PCA를 이용한 효율적 모션 그래프 생성

성혜영[1] 경민호[2]
아주대학교 미디어학부 게임애니메이션 센터
piopio@ajou.ac.kr[1]
kyung@ajou.ac.kr[2]

Fast construction of motion graph using PCA

Hye-Young Seong[1] and Min-Ho Kyung[2]
Game and Animation Center
Media Division
Ajou University

요 약

모션 데이터들을 그래프로 저장하고 이를 모션합성에 이용하는 기존의 연구들은, 모든 모션 프레임간 연결비용계산으로 인하여 그래프 생성에 많은 시간이 걸린다는 단점이 있다. 본 논문에서는 이런 단점을 보완하여 빠르고 효과적으로 그래프를 생성하는 방법을 제시한다. 우선, PCA를 이용하여 모션들을 2차원에 투영시키고, 2차원 상의 간단한 거리계산으로 전이에지가 존재할 가능성이 큰 프레임 쌍들을 찾아낸다. 다음으로, 이런 프레임 쌍에 대해서만 연결비용을 계산하여 그래프를 생성한다. 따라서, 모든 프레임에 대한 비용계산에 비해 본 논문에서 제안한 방법은 효율적으로 그래프를 생성하게 된다.

1. 서론

3차원 애니메이션 제작에서 캐릭터의 동작을 사실적이고 자연스럽게 만들어내는 것은 매우 중요한 작업이다. 이를 위해 최근에는 모션 캡쳐 방식이 널리 사용되고 있다. 모션 캡쳐는 실제 배우의 연기를 카메라로 찍어 모션 데이터를 얻는 방법으로 매우 사실적이고 고품질의 데이터를 얻을 수 있다는 장점이 있다. 그러나 이 방식은 장비를 이용하는 데 드는 비용이 높은데다, 한번 얻어진 데이터는 애니메이터가 원하는 대로 수정해서 재사용하는 것이 쉽지 않다는 단점이 있다.

최근, 기존의 모션을 바탕으로 모션 데이터의 재사용성을 높이고, 효율적으로 새로운 모션을 생성하려는 연구들이 많이 진행되고 있다. 이러한 연구 중에는, 여러 모션 데이터들 간의 연결성을 고려하여 그래프 형태로 만들어 저장하고, 이것을 적절한 방식으로 검색하여 모션을 재구성하는 방법이 있다. Kovar[6]는 3차원 캐릭터의 모션을 제어할 수 있도록 한 모션그래프 개념을 제시하고, 자동적으로 그래프를 만들어 이로부터 사실적인 모션을 합성해내는 과정을 보여주었다. Lee[3]는 모션 데이터 간의 전이가 가능한 연결관계를 관절들의 각 차이, 관절들의 속도 차이, 자세의 위치 차이와 속도 차이 등을 고려하여 확률의 매트릭스 형태로 나타내고 이로부터 그래프 구조를 얻었다. Arikan[8]은 관절들의 각 차이와, 자세의 위치 차이 등을 바탕으로 그래프를 계층적으로 만드는 방법을 제시하였다. 그러나, 이런 연구들은 그래프 생성시, 모든 모션의 모든 프레임 간 연결성을 계산하여 전이에지를 만들어야 하므로 너무 많은 시간이 걸린다는 문제점이 있다. 즉, 그래프 생성시간이 모든 모션의 프레임 수의 합의 제곱에 비례하므로 모션의 수가 많아지면 많아질수록 그 시간은 크게 늘어난다. 또한, 데이터베이스에 새로운 모션이 추가될 경우에도 새로운 모션과 이미 존재하는 모션들 간에 다시 연결성 계산을 해야 하므로 데이터베이스 갱신에도 많은 시간이 요구된다. 모션을 합성하는데 있어서 데이터베이스의 풍부함은, 얻어지는 모션의 자연스러움과 다양성에 끼치는 영향이 크기 때문에 모션 수의 증가는 피할 수 없다. 따라서, 많은 모션을 다루어야 하는 모션 데이터베이스에 있어서, 그래프 생성에 대한 효율적 알고리즘의 필요성이 크다고 할 수 있다.

