미디어학 석사학위 논문

춤 동작을 위한 모션 그래프 생성 및 음악 동기화

아주대학교 대학원

미디어학과

최종원

춤 동작을 위한 모션 그래프 생성 및 음악 동기화

지도교수 신 현 준

이 논문을 미디어학 석사학위 논문으로 제출함.

2024 년 02 월

아 주 대 학 교 대 학 원

미디어학과

최 종 원

최 종 원의 미디어학 석사학위 논문을
인준함.

심사위원장 신 현 준 인

심 사 위 원 경 민 호 인

심 사 위 원 주 은 정 인

아 주 대 학 교 대 학 원

2023 년 12 월 20 일

요 약

본 논문은 춤 동작의 박자를 분석해 마디 단위로 분할한 후 음악에 맞추어 동작 데이터를 합성하는 방법을 제안한다. 마디 단위로 분할 하기 위해서는 먼저 동작의 비트를 찾아내는 것이 중요하다. 이를 위해 관절들의 속도를 이용하여 비트를 찾는 기존의 방법들을 차용하였으며 관절과 동작의 평균 운동량을 사용해 조건을 추가한다. 이렇게 찾아낸 비트들에 대해 고속 푸리에 변환을 사용하여 가장 우세한 주기를 찾는다. 주기를 이용해 조건을 만족하는 비트들을 묶어 마디로 설정한다. 이렇게 추출한 마디들을 적절하게 합성한다. 일반적으로 여러 개의 동작 데이터를 합성하여 새로운 동작을 생성하는 방법으로 모션 그래프가 사용된다. 하지만 본 논문의 방법에 그대로 사용하기에는 어려움이 있다. 따라서 본 논문은 모션 그래프의 그래프 생성 과정을 변형하여 춤 동작을 위한 모션 그래프를 생성한다. 이렇게 생성된 모션 그래프에서 동작을 합성할 때 기존의 방법은 연결되는 동작의 앞뒤에 추가 프레임을 설정한다. 하지만 이러한 방식을 마디 단위로 나누어진 춤 동작에 적용하면, 마디의 첫 박과 마지막 박이 블렌딩 되어 리듬감을 느끼기 어려운 상황이 발생할 수 있다. 따라서 본 논문은 비트의 느낌을 살리면서 동작을 합성하기 위해 기존의 블렌딩 방식을 변형하여 새로운 방법을 제시한다.

차 례

그림 차례

제 1 장 서론

본 논문은 음악과 동기화된 춤 동작을 생성하는 방법을 제안한다. 동작을 생성하는 방법으로는 모션 그래프와[[1],[2]] 생성형 모델이[[3],[4]] 있다. 모션 그래프는 모션 캡쳐된 데이터를 그래프의 형태로 만들고 그래프에 저장된 작은 동작들을 연결하여 새로운 동작을 생성한다. 모션 캡쳐된 데이터와 같은 양질의 데이터를 사용하여 연결하기 때문에 결과에 대한 품질을 보장할 수 있다는 장점이 있다. 하지만 이 방법은 리듬감을 고려하여 그래프를 생성하지 않기 때문에 춤 동작을 생성하는 데는 어려움이 있다. 이를 해결하기 위해 비트를 찾아 해 리듬감을 갖는 춤 동작을 생성하는 방법들도 연구되었으며, 먼저 생성형 모델을 사용한 방법이 있다[[3],[4]]. 이는 모션 캡쳐 데이터와 같은 양질의 동작 데이터 없이도 다양한 동작들을 생성할 수 있다. 하지만 학습을 위해서 방대한 양의 동작 데이터가 필요하며, 생성된 동작들이 모션 캡쳐를 통한 데이터만큼의 품질을 보장하기 어렵다. 인공지능과 모션 그래프를 적절히 활용한 방법도[2] 있다. 해당 방법은 모션 캡쳐된 춤 동작들에 대해 모션 그래프를 생성하고, 음악과 춤의 스타일과 리듬을 학습하여 모션 그래프 탐색의 가중치로 사용한다. 이러한 방법은 양질의 모션 캡쳐 데이터를 합성하여 사용하였기 때문에 품질을 보장할 수 있다. 그러나 이 방법도 좋은 품질을 위해선 방대한 양의 학습을 위한 데이터가 필요하다. 따라서 본 논문에서는 모션 그래프의 구조를 활용해 생성된 데이터의 품질을 보장하면서, 관절들의 구조적인 특성을 통해 안무의 마디를 추출하여 음악과 동기화된 안무 데이터를 생성 방법을 제시한다. 이는 다른 방법들과 비교했을 때 합리적인 품질을 보장하며, 인공지능을 사용하지 않기 때문에 데이터에 대한 의존을 줄일 수 있다.

일상의 동작들과는 다르게 춤 동작은 음악의 분당 비트 수(Beats Per Minute), 박자 등을 고려한다. 특히 음악의 비트에 맞추어 동작의 변화를 갖는 특성이 있으며, 박자에 따라 이러한 동작들을 연결해 하나의 마디를 표현한다. 마디는 안무에서 의미를 갖는

기본이 동작이 된다고 볼 수 있다. 이러한 관점에서 춤 동작은 마디 단위의 합성이 자연스러운 결과를 갖는다.

춤 동작 데이터를 마디 단위로 나누기 위해서는 동작에서의 비트를 찾아야 한다. 이런 문제를 해결하기 위해 기존에 여러 연구가 제시되었다[[5],[6]]. 제시된 방법들은 각 관절 속도의 크기를 계산하여 속도가 줄어드는 지점을 비트로 처리한다. 하지만 속도만을 사용하여 춤 동작의 비트를 검출하기에는 어려움이 있다. 우선 실제 춤 동작에서는 신체의 흔들림으로 인해 비트가 아니어도 관절의 속도가 줄어들 수 있다. 이를 고려하여 작은 움직임을 비트 탐색에서 제외하더라도. 춤 동작의 종류마다 속도와 동작의 크기 달라 신체 흔들림과 비트의 구분이 어렵다. 이러한 이유로 움직임이 다양하고 복잡하게 조합된 춤 동작이라는 주제에서는 더 많은 조건을 고려해야 정확한 비트를 검출할 수 있다. 이런 문제를 해결하기 위해 본 논문에서는 춤 동작들의 평균 운동량을 계산하여 비트들이 검출되는데 조건으로 사용한다.

마디 단위로 동작 데이터를 분해하기 위해서는 비트의 주기를 검출해야 한다. 이는 음악에 맞춰 창작되는 안무라는 개념의 특성상 음악의 박자와 분당 비트 수에 맞춰서 구성되어야 한다. 따라서 음악처럼 비트들의 간격이 같이 유사한 특징을 이용해 본 논문에서는 앞서 검출한 비트들에 대해 고속 푸리에 변환을 진행하여 주된 주파수를 찾아내 비트들의 평균 주기를 얻는다. 이렇게 얻은 평균 주기를 기반으로 동작 데이터에서 마디를 추출한다.

마지막으로 음악의 분당 비트 수를 입력받으면 추출한 마디들에 대해 음악과 동기화된 동작을 생성한다. 실제 춤 동작들의 빠르기는 다양하고, 같은 안무에서도 춤 동작의 속도가 상이 할 수 있다. 만약 춤 동작 데이터가 많다면 시간 변형 없이 입력 음악의 마디 길이와 비슷한 춤 동작 마디들을 사용할 수 있다. 하지만 본 논문에서는 시간 변형을 통해 음악 마디와 동작 마디가 시간상으로 일치되도록 한다. 이러한 방법은 데이터가 적은 상황에서도 다양한 속도를 가진 음악에 동기화된 춤 동작을 생성할 수

있다.

본 논문은 다음과 같은 순서로 구성된다. 제 2 장 에서는 비트 탐색과 모션 그래프 생성, 모션 블렌딩에 관한 관련 연구들을 소개한다. 제 3 장 에서는 본 논문에서 제안하는 동작 비트 탐색 대한 내용을 서술한다. 제 4 장 에서는 춤 동작을 위한 모션 그래프 생성 방법에 대한 내용을 기술한다. 제 5 장 에서는 비트를 고려한 모션 블렌딩에 대한 내용을 소개한다. 제 6 장 에서는 제안한 방법의 실험 결과에 대해 서술한다. 제 7 장 에서는 결론을 맺고 한계 및 향후 방향에 대해 서술한다.

제 2 장 관련 연구

제 1 절 비트 탐색 및 마디 추출

춤 동작을 합성하여 음악과 동기화된 안무 데이터를 생성하기 위해서는 동작에 대한 분석이 필요하다. 일반적으로 음악에 맞는 동작을 생성하기 위해서는 음악의 비트와 춤 동작의 비트를 동기화시키는 것이 중요하다. 음악의 비트는 악보 정보나 MFCC(Mel-Frequency Cepstral Coefficient)를 적용해 소리의 신호를 분석하여 상대적으로 쉽게 추출이 가능하다[7]. 하지만 춤 동작의 경우 수많은 관절의 3차원 정보를 이용하여 비트를 추출해야 하므로 어려움이 있다. 이런 어려움을 극복하기 위해 동작으로부터 비트를 추출하는 방법에 대한 연구들이 제시되어 왔다[5][6].

[5]에서는 각 관절에 대해 가속도가 양수에서 음수 혹은 음수에 양수가 되는(Zero-Crossing) 순간을 비트의 후보로 선정한다. 이러한 방법은 모든 후보 비트를 추출하는데 장점이 있으며, 춤 동작에 국한되지 않고, 모든 동작들의 리듬을 찾을 수 있다. 춤 동작에 더 집중한 방법으로는 [6]에서 제안한 방법이 있다. 해당 방법에서는 속도가 빠르게 감소했다가 다시 빠르게 증가하는 순간이 춤에서 강조되는 동작이라고 가정한다. 이를 계산하기 위해 관절들의 속도가 감소해 0에 가까워지고 다시 속도가 증가하는 부분을 확인한다. 본 논문에서는 [6]에서 제안한 방법을 사용하여 후보 비트를 구한다.

