곡선들에 대한, 전이 부분을 제거한 곡선들, (d) (b)의 곡선들에 대한 전이 부분을 제거한 곡선들

보통은 전이 부분이 각 동작 클립의 앞과 뒷부분에 몇 프레임 되지 않는 짧은 구간이다. 이에 대한 전형적인 예를 그림 3에 보였다. 그림 3의 (a)는 본래의 동작 데이터의 곡선, 그림 3의 (b)는 처음 4개의 고유벡터의 계수들의 곡선을 나타낸다. 이 중에서 주성분 PC1~PC3의 곡선은 30번째 프레임과 362번째 프레임에서 급격한 변화를 보인다. 이 부분이 전이 순간을 나타낸다. 그림 3의 (c)와 (d)는 전이 구간을 제거한 후의 곡선을 나타낸다. 비록 이 방법으로 모든 전이 부분을 찾아내지 못할지라도, 주성분 곡선을 사용하여 전이 구간으로 결정될 가능성이 있는 위치들을 얻을 수 있고, 이것은 본래의 동작 곡선을 사용하는 것 보다 효율적이다.

4.3 동작 연결과 재생(display)

본 논문에서 개발한 동작 재생기는 선택된 동작 클립들로부터 가상 캐릭터를 생성하고, 그 클립들을 부드럽게 연결한다. 동작 클립들은 본래 동작의 순서나 동작 유사도 측정법(motion similarity measure)[20]에 의해 자동적으로 선택될 수 있다. 본 응용프로그램에서는 동작 클립의 종류와 순서를 사용자가 선택할 수 있도록 하였다. 동작 클립을 전달받은 동작 재생기는 동작 클립 파일 계층 정보를 이용하여 장면그래프 내에 segment 계층구조로 캐릭터를 생성한다. 그 다음 과정에서 동작 클립의 기본자세(base pose)정보를 읽어 들여 캐릭터의 각각의 세그먼트에 변환 정보를 설정한다. 이와 비슷한 방법으로 캐릭터의 각각의 세그먼트에 프레임 단위의 변환 정보를 설정하여 애니메이션을 적용한다.

4.3.1 동작 재생(Implementation details)

동작 클립을 읽어 들여 캐릭터의 움직임을 화면에 보여주기 위해, 애니메이션의 주기를 손쉽게 제어할 수 있는 기능과 데이터의 보간 기능이 내장되어 있는 실시간 그래픽스 라이브러리인 OpenGL Performer™[25]를 사용하였다.

키프레임 애니메이션을 효과적으로 제어하기 위해, 다수의 애니메이션 엔진의 재생에 관련된 부분을 조정하는 타임 엔진을 구성하였다. 타임 엔진은 'framerate'와 'framerange'인자 각각에 설정된 값에 따라 애니메이션의 속도와 재생 범위를 제어한다. 애니메이션 엔진은 각각의 시간 단위(프레임 단위)에 따른 보간된 변환 정보를 출력한다. 그림 4에 표현되었듯이, 캐릭터의 각각의 세그먼트는 애니메이션 엔진과 연결되어 있다. 애니메이션 엔진의 소스(source)인 변환 정보는 기본 자세의 변환 정보와 각 프레임단위의 변환 정보가 결합된 벡터 배열로 구성되어 있다. 애니메이션 엔진의 출력인 'flux data(pfFlux)'는 장면그래프 내에서 캐릭터의 각각의 세그먼트에 연결되어 있는 변환 노드의 값을 설정하는데 사용한다.