이 논문에서는 모션 데이터를 더욱 빠르고 효과적으로 그래프화하는 방법을 제시하고자 한다. 기존의 연구에서처럼 모든 프레임 간의 연결성을 계산하는 대신, 간단한 비용계산으로 전이에지가 존재할 가능성이 큰 프레임 쌍들을 먼저 찾아낸 후, 이런 프레임 간에만 정확히 연결성 계산을 다시 하여 그래프를 생성한다. 이와 같이, 비용 계산을 모든 프레임에 대해서 하는 것이 아니라 미리 가능성이 큰 에지들을 선별하여 계산하기 때문에 그래프를 생성하는 시간을 효과적으로 줄일 수 있다.

2. 관련연구

새로운 모션을 얻는 과정에서 모션 캡쳐 데이터들을 이용한 연구들은 지금껏 많이 있어 왔다. Liu[2]는 사용자가 원하는 역동적인 모션을 만들어내기 위해, 상대적으로 정적이고 간단한 모션들의 집합으로부터 constrained state와 unconstrained state 사이의 전이 자세를 예측하고, 운동량을 조절하여 합성하였다. Ikemoto[5]는 데이터베이스에 있는 모션들에 대해서 몸의 일부분-팔, 다리 몸통 등-을 서로 잘라 붙이는 방법으로 새로운 모션을 만들어내었다. Pullen[4]은 사용자가 입력한 간단한 키프레임 위에 다 모션 캡쳐 데이터들을 이용하여 사실적인 모션의 뉘앙스를 더해주어 애니메이션을 만들어내었다. Li[9]는 데이터들을 LDS(Linear dynamic system)로 만들어진 각각의 *motion texton*들로 나누고, 하나의 *texton*으로부터 다음 *texton*으로 연결 가능성을 고려하여 모션을 합성하였다.

모션 집합으로부터 새로운 모션을 얻고자 하는 연구에 있어, 모션 데이터 집합을 그래프의 형태로 저장하고 이것을 검색하여 모션을 재구성하는 방식의 연구들이 있다. Lee[3]는 여러 인터페이스를 통해 사용자가 3차원 캐릭터를 컨트롤하도록 하는 어플리케이션을 제안하였다. 이를 위하여, 각 프레임을 나타내는 노드가 정해진 깊이만큼의 트리구조를 가지도록 하고, 이 트리들의 집합으로 그래프를 구성하였다. Arikan[8]은 사용자로부터 모션의 길이, 주어진 시간에서의 자세를 결정하는 키프레임 등의 제한조건을 입력 받아 데이터베이스 검색을 통해 알맞은 모션을 합성하였다. 그래프는 연결 가능한 모션 간의 에지들을 계층적 구조로 만들고, 이를 임의검색방식을 통해 모션을 합성하였다. 이런 연구들은 공통적으로 모든 모션의 모든 프레임간의 비용 계산 후 그래프를 만드는 방식을 택하고 있어 규모가 큰 모션 데이터베이스를 만들 경우 많은 계산 시간을 필요로 한다.

풀기가 쉽지 않은 문제를 해결하고자 할 때, 차원을 낮추어 접근하면 쉽게 해결될 수 있다. 특히 사람과 같이 복잡한 캐릭터의 경우 DOF(degree of freedom)의 수를 줄여서 계산하는 것이 그 경우일 것이다. Safonova[1]는 대부분의 사람 모션이 적은 수의 DOF로도 충분히 표현될 수 있다는 관찰을 바탕으로, 저차원에서 최적화를 수행하여 물리적으로 자연스러운 모션을 만들어냈다. 모션의 차원을 줄이기 위해서, PCA(principal component analysis)를 이용하여 각 프레임에서 모든 관절들의 각 값을 원래의 공간에 대해 자처원의 선형적 부공간에서 나타내었다. 이때, 원래의 모션과 저차원에서의 모션간 차이를 최소화하면서 부공간을 표현할 수 있는 베이시스를 찾아내게 된다. 이렇게 차원을 낮추어 계산하게 되면, 비교적 쉽고 간단하게 원하는 결과를 얻을 수 있다는 장점이 있다.