후보 비트를 구한 후 마디를 검출하기 위해 해당 방법[5]에서 참조 비트를 얻는 방식을 활용하였다. 우선 검출한 비트들을 정현파의 형태로 만든다. 이를 사용하여 고속 푸리에 변환을 통해 주파수 영역의 PSD(Power Spectral Density)를 구한다. 가장 진폭이 큰 주파수를 찾은 후 이에 해당하는 주기를 구해 평균 주기로 설정한다. 평균 주기와 유사한 간격을 가진 비트들을 찾아 마디를 추출한다. 해당 방법에 대해서는 제 3 장에서 자세히 소개한다.

제 2 절 모션 그래프 생성

전통적으로 여러 동작 데이터를 합성하여 새로운 동작을 생성하는 방법으로 모션 그래프가[1] 사용된다. 해당 방법은 여러 동작 데이터에 대해 동작 데이터들을 노드로 만들고 연결될 수 있는 동작들을 엣지로 만들어 그래프의 구조를 갖는다. 이 그래프를 탐색하며 각각의 노드들을 합성해 자연스러운 동작 데이터를 생성할 수 있다. 이러한 방법은 최근까지 여러 연구에서 사용된다[5][8][2].

모션 그래프는 일반적인 동작들을 합성해 새로운 모션 데이터를 생성하는데 범용적이고 효율적인 방법이다. 본 논문에서는 리듬감과 연속성을 보존하는 최소한의 단위로 춤 동작을 나누기 위해 동작을 마디 단위로 나누었다. 이때 나누어진 마디 단위로 그래프의 노드를 설정하면 마디가 끝난 프레임에서 시작되는 프레임으로만 동작을 연결할 수 있다. 따라서 기존 모션 그래프의 거리 지도에서 지역 최솟값을 찾아 그래프를 생성하면 본 논문의 접근에서는 거리가 매우 가까움에도 지역 최솟값에 해당하지 않아 마디가 연결될 수 없는 경우가 발생한다. 이러한 문제를 해결하기 위해 본 논문에서는 마디 단위로 그래프를 생성하여 연결한다. 해당 문제의 해결 방법은 제 4 장에서 자세히 소개한다.

제 3 절 동작 블렌딩

동작들에 대한 블렌딩은 일반적으로 시작과 끝에 프레임을 추가하고, 특정 크기의 프레임만큼 관절들의 위치 좌표와 회전에 대해 보간한다. 이는 동작을 합성하는데 기본이 되는 중요한 과정이며, 여러 연구에서 이러한 방법이 사용된다[8][2][9][10][1]. 충분히 비슷한 동작들에 대해 이 방법으로 블렌딩을 진행하면 자연스러운 합성이 가능하다. 또 하나의 연속적인 동작 데이터를 나누어 블렌딩을 진행하더라도 원본 데이터를 보존한다. 이러한 장점들을 이유로 모션 그래프에서 동작들의 합성에도 사용되며, 모션 그래프를 활용하는 여러 연구에도 이 방법이 사용된다[8][2][1]. 하지만 본 논문에서

생성하는 모션 그래프의 노드 특성과 맞지 않아 해당 방법의 사용에 어려움이 있다.

한 마디에 4개의 비트를 갖는 4분의 4박자 음원에 맞춰 동작의 마디를 추출한다고 가정한다. 이때 마디의 시작과 끝 프레임은 첫 번째와 네 번째 비트에 해당한다. 일반적인 블렌딩으로 시작과 끝 프레임에 보간을 위한 프레임을 추가하면, 블렌딩 과정에 의해 마디의 처음과 마지막 비트의 느낌이 퇴색된다. 따라서 본 논문에서는 마디의 시작 부분에만 기존의 추가 프레임을 적용하고, 앞 동작의 마지막 비트에 대해서 보간을 진행하여 비트를 보존한다. 해당 내용에 대한 자세한 설명은 제 5 장에서 소개한다.

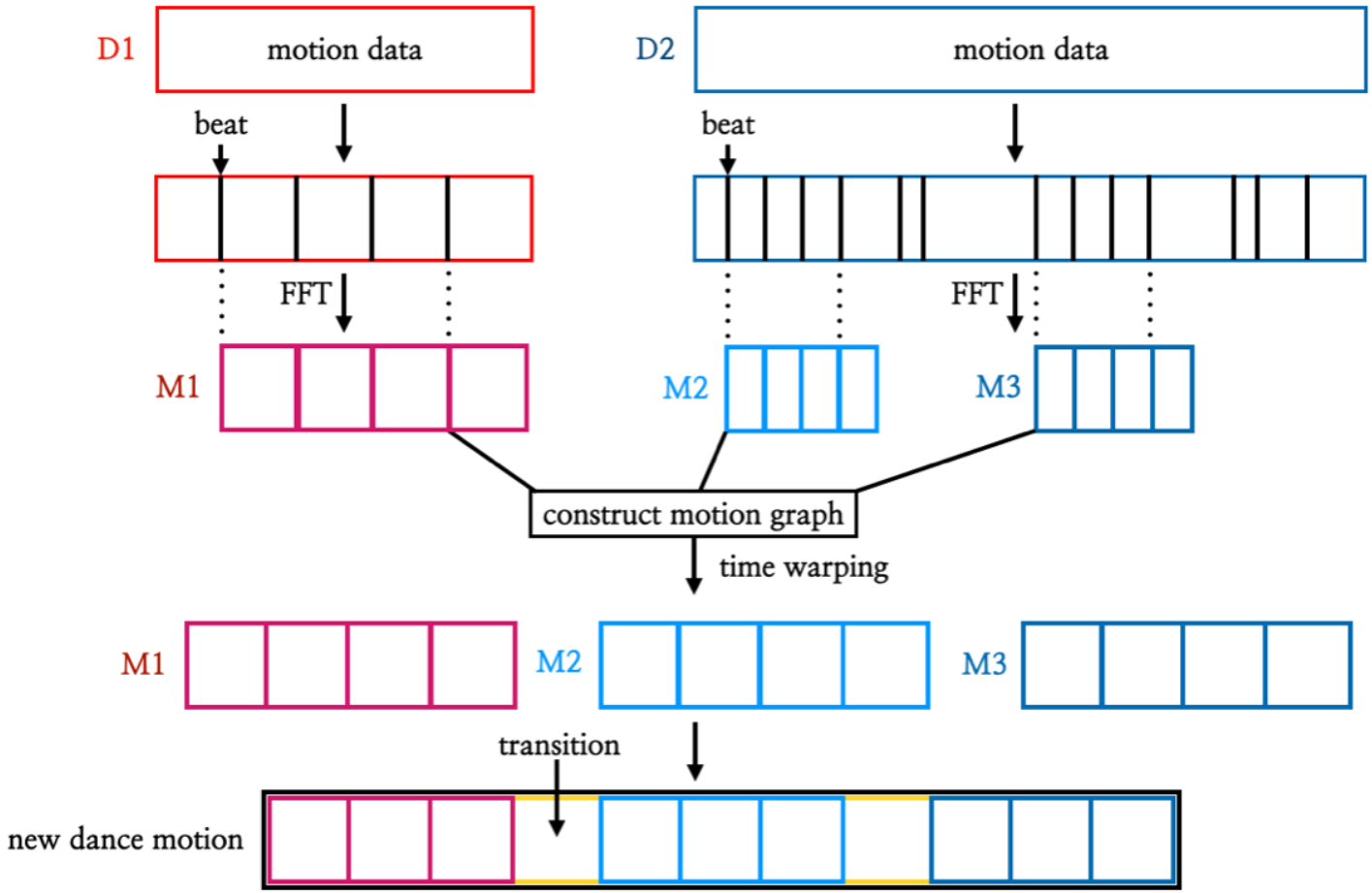


그림 1. 본 논문의 방법에 대한 도식화

제 3 장　비트 탐색 및 마디 추출 방법

이 장에서는 안무 데이터로부터 비트를 추출하고, 일정한 주기로 비트가 반복되는 마디를 찾는 방법에 관해 서술한다. 제 1 절에서는 모션 데이터에서 관절의 정보를 바탕으로 비트를 추출하는 방법을 소개한다. 제 2 절에서는 추출된 비트를 바탕으로 주된 리듬을 갖는 주기를 찾고 마디 단위의 분절하는 방법에 관해 설명한다.