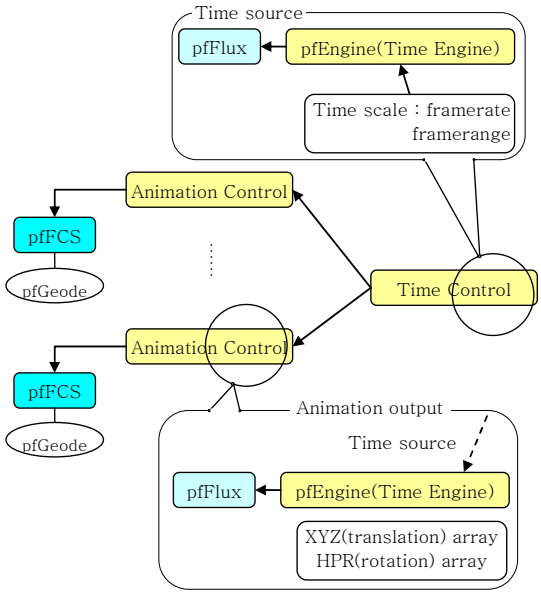

그림 4 pfEngine를 이용한 장면그래프의 구성

4.3.2 동작 연결

춤추는 가상 캐릭터의 이동 경로 제어(global path control)는 본 논문의 연구 범위를 벗어난다. 그 대신 본 논문에서는 동작 클립에 포함된 루트 세그먼트의 변환정보를 그대로 사용하되, 클립들을 연결할 때 발생하는 비연속성 문제에 초점을 맞추었다. 캐릭터의 루트 세그먼트의 위치 정보가 각각의 동작 클립에 저장되어 있기 때문에, 애니메이션 중 클립이 바뀌는 순간에 캐릭터의 갑작스런 튐 현상이 발생한다. 이 문제를 해결하기 위하여, 루트 세그먼트의 위치 정보가 동작 클립 단위로 저장되어 있는 루트 세그먼트의 위치 값에 구속 받지 않도록 하였다. 우선 첫 번째 클립의 첫 프레임을 제외한 모든 동작 클립 내의 각 프레임의 위치 값을 이전 프레임의 위치 값과의 차이로 대체한 후, 클립이 바뀌는 순간의 뒤 클립의 루트 세그먼트의 초기 위치를 앞 클립의 루트 세그먼트의 마지막 위치로 대체하여 해결하였다. 이러한 방법으로 뒤따라 오는 동작은 그 전 동작이 끝나는 위치에서 시작하게 되고, 그 결과 캐릭터의 위치 변환이 부드럽게 연결된다. 루트 세그먼트를 제외한 나머지 관절각들의 연결을 위하여, slerp을 사용하여 얻은 보간된 프레임들로 클립의 시작과 끝부분에 존재하는 갭을 메웠다(동작 클립 대부분은 4.2.2 과정에서 거친 전이 동작(transition movements)의 제거로 인하여 갭이 존재한다). 수정된 동작 클립에 갭이 없을 때에는, 클립의 시작과 끝부분의 5%에 해당하는 프레임들을 slerp을 이용한 연결(보간) 동작으로 대체하였다.

5. 통합

본 절에서는 3절에서 기술한 음악 비트 인식기와 4절에서 기술한 동작 매니저(Motion Manager)를 동기화 하고 통합하는 방법을 기술한다. 이들은 TCP/IP 기반의 서버-클라이언트 구조로 구현되었으며, 시스템의 개관은 그림 5에 나타나 있다. 사용자가 음악을 선택한 후, 음악 서버는 오디오 신호를 해석하기 시작하고 메시지를 생성한다. 서버에 연결되어 있는 클라이언트인 동작 매니저는 음악의 정보, 구체적으로 비트와 템포, 박자(time signature)가 포함된 메시지를 받는다.

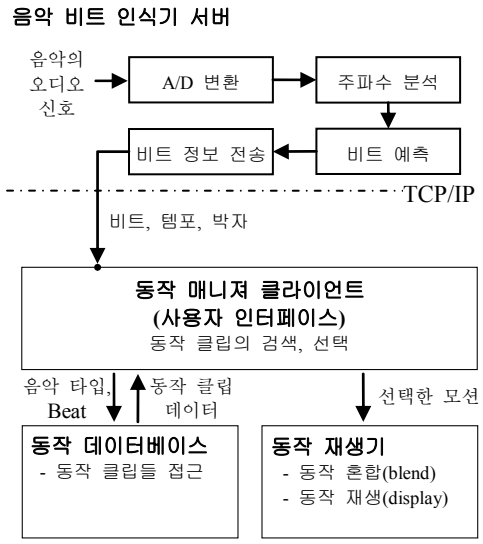

그림 5 시스템의 개관

5.1 음악과 동작의 비트 동기화

음악에 동기화된 동작을 재생하기 위해서, 우선 음악과 동작 클립의 템포를 비교해서 동작의 새로운 템포를 계산한다. 앞에서 언급하였듯이, 춤동작을 캡쳐하여 동작 데이터를 획득하였고, 이 데이터를 잘라서 만든 모든 클립들의 템포는 캡쳐 당시에 사용된 배경 음악의 템포와 동일하다고 가정하였다. 이제 음악에 대한 동작의 타임 와핑은 동작의 프레임 비율을 수정함으로써 간단하게 구현할 수 있다. 즉, 두 템포 사이의 비율로부터 다음의 식을 통해 동작 클립의 새로운 프레임 비율을 계산하였다.

$$F = (F_1 \times T_2) / T_1 \qquad (3)$$

T_1 과 F_1 은 각각 동작 본래의 템포와 프레임 비율이고, T_2 는 재생되고 있는 음악의 템포이다.