3. 프레임 간 연결 가능의 필요 조건

기존의 연구들을 보면 모션그래프 생성을 위해 모든 모션의 모든 프레임 간의 비용 계산을 하고 있다. 또한, 프레임간 비용계산에 있어 일반적으로 관절들의 위치와 속도 차이를 주요하게 고려하는 공통점이 있다. 이 장에서는, 이런 연구들 중 Lee[3]의 연구를 비롯한 기존 연구들의 연결성 계산을 위한 차이계산 함수에 대해 살펴본다.

Lee[3]는 i에서 j로 가는 전이는 j의 이전 프레임에서부터 j로 변화하는 관계와 연관된다는 first-order markov model을 바탕으로 하여 그래프를 만들었는데, 이를 위해 사용하는 차이계산 식은 다음과 같다.

$$D_{i,j} = d(p_i, p_j) + v d(v_i, v_j) \tag{1}$$

여기서, $d(p_i,p_j)$는 i프레임과 j프레임 사이의 관절들의 각 차이에 가중치를 준 값과 몸통(root)의 위치 차이를 합산한 값이고, $d(v_i,v_j)$는 i, j프레임 간의 관절들의 속도 차이에 가중치를 준 값과 몸통의 속도 차이를 합한 값이다. 파라미터 v는 위치 차이 항에 대한 속도 차이 항의 가중치이다.

Arikan[8] 역시 Lee[3]가 정의한 것과 마찬가지로 차이계산함수를 관절의 위치 차이를 비롯하여 몸통의 위치 차이와 회전 차이를 고려하여 정의하였다. Kovar[6]의 경우는 캐릭터 위의 점들을 샘플링하고, 이들간의 3차원 위치 차이를 이용하여 연결비용함수를 정의하였다. 이것 역시, 샘플링 된 점들이 관절각의 변화를 반영한 것이라 할 수 있다.

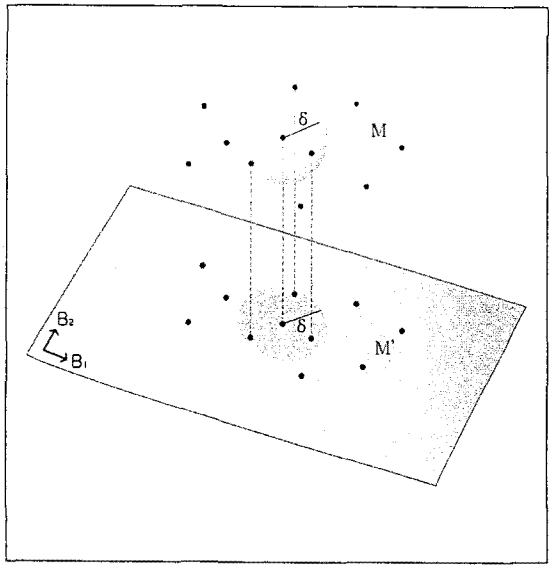
기존 연구들에서 본 것처럼 동작의 연결성을 나타내는 함수에서 관절각의 차이는 중요한 항으로 더해지게 된다. 이 사실을 통해, Lee[3]의 연결비용함수로부터 다음과 같은 관계를 알 수 있다.

$$\begin{aligned} D_{i,j} &= d(p_i, p_j) + v d(v_i, v_j) \\ &\geq d(p_i, p_j) \\ &\geq \| p_{i,0} - p_{j,0} \|^2 + \sum_{k=1}^{m} w_k \| \log(q_{i,k}^{-1} q_{j,k}) \|^2 \\ &\geq \| q_i - q_j \|_w^2 \end{aligned} \tag{2}$$

위 관계를 통해, 프레임 i와 j사이가 $D_{i,j} \leq \delta$ 인 경우에 대하여 전이 가능한 에지로 판단한다면, 관절각의 차이 $\|q_i - q_j\|_w^2$가 δ 이상인 에지들은 전이 불가능으로 판단할 수 있다는 것을 알 수 있다. 따라서 $\|q_i - q_j\|_w^2 \leq \delta$는 $D_{ij} \leq \delta$의 필요 조건이라고 할 수 있다. 이 필요 조건을 만족하는 프레임 쌍을 연결 가능한 후보 에지로 선별한다면 이들에 대해서만 $D_{i,j}$ 계산을 하면 되기 때문에 계산량을 많이 줄일 수가 있다. 이것이 본 연구의 효율적인 모션 그래프 생성 알고리즘이 바탕으로 하는 핵심 아이디어이다.