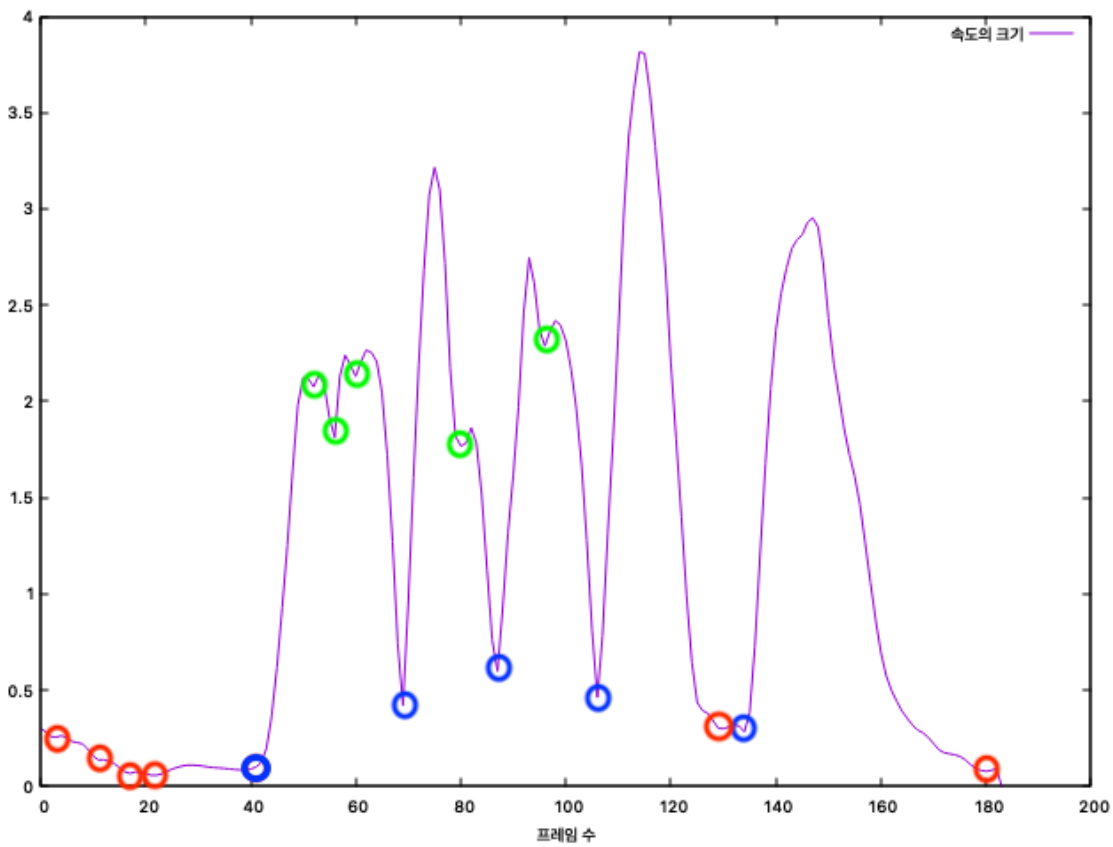


그림 2. 관절들의 속도로 부터 추출한 동작 비트: (빨강) 운동량이 적은 노이즈, (초록) 속도가 빠른 노이즈, (파랑) 적절한 비트.

제 1 절 비트 탐색 방법

본 논문에서 가장 기초적이고 중요한 과정이 비트를 탐색하는 과정이다. 비트의 주기를 구하기 위해서는 정확한 비트가 필요하고, 계산된 주기를 기준으로 마디 단위의 동작 데이터를 추출하기 때문에 적절한 비트의 판별이 필수적이다. 이를 위해 본 논문에서는 기존의 춤 동작으로부터 후보 비트를 찾고, 이를 정제하기 위해 세 가지 조건을 추가로 적용하였다. 우선 후보 비트들을 추출하기 위해 프레임마다 각 관절 속도의 크기를 더하고, 그림 2에서 원형으로 표시된 부분을 찾는다[6]. 하지만 이렇게 추출된 후보 비트들은 관절의 작은 흔들림까지도 비트로 판별하게 된다. 본 논문에서는 이 문제를 해결하기 위한 세 가지 조건을 사용한다.

그림 2에 표시된 것처럼 빨간색과 초록색 원에 해당하는 노이즈 비트들을 처리하기 위해 세 가지 방법을 사용하였다. 먼저 일반적인 춤 음악의 평균 분당 비트 수는 약 128 이므로[11] 더 빠른 분당 비트 수까지 고려하면 비트 사이의 간격은 3분의 1초로 모션 그래프[1]에서 설정하는 윈도우 크기와 유사하다. 따라서 동작 비트는 최소한 윈도우 크기만큼의 간격을 갖는다고 가정하고, 후보 비트의 프레임에서 앞뒤 윈도우 크기만큼의 프레임을 비교하여 속도의 크기가 더 작은 프레임이 존재하는 경우 후보 비트에서 제외한다. 해당 조건을 식으로 나타내면 다음과 같다.

$$F_1(c_i) = \begin{cases} 1 & \text{if } y(c_i + j) > y(c_i), j \in [-w, w], \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

위의 식에서 c_i는 후보 비트의 프레임 번호를 갖는 집합 C의 원소이고 $y(t)$는 t 프레임에서 관절 속도 크기의 합, w는 윈도우 크기다. 이 조건을 사용해 비트가 발생할 수 있는 최소한의 간격에서 속도가 연속적으로 감소하고 증가하는 지점만을 검출할 수 있다.

두 번째로 빨간색으로 표시된 비트들은 사람의 동작에서 흔들림으로 인해 발생하

는 노이즈다. 준비 동작이나 움직임이 적은 동작에서 발생하며, 느린 속도와 적은 운동량을 가지고 있는 특징이 있다. 이러한 문제는 후보 비트로 추출된 프레임과 주변 프레임에서 속도 차이의 합이 임계값 보다 작으면 비트에서 제외하여 해결한다. 하지만 안무마다 속도와 움직임의 크기가 다르기 때문에 적절한 임계값을 찾는 데 어려움이 있다. 본 논문에서는 적절한 임계값을 찾기 위해 동작에서 모든 관절에 대한 속도의 합을 구하고 총 프레임 수로 나누어 평균 운동량을 계산해 임계값으로 설정한다. 해당 조건은 다음과 같은 식으로 표현할 수 있다.

$$F_2(c_i) = \begin{cases} 1 & \text{if } \sum_{j=-w}^{w} y(c_i + j) - y(c_i) \geq T, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

위의 식에서 c_i는 후보 비트의 프레임 번호를 갖는 집합 C의 원소이고 $y(t)$는 t 프레임에서 관절 속도 크기의 합, w는 윈도우 크기, T는 임계값이다.

초록색으로 표시된 비트들은 빨간색으로 표시된 비트들과 마찬가지로 흔들림으로 인해 발생하는 노이즈다. 빨간색으로 표시된 비트들과의 다른 점은 움직임이 큰 동작에서 발생하며, 빠른 속도와 크지 않은 운동량을 갖는 특징이 있다. 식(1)과 식(2)으로 대부분 해결이 가능하지만, 주변 프레임과 운동량의 차이가 큰 경우 두 조건으로 판별되지 않는다. 이 경우 큰 동작에서 발생한다는 특성을 고려하여, 해당 프레임에서 속도의 크기가 임계값 보다 클 때 후보 비트에서 제외하여 해결한다. 이 또한 임계값이 동작의 운동량에 영향을 받기 때문에 식(1)과 마찬가지로 평균 운동량을 임계값으로 설정한다. 해당 조건은 다음과 같은 식으로 표현할 수 있다.

$$F_3(c_i) = \begin{cases} 1 & \text{if } y(c_i) \leq T, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

위의 식에서 c_i는 후보 비트의 프레임 번호를 갖는 집합 C의 원소이고 $y(t)$는 t 프레임에서 관절 속도 크기의 합이다.

위에서 제시한 세 가지 방법의 식(1,2,3)을 후보 비트의 프레임 값을 갖는 집합 C의 원소들에 적용한다. 이를 통해 그림 2의 빨간색과 초록색 비트들이 제외되고, 파란색 비트에 해당하는 원소들이 최종 비트들로 선정된다.

제 2 절 마디 추출 방법

춤 동작은 특정한 간격으로 강조되는 동작 즉, 비트가 발생한다. 따라서 마디를 추출하기 위해서는 먼저 비트들의 적절한 주기를 찾아야 한다. 이를 해결하기 위해 해당 연구에서[5] 제안한 방법을 사용하였다. 이 방법에서는 각 관절에 대해 가속도가 양수에서 음수 또는 음수에서 양수가 되는(zero-crossing) 지점을 찾고 후보 비트로 선정한다. 이렇게 얻은 각 관절에 대한 후보 비트들을 해당 논문의 식(4)과 같이 정현파와 유사한 형태를 보이는 함수로 만들고 모두 더한다. 유사 정현파들을 모두 더해 나온 신호에 고속 푸리에 변환을 진행하여 주파수 범위의 PSD를 구하고 주된 주파수를 찾아 주기를 구한다. 본 논문에서는 제 1 절에서 설명한 조건들을 사용해 최종 비트들을 결정했기 때문에 비트로부터 주기를 찾는 부분만 사용한다. 먼저 앞에서 구한 정제된 비트들을 정현파와 유사한 형태로 만든다. 이를 그림 2에서 구한 정제된 비트들에 적용하면 그림 3와 같이 나타난다. 이 유사 정현파에 고속 푸리에 변환을 적용하고, PSD를 구한다. 계산된 PSD는 그림 4처럼 표현된다. 고속 푸리에 변환에서 동작 데이터의 총 프레임 수 만큼 샘플링하였기 때문에 식(4)으로 비트의 주기를 구할 수 있다.

$$T_{\text{beat}} = \frac{N_{\text{frame}}}{f_{\max}}. \tag{4}$$

T_{beat}는 비트의 주기, N_{frame}는 동작 데이터의 총 프레임 수, f_{max}는 PSD의 진폭이 최대가 되는 주파수다.

이렇게 계산한 주기를 기준으로 마디를 찾는 과정은 다음과 같다. 먼저 식(1,2,3)을 통해 정제된 비트들의 프레임 번호가 담겨 있는 집합 C_{refine}의 원소들을 프레임 번호 기준 오름차순으로 정렬한다. 그다음 첫 번째 원소부터 순회하며 다음 원소와의 프레임 차이가 주기의 크기와 유사한지 확인한다. 이때 오차는 윈도우의 크기다. 연속된 4개의 원소가 앞 조건을 만족하면 하나의 마디가 된다.