그림 6은 비트 동기화 과정의 전체적인 단계와 흐름을 나타낸 것이다. 동작 재생기는 동작 클립 데이터를 읽어 들여 캐릭터를 생성하고, 서버로부터 음악에 대한 메시지를 기다린다. 메시지를 받으면 템포 정보를 이용하여 동작의 프레임 비율을 계산한다. 계산이 끝난 후 음악의 비트 시각이 포함된 메시지를 받으면 캐릭터의 애니메이션을 시작한다. 사용자가 음악을 정지(stop)시키거나 일시 정지(pause)시키면 음악 서버에서 클라이언트로 idle 메시지를 보내고, 클라이언트에서는 비트 메시지를 받지 못한 시간이 일정 시간 τ를 넘으면 캐릭터의 동작을 일시 정지시킨다(τ를 조절하여 딜레이 시간을 줄일 수 있다). 클라이언트는 다음 비트 메시지를 받을 때까지 대기상태가 되고, 비트 메시지를 받으면 동작을 재시작한다.

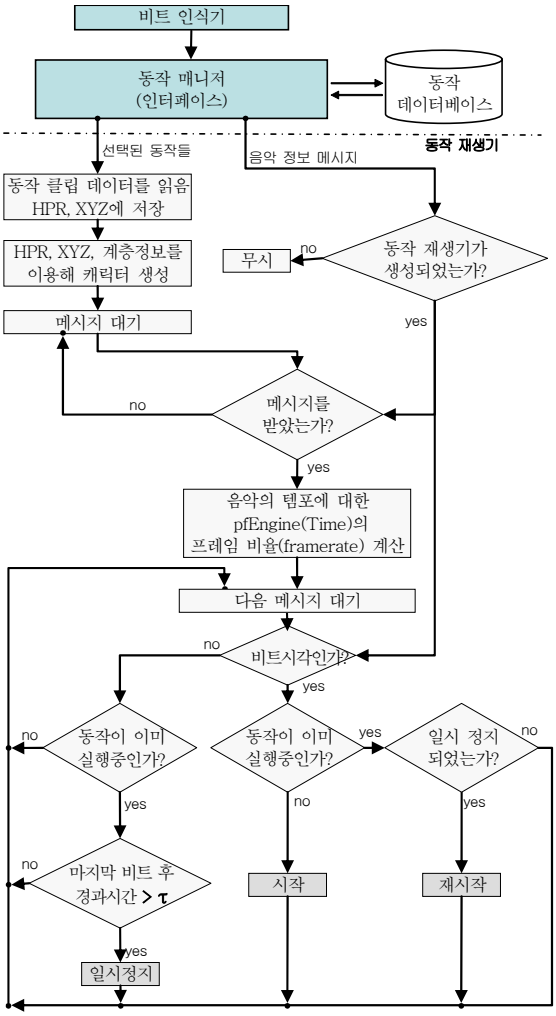

그림 6 음악신호에 대한 실시간 동작 동기화 단계

6. 결과

표 1 테스트에 사용된 대중 음악들의 정보

번호	노래 제목	평균 bpm	노래 길이 (분:초)	가사
1	Hey, oh	98	3:35	프랑스어
2	Childhood	116	3:51	중국어
3	Oxygen	136	4:46	가사 없음

제안된 시스템은 Intel Pentium IV 3.0GHz CPU에 1GB RAM, 128MB Radeon 계열의 그래픽카드를 설치한 PC환경에서 테스트 하였고, Visual C++ 6.0, OpenGL Performer 환경에서 개발되었다. 서버와 클라이언트는 각각이 다른 PC에서 메시지를 주고받으며 실행될 수 있지만, 데모 작업의 편의를 위하여 한 대의 컴퓨터 내에서 실행하였다. 테스트에 사용한 음악의 목록은 표 1에 기술하였다. 테스트의 모든 경우에 입력으로 선택된 음악 속도와 비트에 맞추어 동작 비트 시각이 동기화된 춤추는 캐릭터를 생성하여 보였다.

동작 데이터베이스는 Microsoft Access를 사용하여 구축하였고, ADO[23] 방식으로 데이터베이스에 접근하였다. 그림 7(a)에서 볼 수 있는 레코드들은 기본 동작(동작 클립)들을 나타내고, 동작 클립의 일련번호, 파일 이름, 캡처할 당시 사용된 배경음악의 bpm(beats per minute)값을 속성으로 가지고 있다. 사용자는 그림 7(a)의 인터페이스를 통하여 동작의 bpm 값으로 동작 레코드(motion record)를 검색하여 임의로 선택할 수 있고, 그림 7(b)의 음악 서버에 접속하여 음악 정보 메시지를 받을 수 있다. 선택된 동작 클립들과 전송 받은 메시지는 그림 7(c)의 동작재생기에 전달되어 기본 동작들을 연결하여 재생하고 비트 메시지를 받으면 스포트라이트를 켰다 끔으로써, 비트 시각인 것을 쉽게 확인하도록 하였다.