4. 모션그래프 생성방법

본 논문에서 그래프는 모션의 각 프레임을 나타내는 노드, 그리고 이 노드 간에 가능한 연결관계를 나타내는 에지, 즉 전이에지로 구성된다. 노드는 어느 모션의



<그림1> 2차원에 투영하기

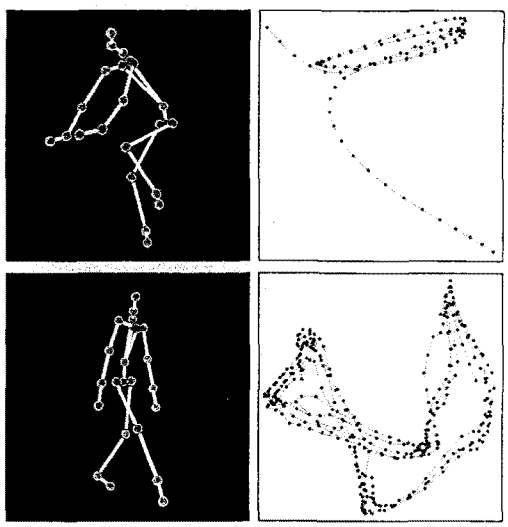
어느 프레임인지의 정보와, 노드로부터 나가는 에지의 집합들로 이루어진다. 에지는 에지 양쪽의 노드와 이들 노드 사이의 연결 비용을 저장한다.

전이에지는 하나의 노드를 기준으로 일정 비용(δ) 이하의 관계에 있는 다른 노드와 연결을 지어주는 것이다. 모션의 DOF가 n이라면, 노드들은 n차원상의 점이라고 볼 수 있다. 이 n차원 상의 점에 대하여 식 (2)의 관계를 통해 도출한 전이 에지가 되기 위해서는 두 노드 간의 거리가 반드시 δ보다 작아야 한다. 이 것은 n차원 상에서 모든 노드에 대한 범위 탐색 문제라고 할 수 있다. 그런데, 일반적인 n차원에서의 범위 탐색 문제는 효율적으로 구현하기가 어렵고 n에 비례해서 시간 복잡도의 차수가 증가하는 문제가 있다. 그래서 본 연구에서는 n차원 범위 탐색 문제를 2차원 문제로 단순화하는 방법을 사용하였다. 2차원 문제로 바꾸고 나면 거리 계산과 범위탐색이 n차원에 비해 알고리즘 구현이 간단하고 빠른 결과를 얻을 수 있다. 그런데, 2차원으로 문제를 단순화하면 이 과정에서 생기는 정보의 손실로 인해 정확한 n차원 범위 탐색 결과보다 더 넓은 범위의 노드들이 포함되는 문제가 있다. 하지만, 이러한 노드들은 이후에 적용되는 D_{ij}계산에서 제거되기 때문에 최종 결과는 달라지지 않고, 단지 후보 에지들의 수만 증가시키게 된다.

2차원 문제로의 단순화는 간단히 동작 데이터를 2차원에 투영시키는 방법을 사용하였다. <그림1>에서와 같이, n차원에서 δ 안에 있는 노드들은, 2차원에 내렸을 때에도 마찬가지로 δ 안에 있게 된다. 즉,

$$Range^n(p,\delta) \subseteq Range^2(p,\delta)$$

의 관계를 가지게 된다. 따라서, 2차원 범위 탐색을 할 경우 연결되어야 할 노드들이 빠지는 경우(false negative)



<그림2> 자전거 타는 모션(왼쪽 위)과 발레걸음 모션(왼쪽 아래)을 각각 2차원에 투영시킨 결과(오른쪽 위, 오른쪽 아래)

는 발생하지 않는다. $Range^2(p,\delta)$에 있는 노드들은 불필요한 노드들도 같이 포함하고 있기 때문에, 다시 $D_{ij} \le \delta$ 테스트를 하여 통과하는 노드들을 최종적으로 그래프의 에지로 노드 p와 연결하여 준다.