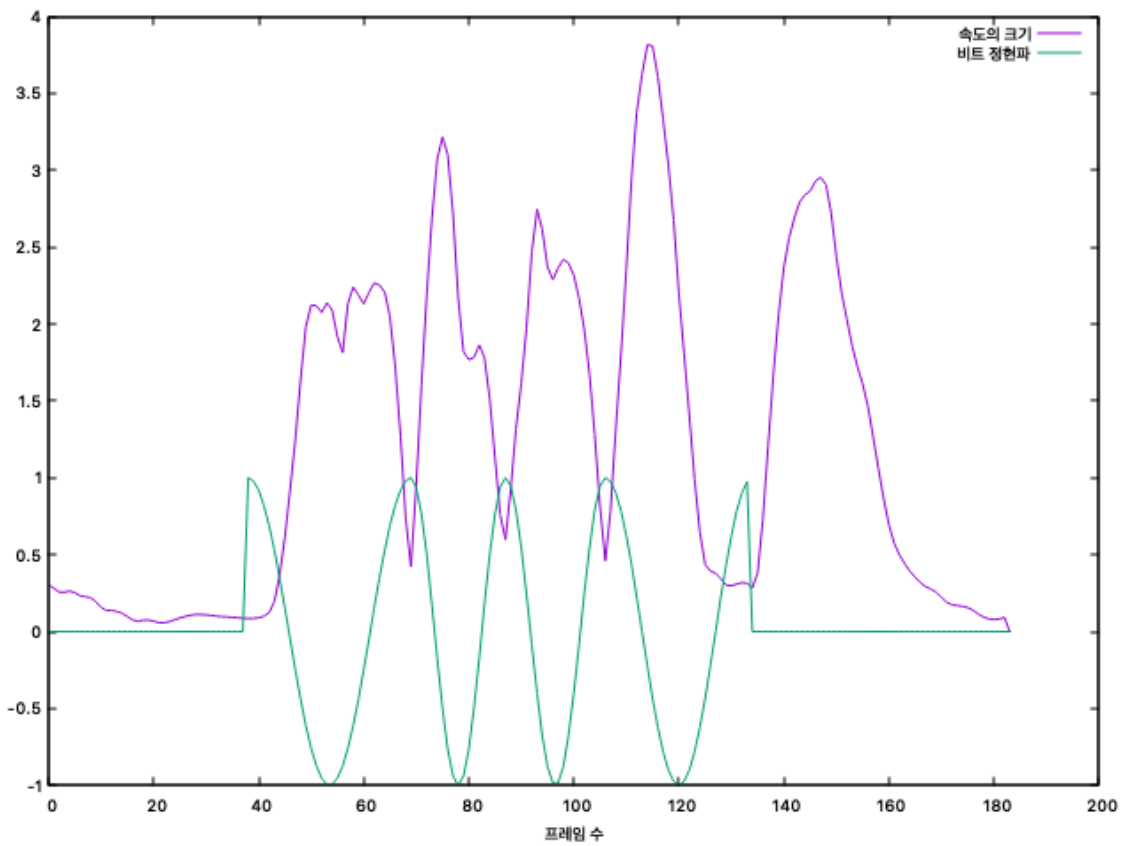


그림 3. 정제된 비트들에 대한 정현파 맞춤.

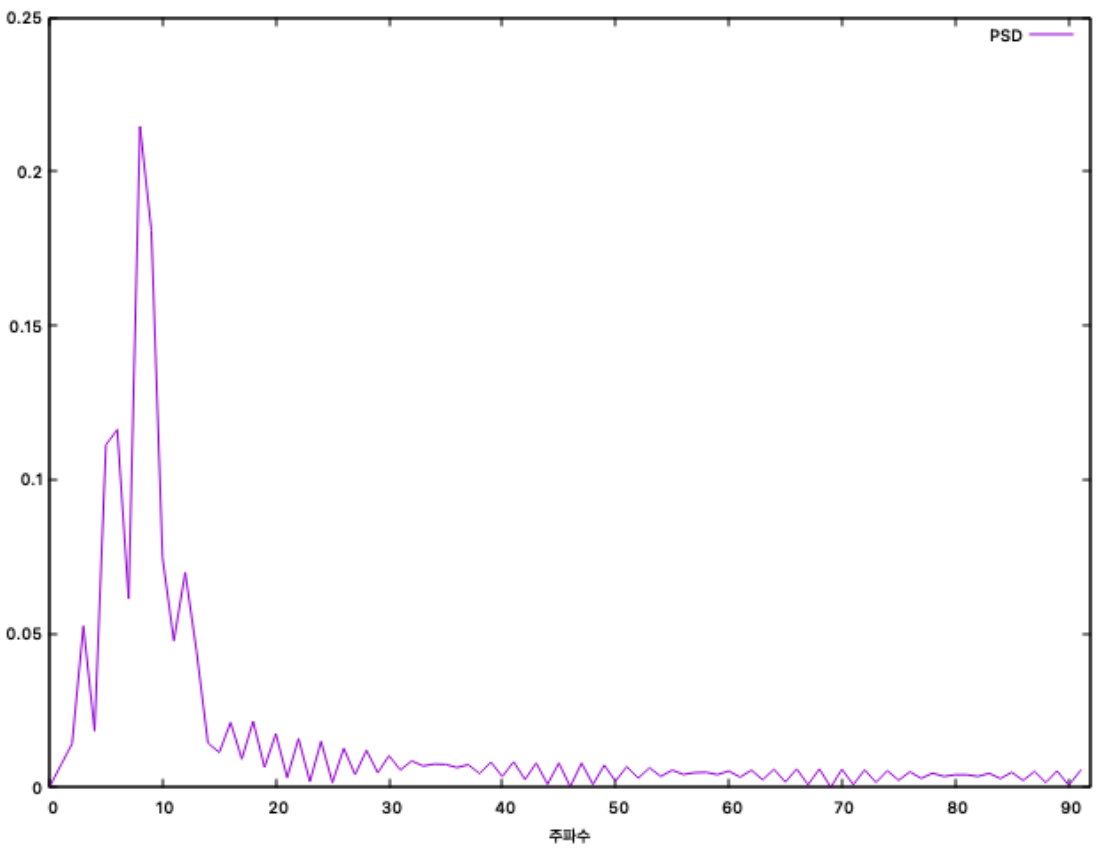


그림 4. 정현파에 대해 고속 푸리에 변환을 적용한 PSD.

제 4 장　춤 동작을 위한 모션 그래프 생성 방법

이 장에서는 추출한 마디를 기반으로 새로운 동작 데이터를 합성하는 춤 동작을 위한 모션 그래프 생성 방법에 대해 소개한다. 먼저 기존 모션 그래프의 생성 방법을 간단히 설명하면 다음과 같다. 춤 동작의 각 프레임마다 관절들을 중심으로 포인트 클라우드를 생성한다. 그다음 모든 프레임에 대해 루트 관절의 xz좌표와 xz축 회전이 0이 되도록 포인트 클라우드를 정렬시킨다. 모든 동작에서 정렬된 포인트 클라우드를 기준으로 거리를 계산해 거리 지도를 구한다. 추출된 거리 지도들에 대해 특정 임계점보다 작은 지역 최솟값을 구하면, 지역 최솟값은 동작의 어떤 프레임이 서로 연결될 수 있는지를 나타낸다. 이렇게 연결될 수 있는 지점을 기준으로 가장 큰 강한 연결 요소들을 찾아 그래프를 완성하고, 이를 기반으로 서로 다른 동작들을 합성하여 새로운 동작을 생성한다.

본 논문에서는 기존의 모션 그래프 생성 과정을 변형하여 춤 동작을 위한 모션 그래프를 생성한다. 우선 동작 데이터들의 거리 지도를 구하는 과정은 동일하나 그래프를 생성하는 과정에서 차이가 있다. 기존의 모션 그래프는 거리 지도에서 임계값 이하의 지역 최솟값을 구하고, 이를 연결 될 수 있는 두 동작으로 가정해 그래프를 생성한다. 하지만 본 논문에서는 춤 동작의 마디를 추출하고 그래프의 노드로 설정한다. 박자와 연속성을 보존해야 하는 춤 동작의 특성상 한 마디가 끝났을 때 다음 마디로 연결되어야 한다. 따라서 엣지는 마디의 첫 프레임과 마지막 프레임을 연결한다. 이는 한마디의 마지막 프레임에서 다른 마디의 첫 프레임으로만 연결될 수 있음을 의미한다. 하나의 마디만 표현하는 동작 데이터도 존재하고, 이러한 경우 동작 데이터에서 한 개의 노드만 생성된다. 동작에서 검출되지 않은 마디도 존재할 수 있기 때문에 본 논문에서 모션 그래프의 크기는 일반적인 모션 그래프에 비해 훨씬 작아진다. 따라서 노드를 생성할 때 연결될 수 있는 노드들의 거리를 정하는 임계값을 적절히 조절하여, 자연스러운 연

결과 충분한 노드 생성이 이루어질 수 있도록 그래프를 생성한다.

이렇게 생성된 그래프는 탐색 중 마디가 반복되거나(sink) 다른 마디와 연결되지 않을 수 있다(dead end). 따라서 강한 연결 요소 알고리즘을 적용하여 가장 큰 강한 연결 요소를 제외한 노드와 엣지들에 대해 가지치기를 진행한다.

제 5 장　음악 동기화 및 비트를 고려한 동작 블렌딩

이 장에서는 음악과의 동기화를 위해 마디의 구조와 길이를 맞추고, 동작 비트의 특징을 유지하는 블렌딩 방법을 소개한다. 제 1 절에서는 음악의 마디와 동작의 마디를 비교하여 구조를 맞추는 방법과 입력받은 분당 비트 수에 맞는 길이를 설정하는 방법에 관해 설명한다. 제 2 절에서는 마디를 블렌딩 할 때 연결되는 비트를 유지하는 변환 동작 생성 방법에 관해 이야기한다.