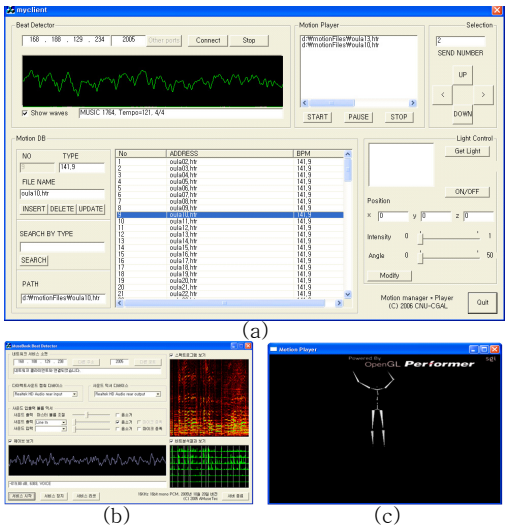

그림 7 (a) 사용자 인터페이스, (b) 비트 인식기, (c) 동작 재생기,

7. 결론 및 향후 연구

본 논문에서는 여러 개의 동작 클립을 사용하여 음악에 동기화하여 다양한 춤동작을 재생하는 가상 캐릭터를 소개하였다. 본 제안한 방법은 캡처를 통해 얻은 춤동작을 자른 후 전처리 과정을 거쳐서 얻은 동작 클립들로 구성된 데이터베이스에 기반한다. 각각의 동작 클립으로부터 전이 동작을 식별 및 제거하여, 임의의 순서로 동작 클립들을 결합하더라도 자연스럽게 보이도록 하였다. 동작 매니저는 실행 중 캐릭터의 동작 비트가 음악의 비트와 동기화 되도록 하기 위해서 동작 클립들의 재생률을 재설정 하고 음악 비트 시각과 동작 시작 시간을 맞추었다. 대중 음악 여러 곡을 사용하여 시스템을 테스트한 결과 잘 동작하였으며, 데모 비디오를 통하여 확인할 수 있다. 향후 계획으로 동작과 음악의 비트를 정확하게 맞추어 이것들을 보다 일치되도록 동기화 하는 방법을 연구할 것이다. 또한, 본 논문에서는 사용자가 직접 동작 클립을 선택하도록 하였으나 이를 시스템이 자동으로 선택하도록 할 계획이다. 나비나 물고기처럼 사람을 닮지 않은 캐릭터도 음악에 맞춰 춤을 추는 기술을 개발하는 것이 최종 목표이다.

감사의 글

모션 캡쳐 데이터의 후처리와 데모 비디오 제작을 도와준 임진우 군(충남대 컴퓨터 전공)에게 감사의 글을 전한다.

본 연구는 KIST 실감 공간 기술 개발 중점 연구의 세부 과제인 “music-to-motion generation of artificial life-forms” 의 지원으로 수행되었다. 본 연구를 위해 조언과 격려를 아끼지 않은 박상호 교수님(충남대 기계설계학과), 정도일 사장님(AMuseTec), 고희동 박사님, 강성철 박사님, 하성도 박사님(KIST)께 감사드린다.