2차원 투영을 할 때, 평면에 잘 fitting이 잘 되지 못하는 경우 선별되는 후보 노드의 개수가 많아지므로 본 논문에서 얻고자 하는 시간절약효과가 크지 못할 수 있다. 그러나, Safonova[1]는 대부분의 사람모션은 낮은 차원의 모션으로 적절히 나타낼 수 있고, 특히 비슷한 모션들의 경우 관절 각들의 상호연관성이 높아 저차원 상에서도 원래 모션에 대해 비교적 오차가 작다는 것을 실험적으로 증명하였다. 이 사실에서 알 수 있듯이 2차원에서 범위탐색을 하더라도 충분히 효과를 볼 수 있는 것이다. 따라서, 본 논문에서 하고자 하는 선별방법은 시간적 효과를 얻으면서도 정확한 모션 그래프를 얻을 수 있다.

4.1 2차원에 투영하기

각 모션은 프레임의 연속으로 구성되고, 각 프레임은 모든 관절들의 각들로 이루어져 있다. 모션의 프레임 수가 m, 캐릭터의 DOF가 n, 각 프레임의 자세를 $p_i = \{q_1, q_2 \dots q_n\}$로 나타내면 자세들의 집합, 즉 모션 M은 다음과 같이 나타낼 수 있다.

$$M = \{p_1, p_2 \dots p_m\}$$

DOF가 n이므로, M는 n차원 공간의 점들이다. 이런 점들에 대하여 <그림1>과 같이 단위 길이이면서 서로 수직인 벡터 B_1, B_2로 이루어진 2차원 선형 부공간를 생

각해 보자. 모션 M'는 이 2차원 공간을 이루는 베이시스 벡터인 B_1, B_2의 선형조합으로서 근사값을 구할 수 있을 것이다.

$$M' = B_1 A_1 + B_2 A_2$$

여기서, A_1, A_2는 계수이고, B_1, B_2는 PCA를 이용하여 구할 수 있다. PCA를 사용하면 서로 연관된 데이터들이 서로 연관되지 않은 성분들로 분리될 수 있는 새로운 좌표공간을 찾을 수 있다. 따라서, PCA로 데이터들이 가장 넓게 분포된 크기에 따라 서로 수직을 이루는 주축들을 순서대로 2개 구하고, 이 2개의 축으로 이루어진 평면에 투영을 시킨 점들이 바로 M'인 것이다. 이때, A_1, A_2는 해당 노드의 B_1, B_2축에 대한 성분이 된다. PCA는 주어진 데이터들에 대하여 공분산 행렬을 구한 후, 이 행렬에 대해 고유벡터를 구하는 과정으로 이루어져 있다. N개의 데이터(p)에 대해서 평균이 m이라고 할 때, 공분산 행렬은 다음과 같이 구할 수 있다.

$$C = \frac{1}{N}\sum (p_i - m)(p_i - m)^T \qquad (2)$$

이 값은 데이터베이스에 새로운 모션이 추가되어 새롭게 계산이 필요한 경우에, PCA과정을 위하여 매번 다시 계산되어야 할 것이다. 그러나 이 공분산 행렬을 식 (3)과 같이 Garland[7]가 말한 *fit Quadric* 형태로 나타내면, 계산상의 부하를 줄일 수 있다. 이는, 처음 한번 계산 후 행렬 A와 벡터 b만 저장하고 있으면 매번 (2)식을 다시 계산하지 않아도 되기 때문이다.

$$\begin{aligned} C &= \sum p_i p_i^T - k(\overline{pp})^T \\ &= A - \frac{bb^T}{c} \end{aligned} \qquad (3)$$

여기서, k는 프레임 수, $A = p_i p_i^T$, $b = p_i$ 그리고 c는 상수값이다.

4.2 연결 가능한 노드쌍 구하기

모션 데이터베이스는 n개의 DOF를 가진 같은 구조의 여러 모션들과 이들 모션 사이의 연결관계를 나타내는 그래프로 이루어진다.