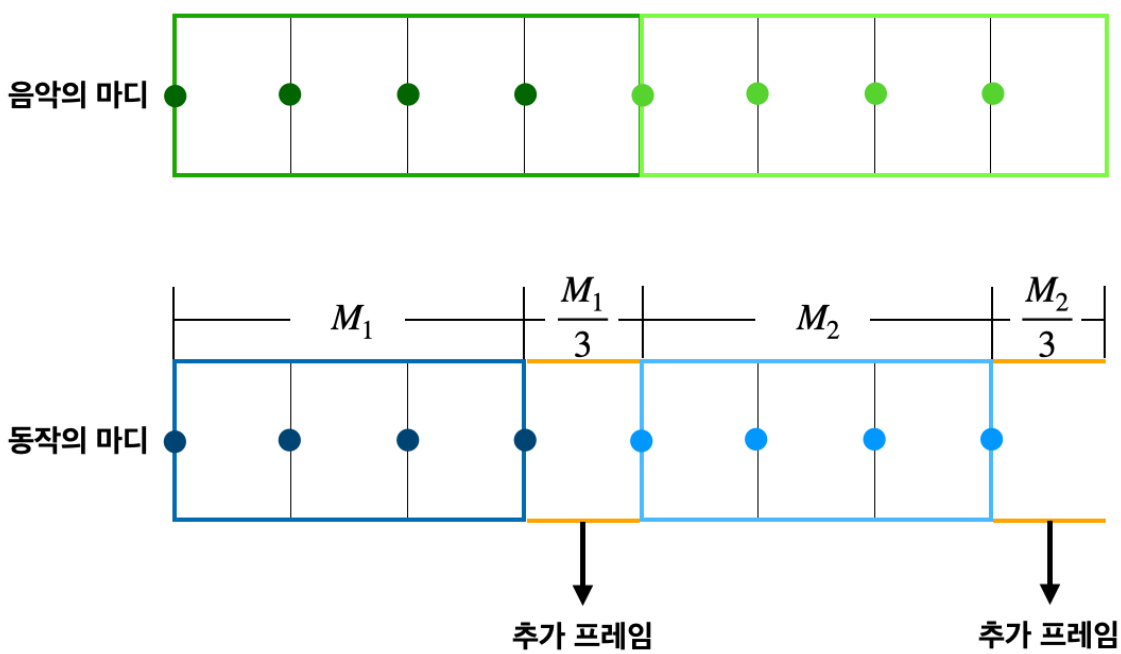


그림 5. 음악의 마디 구조와 본 논문에서 추출된 마디 구조의 비교.

제 1 절 음악 동기화 방법

그림 5을 보면 음악 마디의 구조와 제 2 절에서 구한 동작 마디의 구조가 다른 것을 확인 할 수 있다. 음악과 춤 동작을 동기화하기 위해서는 비트의 위치와 마디의 구조를 맞춰야 한다. 이는 동작 마디 길이의 3분의 1만큼 추가하여 해결할 수 있다. 이 방법은 첫 번째와 네 번째 비트의 위치를 음악의 비트와 정확히 일치시킬 수 있으나 두 번째와 세 번째 비트는 정확히 일치 하지 않을 수 있다. 하지만 사람에 의한 동작이기 때문에 모든 춤 동작이 음악의 비트와 정확히 일치할 수 없으며, 동작 마디를 추출할 때 주기와 근사한 간격을 가진 비트들을 사용하였기 때문에 충분히 자연스러운 결과를 보인다.

음악과 동작의 마디 구조 뿐만 아니라 입력받은 분당 마디 수 에 맞춰 길이도 동기화 시켜야 한다. 본 논문에서는 4분의 4박자를 기준으로 음악의 분당 비트 수를 입력 받기 때문에 한 마디의 길이를 계산할 수 있다. 이를 구하는 방법을 식으로 표현하면 다음과 같다.

$$T_{\text{meter}} = \frac{60}{\text{분당 비트 수}} \times 4. \tag{5}$$

위의 식에서 T_{meter}는 마디의 길이 분당 비트 수는 입력받은 분당 비트 수이다. 식(5)에서 얻은 음악 마디의 길이를 사용하여 동작의 마디 길이가 몇 프레임이 되어야 하는지도 다음과 같은 식으로 계산할 수 있다.

$$F_{\text{meter}} = T_{\text{meter}} \times \text{FPS}. \tag{6}$$

위의 식에서 F_{meter}는 음악과 동기화된 동작 마디의 프레임 수 FPS는 동작의 FPS(Frame Per Second)다. 식(6)에서 구한 F_{meter}으로 동작 마디의 프레임을 시간변형 시키면 음악과 동작의 마디 길이를 동기화 시킬 수 있다.

제 2 절 비트를 고려한 동작 블렌딩

음악과 동작에서 길이와 비트 위치를 동기화했다면, 적절한 블렌딩을 사용한 동작의 합성이 이루어져야 한다. 모션 그래프를 활용하는 이전 연구들은[1][5][8] 동작들의 합성을 위해 앞뒤에 프레임을 추가하여 자연스러운 합성이 가능하게 하였다. 이는 모션 그래프의 생성 과정에서도 포인트 클라우드 생성 시 앞뒤 프레임들의 이동을 고려하여 동작간의 거리를 구하기 때문에 자연스럽고 적절한 방법이다. 그림 6의 (a)는 기존의 블렌딩 방법을 사용한 결과다. 이전 연구들과 같이 추출된 동작 마디의 앞뒤에 프레임을 추가하고, 두 동작에 대하여 추가된 프레임을 블렌딩에 사용하여 변환 동작을 만들었다. 하지만 이러한 방법의 사용은 본 논문의 성격상 몇 가지 어려움이 있다. 첫 번째로 블렌딩을 위해 추가된 프레임이 기존 마디 길이의 3분의 1이기 때문에 주기의 길이와 유사해진다. 따라서 추가된 프레임에 다른 비트가 포함될 가능성이 높다. 이를 포함하여 블렌딩을 진행하면 변환 동작에 비트와 유사한 움직임이 발생한다. 두 번째로 그림 6의 (a)의 궤적을 보면 변환이 시작된 후에 속도에 급격한 변화가 발생한다. 이러한 현상은 블렌딩을 위해 추가된 프레임들의 속도가 빠를수록 심해진다. 변환 동작의 속도가 급변하게 되면 연결되는 비트가 아닌 변환 동작이 비트처럼 인식될 수 있다.

앞서 소개한 문제들을 해결하기 위해 본 논문에서는 다른 방법으로 블렌딩을 진행한다. 우선 자연스러운 블렌딩을 위해 추가적인 프레임이 필요하므로 마디로 추출된 동작의 앞부분에만 마디 길이의 3분의 1만큼 프레임을 추가한다. 이후 뒤 동작에서 첫 번째 프레임 뿌리 관절의 xz평면 좌표를 앞 동작의 마지막 프레임 뿌리 관절의 xz평면 좌표로 2차원 변형시킨다. 마지막으로 앞 동작의 마지막 비트에 해당하는 프레임과 뒤 동작에 추가된 프레임을 윈도우 크기만큼 블렌딩한다. 변환 동작을 블렌딩 하는 방법은 식(7,8,9)를 통해 설명한다. 윈도우의 크기 k에 대하여 가중치 α에 대한 식은 다음과 같다.

$$\alpha(p) = 2(\frac{p}{k})^3 - 3(\frac{p}{k})^2 + 1, \quad 1 \leq p \leq k-1. \tag{7}$$

식(7)은 일차 미분까지 연속을 보장하고 p는 변환 동작의 프레임이다. $1 \leq p \leq k-1$에 대하여 뿌리 관절의 위치를 블렌딩하는 방법에 대한 식은 다음과 같다.

$$R_p = \alpha(p)R_{A_{\text{last}}} + [1-\alpha(p)]R_{B_i}. \tag{8}$$

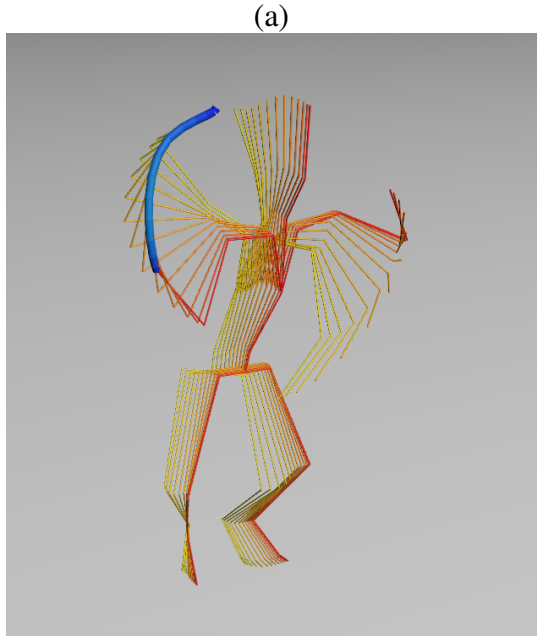
식(8)에서 $R_{A_{\text{last}}}$는 앞 동작에서 마지막 프레임 뿌리 관절의 위치고, R_{B_i}는 $0 \leq i \leq k-2$에 대하여 뒤 동작에서 i번째 프레임의 뿌리 관절의 위치다. 관절들의 회전을 블렌딩하는 식은 다음과 같다.

$$q_p^j = slerp(q_{A_{\text{last}}}^j, q_{B_i}^j, \alpha(p)). \tag{9}$$

식 (9)에서 N개의 관절을 가질 때 $0 \leq j \leq N-1$이다. q_p^j는 변환 동작의 p프레임에서 j번째 관절의 회전값을 의미한다. $q_{A_{\text{last}}}^j$는 앞 동작의 마지막 프레임에서 j번째 관절 회전값이고, $q_{B_i}^j$는 뒤 동작의 i번째 프레임에서 j번째 관절의 회전값이다.