참고문헌

1. Akanksha, Z. Huang, B. Prabhakaran, C. R. Ruiz. “Reusing motions and models in animations” In: J. A. Jorge, N. M. Correira, H. Jones, and M. B. Kamegai (eds). Multimedia 2001, Wien, (2001)21-32.
2. M. Alexa, W. Muller. “Representing animations by principal components” In Proceedings of EUROGRAPHICS 2000, (2000).
3. AMuseTec™, http://www.musebook.co.kr
4. O. Arikan, D. A. Forsyth, J. F. O'Brien. “Motion synthesis from annotations” ACM Transactions on Graphics (TOG), 22(3), (2003)402-408.
5. Y. Ayadin, H. Takahashi and M. Nakajima. “Database Guided Animation of Grasp Movement for Virtual Actors” In Proceedings of Multimedia Modeling '97. (1997) 213-225.
6. J. Barbič, A. Safonova, J.-Y. Pan, C. Faloutsos, J. K. Hodgins, N. S. Pollard. “Segmenting motion capture data into distinct behaviors” In GI '04: Proceedings of the 2004 conference on Graphics interface, Canadian Human-Computer Communications Society, (2004)185-194.
7. P. Belhumeur, J. Hespanha, D. Kriegman. “Eigenfaces vs. fisherfaces: Recognition using class specificlinear projection” In Proceedings of IEEE Transactions on Pattern Analysis and Machine Intelligence, (1997)771 - 720.
8. Demo video. http://cgal.cnu.ac.kr/~data/demo_Motion_Control.avi
9. H. Gao, J. W. Davis. “Recognizing human action efforts: An adaptive three-mode PCA framework” In Proceedings of IEEE International Conference on Computer Vision, (2003)1463-1469.
10. P. Glardon, R. Boulic, D. Thalmann. “PCA-based walking engine using motion capture data” In Computer Graphics International (2004) 292-298.
11. M. Goto. “An Audio-based Real-time Beat Tracking System for Music With or Without Drum-sounds” In Journal of New Music Research, Vol.30, No.2 (2001) 159-171
12. S. Irawati, D. Calderón, H. -D. Ko. “Using Spatial Ontology in the Semantic Integration of Multimodal Object Manipulation in Virtual Reality” Proc. 한국 HCI 학회
13. I. T. Jolliffe. “Principal Component Analysis. Springer series in statistics” Springer-Verlag, (1986).
14. K. Kakizaki. “Generating the Animation of a 3D Agent from Explanatory Text” Proc. ACM Multimedia '98. (1998) 139-144.
15. E. Keogh, T. Palpanas, V. B. Zordan, D. Gunopulos, M. Cardle. “Indexing large human-motion databases” In Proceedings of the 30th VLDB Conference, Toronto, Canada, (2004) 780-791.
16. T Kim., S. Park, S. Shin “Rhythmic-Motion Synthesis Based on Motion-Beat Analysis” pp. 392-401, SIGGRAPH 2003,(2003).
17. L. Kovar, M. Gleicher, F. Pighin. “Motion Graphs” In Proceedings of ACM SIGGRAPH 2002, (2002).
18. I. S. Lim, D. Thalmann. “Construction of animation models out of captured data” In Proceedings of the IEEE Conference of Multimedia and Expo, August, (2002).
19. J. H. Lee, K. H. Lee. “Precomputing Avatar Behavior from Human Motion Data” In Symposium on Computer Animation 2004, pages 79 - 87, August 2004.
20. C. Li, B. Prabhakaran. “A Similarity Measure for Motion Stream Segmentation and Recognition” Proceedings of the Sixth International Workshop on Multimedia Data Mining (MDM/KDD), Chicago, IL USA, pp. 89-94, August, (2005).
21. G. Liu, J. Zhang, W. Wang, L. Mcmillan. “A system for analyzing and indexing human-motion databases” In Proc. ACM SIGMOD (2005)924-926.
22. M. Matarič, A. Fod, O. Jenkins. “Automated derivation of primitives for movement classification” In Autonomous Robots, 12(1), (2002)39- 54.
23. Microsoft® ActiveX® Data Objects, http://msdn.microsoft.com/data/learning/adonet.
24. Motion Analysis™, http://motionanalysis.com
25. OpenGL Performer™, http://www.sgi.com/products/software/performer/
26. S. I. Park, H. J. Shin, T. H. Kim, S. Y. Shin. “On-line motion blending for real-time locomotion generation” In Computer Animation and Virtual Worlds, 15(3), (2004) 125-138.
27. D. Ramanan, D. A. Forsyth. “Automatic annotation of everyday movements” In NIPS 16, (2003).
28. C. Rose, M. F. Cohen, B. Bodenheimer. “Verbs and adverbs: multi-dimensional motion interpolation” In IEEE Computer Graphics and Applications, (1998) 32-41.
29. K. Shinozaki, Y. Oda, S. Tsuda, R. Nakatsu, A. Iwatani. “Study of Dance Entertainment Using

Robots" In Proceedings of Edutainment, (2006) 473-483.

30. Stepman, http://www.inigraphics.net/press/topics/2004/issue1/1_04a07.pdf
31. D. Thalmann, N. Farenc, and R. Boulic. "Virtual Human Life Simulation and Database: Why and How" Proc. International Symposium on Database Applications in Non-Traditional Environments (DANTE'99). IEEE CS Press, (1999).
32. T. Yu, X. Shen, Q. Li, Weidong Geng. "Motion retrieval based on movement notation language" Computer Animation and Virtual Worlds Volume 16, Issue 3-4, (2005) 273 - 282.