그래프 생성은 초기에 노드를 전혀 포함하지 않은 빈 그래프로 시작한다. 데이터베이스에 있는 모션들을 두 개씩 짝지어 전이에지를 찾는 과정은 <표1>과 같다. <표1>에서, (1)번 줄의 $Range^2(M,\delta)$ 모션 데이터 M 을 2차원 평면에 투영한 후 모든 노드들에 대한 범위 탐색 계산을 하여 $Range^2(i,\delta)$를 구한다. 다음은 각 노드 i마다 $Range^2(i,\delta)$에 들어 있는 노드 j와 D_{ij}을 계산하여 δ 안에 들어 오는 경우에 새로운 에지를 생성하여 그래프에 추가한다.

<표1>전이에지를 찾는 알고리즘

```
// For motions stored in M
(1) Find Range²(M,δ)
(2) For each node i ∈ M
(3)   For each node j ∈ Range²(i,δ)
(4)     If D_ij ≤ δ then
(5)       Make a new edge e = (i, j, D_ij)
(6)   end
(7) end
```

$Range^2(M,\delta)$을 계산하기 위해서 본 연구에서는 두 가지 방법을 시도하였다. 첫 번째는 간단히 모든 노드 쌍에 대하여 2차원 거리를 계산한 후 δ 안에 들어오는 노드 쌍을 찾았다. 이 방법은 $O(n^2)$의 시간이 걸린다. 두 번째는 보다 효율적인 방법으로 plane sweep을 이용하였다.

Plane sweep 방법은 평면의 y축을 따라 scanning 하면서 각 노드의 y축 범위가 시작하는 순간과 끝나는 순간들을 이벤트로 처리하여 y축 범위 탐색을 수행한다. Y축 범위의 시작에서는 Active_List에 그 노드의 x축 범위의 시작과 끝 점을 추가하여 x축 범위 탐색을 수행한다. Y축 범위의 끝에서는 Active_List에서 해당 노드의 x축 범위를 제거한다. 각 노드 $n_i=(x_i, y_i)$에 대하여 x축 범위는 $[f_{i,x}, t_{i,x}]=[x_i-\delta, x_i+\delta]$, y축 범위는 $[f_{i,y}, t_{i,y}]=[y_i-\delta, y_i+\delta]$ 로 주어진다. 구체적인 내용은 <표2>에 나와 있다.

<표2>범위탐색 알고리즘

```
List_Y ← Sort f_y, t_y for all nodes on y-axis
Do while List_Y is not NULL
  take out the smallest element e from List_Y
  If e = t_y
    remove (f_x,t_x) of node of e from Active_List
  else
    insert (f_x,t_x) of node of e to Active_List
    find node-pair of e in Active_List
  end
end
```

5. 실험 결과

우리는 본 연구에서 제시한 알고리즘과 기존의 연구간의 효율성을 비교해보기 위해, 몇 가지 모션에 대하여 그래프 생성 시간을 측정하는 측정 실험을 하였다.

<표3> 모션에 대한 정보

	모션1	모션2	모션3	모션4
DOF	18	18	18	29
프레임 수	350	835	789	774
모션의 특징	발레 걸음걸이로 직선상으로 움직이는 모션	속도를 달리하며 살금살금 직선으로 걷는 모션	일반적인 걸음걸이로 직선상으로 움직이는 모션	걷는 속도와 움직이는 패스가 다양하게 변화하는 다이나믹한 모션

<표4> 시간측정 결과(δ = 0.05 일 때, 단위는 초)

	모션1	모션2	모션3	모션4
생성된 에지 개수	2896	3259	61799	2156
All-pairs test	6.18	35.28	31.38	54.12
PAP	0.85	5.05	9.72	6.30
PRS	0.43	2.51	4.92	3.16

<표5> 시간측정 결과(δ = 0.1 일 때, 단위는 초)

	모션1	모션2	모션3	모션4
생성된 에지 개수	6952	6203	126357	4156
All-pairs test	6.21	34.40	31.42	54.21
PAP	2.38	7.06	14.45	8.70
PRS	1.19	3.50	7.36	4.39

<표6> 시간측정 결과(δ = 0.3 일 때, 단위는 초)