이러한 방법을 사용하면 앞 동작은 마지막 비트만 사용하여 블렌딩 하므로 해당 동작이 더 오래 유지되어 비트가 강조되는 효과를 갖는다. 뒤 동작은 변환 동작이 끝난 후 추가 되었던 나머지 프레임이 재생되기 때문에 본래의 비트를 살릴 수 있다. 그림 6의 (b)를 보면 (a)의 궤적보다 단순하고 속도의 변화도 작은 것을 확인 할 수 있다. 이는 비트 사이의 변환 동작이 단순하고 부드럽게 블렌딩 되어 마디와 마디 사이의 비트 연결을 해치지 않음을 보여준다.

(a)



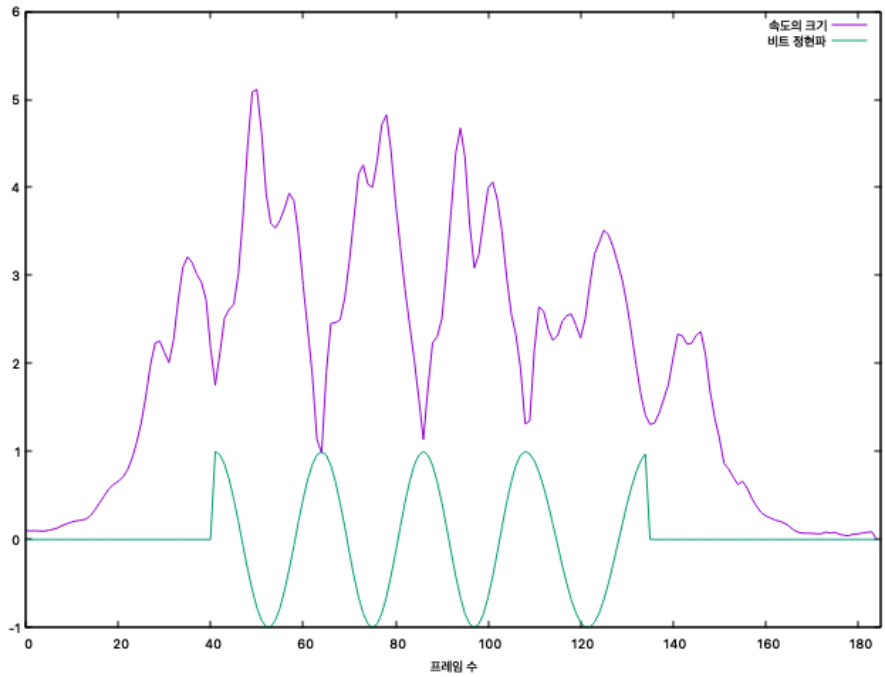
(b)

그림 6. 두 가지 방법에 대해 블렌딩을 진행한 변환 동작이다. 그림에서 궤적은 비트가 되는 관절의 이동 경로를 나타내며, 파란색과 초록색에 가까울 수록 느린 속도와 빠른 속도를 의미한다. (a) 모션 그래프의 방법. (b) 본 논문에서 제안하는 방법.

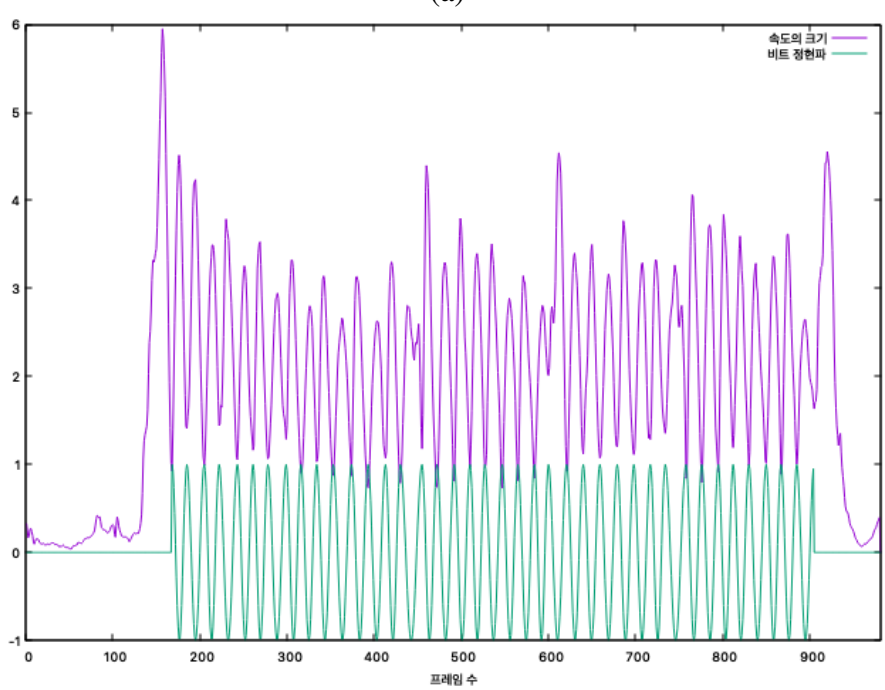
제 6 장 결과

본 논문에서 제안하는 방법의 결과를 보이기 위해 다음과 같은 순서로 진행한다. 먼저 동작 데이터에서 제 1 절의 조건을 통해 추출된 비트들이 그림 2의 노이즈들을 처리하였는지 그래프를 통해 보인다. 그다음 고속 푸리에 변환을 통해 찾은 평균 주기가 실제 춤 동작의 비트 간격을 표현하는지 확인한다. 이는 생성된 마디를 프레임 단위로 조금씩 이동시켜 나타내고, 비트에 해당하는 프레임과 비트가 되는 주요 관절의 궤적을 표시하여 나타낸다. 변환 동작도 이와 같은 방법으로 보인다. 마지막으로 120분당 비트 수를 가정하고 마디의 길이를 맞춰 시간 변형을 진행한다. 이렇게 시간 변형된 춤 동작들을 합성하여 최종적으로 생성되는 춤 동작을 보인다.

그림 7은 두 개의 춤 동작 데이터에 대한 속도의 크기와 검출된 비트 정현파의 그래프를 보여준다. 본 논문에서 제시한 식(1,2,3)을 사용하여 비트를 검출하면, 그림 7의 그래프를 통해 알 수 있듯이 제 1 절에서 이야기한 노이즈들을 제외하고 비트가 검출된 것을 확인 할 수 있다. 이렇게 검출된 비트들에 대해 고속 푸리에 변환을 진행하여 주파수 영역의 PSD를 구한다. 이때 가장 큰 진폭을 가지고 있는 주파수를 선정하고, 식(4)을 통해 평균 주기를 구한다. 그림 8에서 이러한 과정을 그래프로 나타낸다. 지금까지 춤 동작의 최소 단위인 마디 단위로 동작 데이터를 나누기 위해 비트와 주기를 계산하고 이에 대한 결과를 그림으로 나타냈다. 마지막으로 평균 주기의 간격을 가진 비트들을 묶어 하나의 마디를 결정한다. 이렇게 만들어진 마디들은 4개의 비트를 가지고 있어야 하고 비트 사이의 간격이 비슷해야 한다. 또 눈으로 보았을 때 충분히 비트라고 느낄 수 있어야 한다. 이를 보이기 위해 그림 9에서 프레임마다 조금씩 이동시킨 마디 단위의 동작을 나타낸다. 빨간색으로 표현된 프레임과 관절의 궤적을 통해 마디에 담긴 4개의 비트들을 확인 할 수 있다.

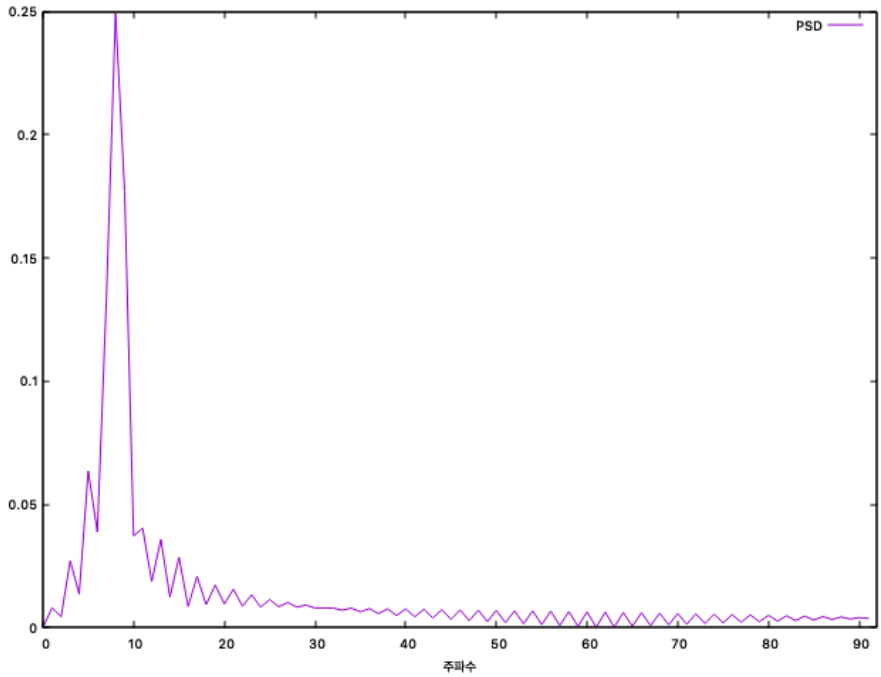


(a)

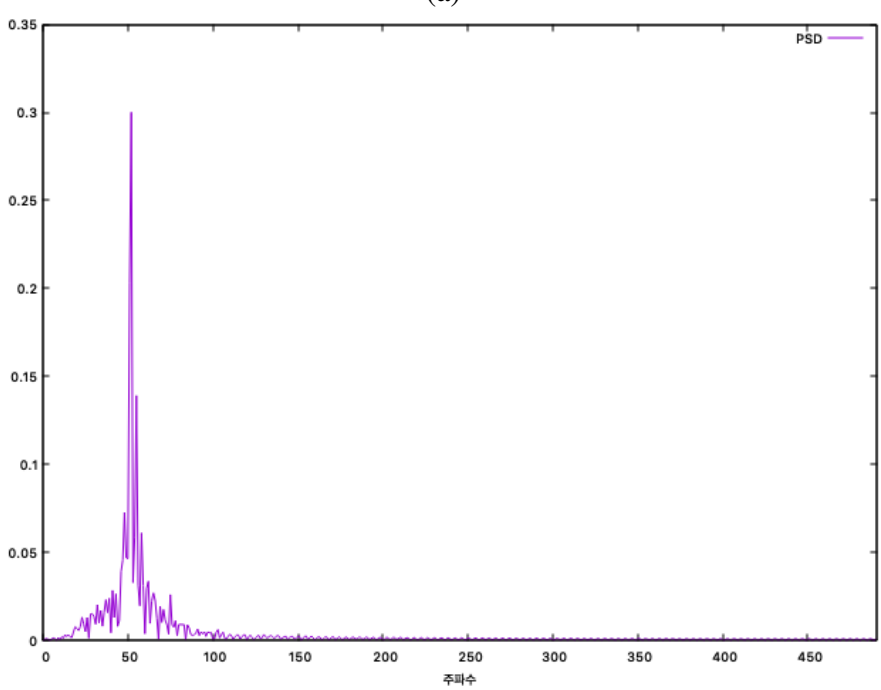


(b)

그림 7. 동작 데이터 속도의 크기와 본 논문의 방법을 통해 검출한 비트 정현파의 그래프. (a) 짧은 동작 데이터. (b) 긴 동작 데이터.

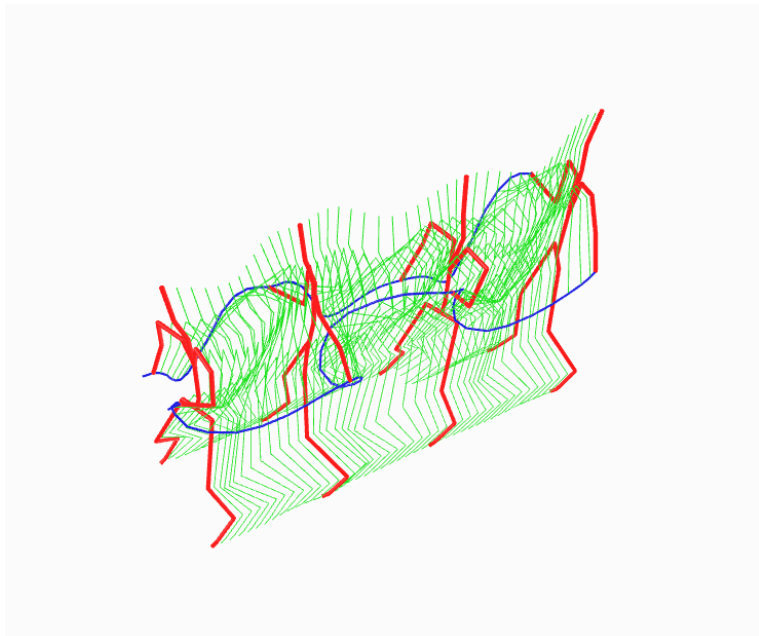


(a)

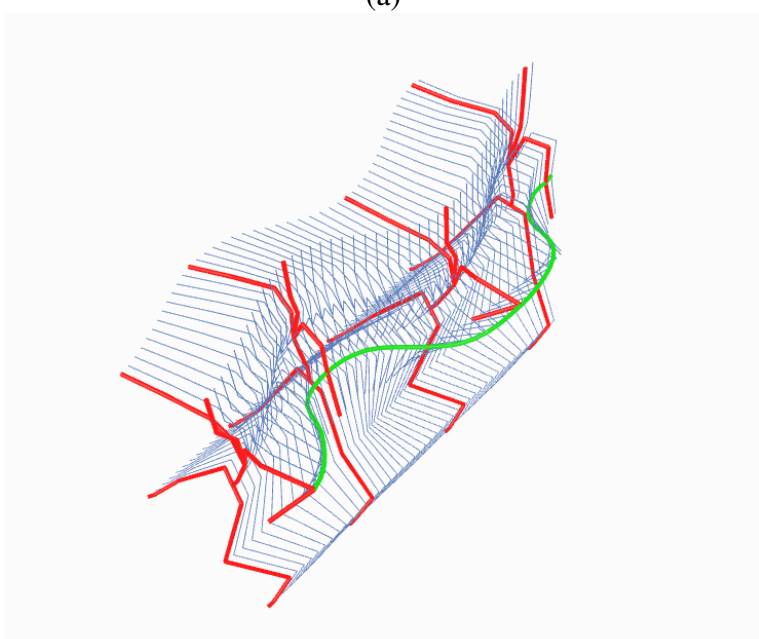


(b)

그림 8. 비트들을 통해 고속 푸리에 변환을 진행한 결과. (a) 짧은 동작 데이터. (b) 긴 동작 데이터.



(a)



(b)

그림 9. 동작 데이터에 대해 추출된 마디. 비트에 해당하는 프레임은 빨간색으로 표시하였다. (a) 짧은 동작 데이터. (b) 긴 동작 데이터.

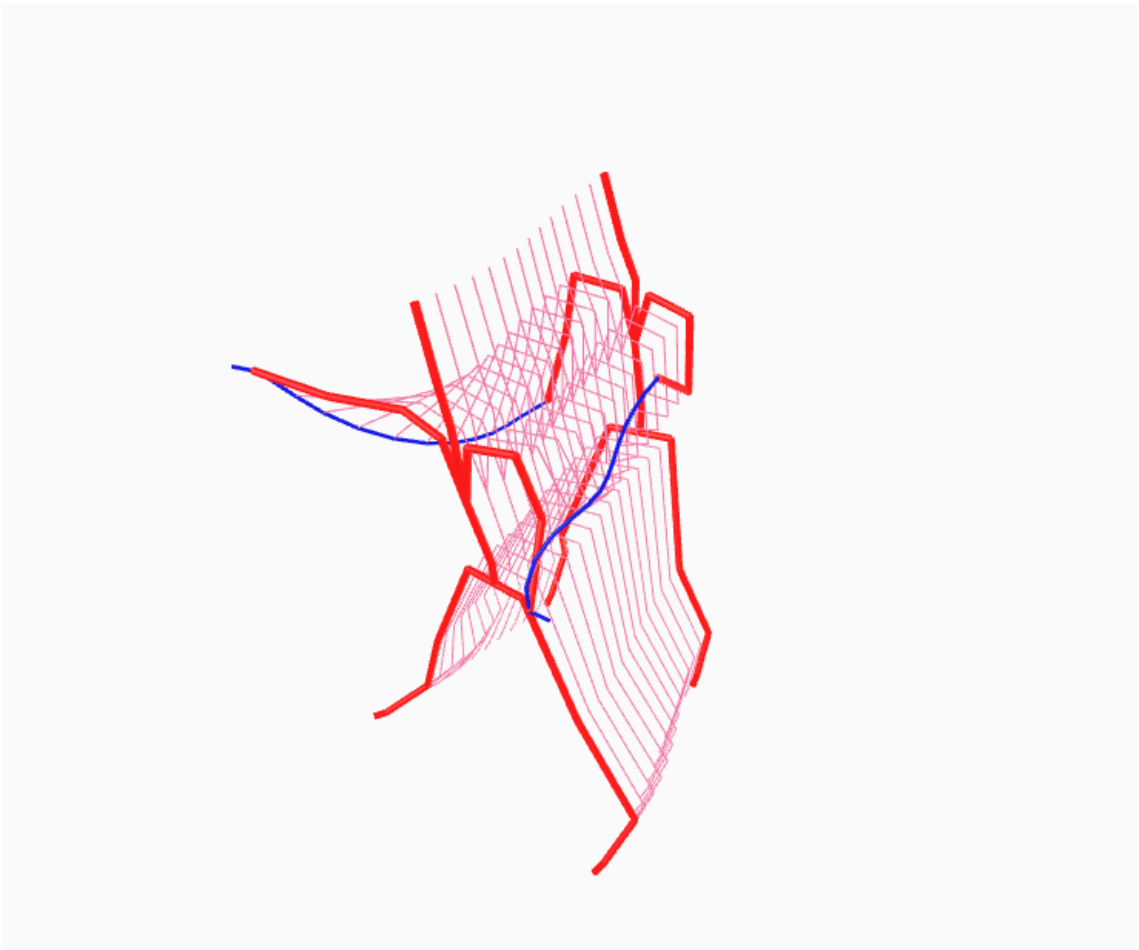


그림 10. 두 동작 마디의 변환 동작.

춤 동작 데이터에서 마디들이 적절하게 추출된 것을 확인하였다. 이 마디들을 합성하기 위해서는 적절한 블렌딩 방법이 필요하다. 본 논문에서 제시한 제 5 장의 방법을 사용하여 변환 동작을 생성하면 다음 그림10과 같이 나타난다. 해당 동작을 보면 시작과 끝의 비트 동작을 최대한 유지하면서 부드럽게 변환되는 것을 확인할 수 있다.

마지막으로 분당 비트 수에 맞춰 추출한 마디들을 시간 변형하고, 모션 그래프의 형태로 만들어 탐색하는 노드들을 합성해 새로운 춤 동작을 만든다. 앞서 보여준 결과들을 통해 생성된 춤 동작은 다음 그림11에서 보여준다.

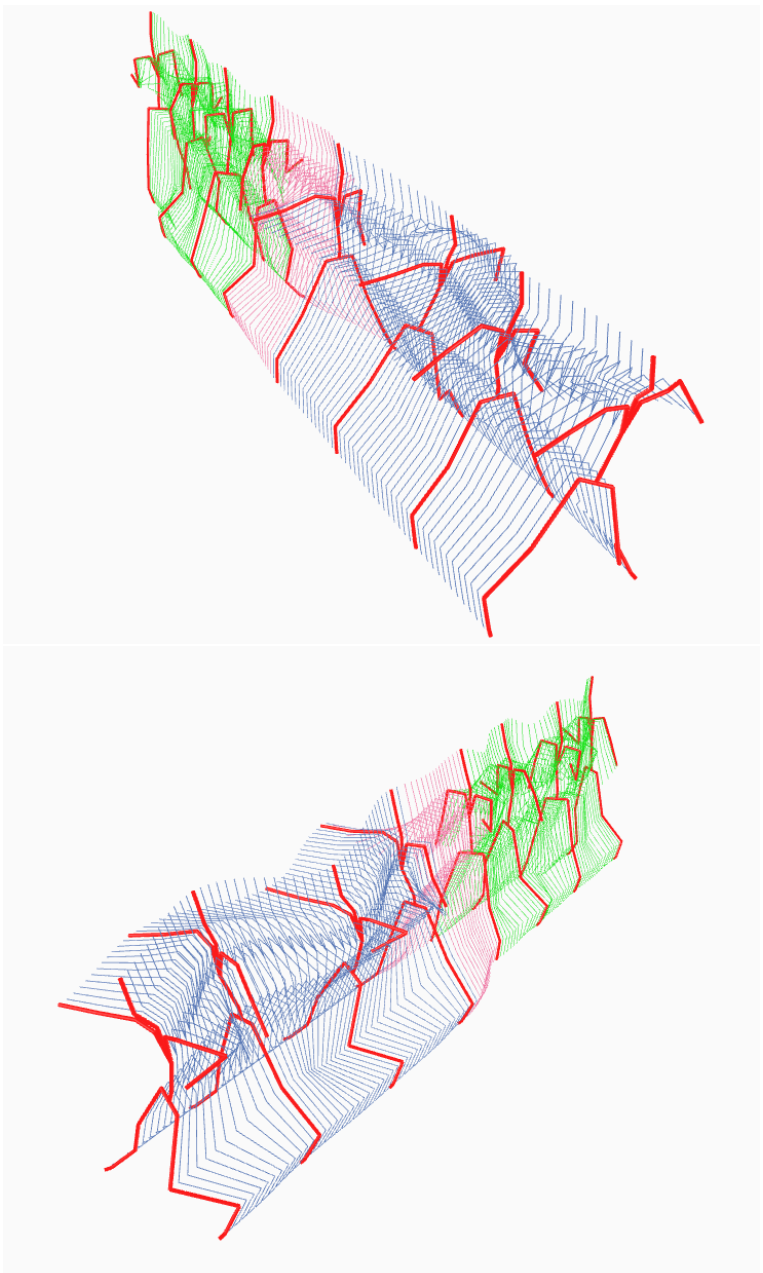


그림 11. 최종적으로 생성된 춤 동작.

제 7 장　결론

본 논문은 춤 동작 데이터에서 비트 탐색 후 마디 단위로 추출하여 모션 그래프를 생성하고, 적절한 합성을 통해 새로운 춤 동작을 생성하는 방법을 제안한다. 해당 방법은 임의의 춤 동작 데이터에서도 자연스러운 마디 단위의 동작을 추출할 수 있으며, 추출된 모든 마디를 음악의 길이에 맞게 변형하여 합성하기 때문에 적은 데이터로도 다양한 춤 동작을 생성할 수 있다. 또 모션 캡쳐된 데이터를 활용하고 춤 동작의 맞는 합성 방법을 진행하기 때문에 결과의 품질도 보장할 수 있다. 결과적으로 임의의 음악 정보가 입력되면 동기화된 자연스러운 춤 동작을 생성할 수 있다는 것을 확인했다.

본 논문의 방법은 크게 두 가지 방법으로 개선될 수 있다. 먼저 음악에 대한 분석을 통해 훨씬 더 동기화된 춤 동작을 생성할 수 있다. 현재 음악에 대한 분당 마디 수를 입력받아 길이에 대한 동기화만 가능한 상태다. 이러한 이유로 본 논문의 모션 그래프 탐색 시 임의의 노드를 찾아 합성을 진행하여 음악의 구조와 흐름에 맞지 않는 진행이 이루어진다. 하지만 음악의 신호에 대한 분석과 최근 활발히 연구되고 있는 인공지능 기술을 활용하면, 음악의 구조나 분당 마디 수, 장르 등을 분석해 이를 가중치로 그래프를 탐색하여 더 자연스러운 춤 동작을 생성할 수 있다. 다음으로 동작 데이터에 대한 충분한 확보를 통해 분석된 음악에 맞는 춤 동작을 선택할 수 있다. 춤에는 박자가 같더라도 장르에 따라 다른 동작으로 구성된다. 따라서 각 장르에 따른 춤 동작이 충분히 확보된다면, 앞서 분석한 음악들에 대해 장르를 추출하고 이에 맞춰 춤 동작을 생성할 수 있다.

본 논문의 접근 방법에서 근본적인 문제도 있다. 기존의 비트 탐색 방법과 본 논문에서 제시한 조건들을 활용하여 대부분의 춤 동작에서 적절한 비트의 탐색이 가능하다. 하지만 속도 크기의 극솟값을 기반으로 탐색하는 방법의 특성상 웨이브 동작처럼 강조되는 동작 없이 여러 개의 비트를 비슷한 속도로 이동하기 때문에 검출이 힘들다.

또 춤 동작 데이터에 준비 동작같이 춤이 아닌 다른 동작이 존재할 경우 비트로 탐색되어 잘못된 마디가 추출될 수 있고, 다른 박자를 갖는 동작이 섞여 있는 경우 주기 검출에 어려움이 있다. 이러한 문제를 해결하기 위해 비트 탐색을 위한 추가적인 연구가 필요하다.

참고 문헌

[1] L. Kovar, M. Gleicher, and F. Pighin, "Motion graphs," *ACM Trans. Graph.*, vol. 21, p. 473–482, jul 2002.

[2] K. Chen, Z. Tan, J. Lei, S.-H. Zhang, Y.-C. Guo, W. Zhang, and S.-M. Hu, "Choreomaster: Choreography-oriented music-driven dance synthesis," *ACM Trans. Graph.*, vol. 40, jul 2021.

[3] Z. Ye, H. Wu, J. Jia, Y. Bu, W. Chen, F. Meng, and Y. Wang, "Choreonet: Towards music to dance synthesis with choreographic action unit," in *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, (New York, NY, USA), p. 744–752, Association for Computing Machinery, 2020.

[4] R. Li, S. Yang, D. A. Ross, and A. Kanazawa, "Ai choreographer: Music conditioned 3d dance generation with aist++," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 13401–13412, October 2021.

[5] T.-h. Kim, S. I. Park, and S. Y. Shin, "Rhythmic-motion synthesis based on motion-beat analysis," *ACM Trans. Graph.*, vol. 22, p. 392–401, jul 2003.

[6] C. Ho, W.-T. Tsai, K.-S. Lin, and H. H. Chen, "Extraction and alignment evaluation of motion beats for street dance," in *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pp. 2429–2433, IEEE, 2013.

[7] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of mfcc," *J. Comput. Sci. Technol.*, vol. 16, p. 582–589, nov 2001.

[8] H. J. Shin and H. S. Oh, "Fat graphs: Constructing an interactive character with continuous controls," in *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, (Goslar, DEU), p. 291–298, Eurographics Association, 2006.

[9] C. Rose, M. Cohen, and B. Bodenheimer, "Verbs and adverbs: multidimensional motion interpolation," *IEEE Computer Graphics and Applications*, vol. 18, no. 5, pp. 32–40, 1998.

[10] S. I. Park, H. J. Shin, and S. Y. Shin, "On-line locomotion generation based on motion blending," in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, (New York, NY, USA), p. 105–111, Association for Computing Machinery, 2002.

[11] D. Moelants, “Dance music, movement and tempo preferences,” in *Proceedings of the 5th Triennial ESCOM Conference*, pp. 649–652, Hanover University of Music and Drama, 2003.

Abstract

This paper proposes a method for analyzing the rhythm of dance movements, dividing them into phrases, and synthesizing motion data to match the music. To achieve phrase-level segmentation, it is crucial to first identify the beats of the movements. Existing methods utilizing joint velocities to detect beats were adopted, with the addition of conditions based on joint and motion average kinetic energy. The dominant periodicity of the identified beats is determined using fast Fourier transform. By using this periodicity, beats satisfying the specified conditions are grouped to form phrases. The extracted phrases are then appropriately synthesized. Traditionally, motion graphs are employed to generate new movements by blending multiple motion data. However, applying this method directly to the proposed approach presents challenges. Therefore, this paper modifies the graph generation process of motion graphs to create motion graphs specifically for dance movements. When synthesizing movements from the generated motion graph, the conventional approach involves adding extra frames at the beginning and end of connecting movements. However, applying this blending technique to dance movements segmented into phrases may result in a lack of rhythmic coherence, particularly when blending the first and last beats of a phrase. To address this issue, this paper introduces a new blending method that transforms the conventional blending approach, preserving the sense of beats while synthesizing movements.