	모션1	모션2	모션3	모션4
생성된 에지 개수	18492	30109	404681	23166
All-pairs test	6.17	34.47	31.91	54.29
PAP	2.51	12.46	24.10	14.70
PRS	1.27	6.25	12.30	7.34

본래 모션 데이터베이스의 그래프를 생성할 경우에는 다양한 모션 간에 연결관계를 짓게 된다. 그러나, 우리의 실험에서는 알고리즘 성능을 비교 테스트 하는 것이므로, 모션의 프레임 수와 캐릭터의 자유도를 달리하여 각각 따로 그래프를 생성하도록 하는 측정방법을 택하였다. 사용한 모션은 <표3>에서 보이는 것을 사용하였고, 전이에지를 판별하는 연결비용의 한계값(δ)을 달리하여 다음의 3가지 방법에 대해 측정하였다.

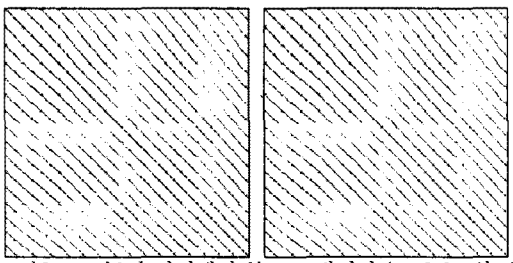
- **All-pairs test**: 기존의 그래프 생성 알고리즘으로, 일반적으로 모든 노드 쌍에 대하여 관절들의 각 차이, 속도 차이, 몸통의 위치 차이 등을 고려한다.
- **PAP(projected all-pairs test)**: 2차원에 투영 후, 모든 노드 쌍에 대해 거리 비교로 그래프를 만든다.
- **PRS(projected range search)**: 본 연구에서 제안한 방법으로서 2차원에 투영 후 범위탐색 방식으로 그래프를 만든다

모든 값은 Pentium4 2.53GHz, 1GB RAM의 PC환경에서 측정하였다.

<표4>, <표5>, <표6>에서와 같이 본 연구에서 제안한 방법은 수행속도 면에서 기존의 연구에 비해 현저한 성능 향상을 나타내었다. 총 프레임 수가 n이고 최종적으로 얻어지는 전이에지 개수가 e라고 했을 때, 기존의 연구(All-pair test)는 $O(t_a*n^2)$의 수행시간이 걸리는데 반해, PAP는 $O(t_b*n^2+ t_a*e)$, PRS는 $O(t_c*n\log n + t_a*e)$의 수행시간이 걸린다.(여기서, t_a는 하나의 에지에 대한 연결비용 계산시간, t_b는 2차원 거리 계산에 드는 시간, t_c는 범위 탐색에 드는

시간이다.) 모든 테스트에 있어서 All-pair test에 비해 PAP와 PRS의 수행시간이 적은 것은, 2차원 투영의 효과를 증명하고 있다. 같은 2차원 투영방법이라도 PRS가 PAP보다 2배정도 우수한 성능을 나타내었는데, 이는 범위탐색의 효율을 보여주는 결과이다. 또한, 모션1과 모션2를 비교해 보면, 2차원에 투영시키는 방법의 경우 프레임 수의 영향을 아주 적게 받으면서 빠른 결과를 낸다는 사실을 알 수 있다. 따라서, 모션 데이터베이스에 모션이 많이 들어있으면서 생성에지 수가 적은 경우에 본 논문에서 제시한 방법은 아주 큰 효과를 기대할 수 있다. 본 연구에서 제안한 방법은 사람구조의 자유도(DOF)에도 크게 영향을 받지 않음을 볼 수 있다.(모션3-모션4)

모션4의 경우, 2차원에 투영시킨 방법으로 얻은 전이에지 개수가 기존 연구에서 사용한 방법(All-pair test)으로 찾은 수보다 적은 경우가 있었다. 이는 사원수(Quaternion)의 등가대척점(antipodal equivalence)에 대해 어떤 값을 선택하느냐에 따라 노드들의 2차원 상의 위치가 달라지기 때문이다.



<그림3> 모션3의 전이에지 분포 그림이다.(δ=0.05 일 때) 왼쪽과 오른쪽은 각각 All-pair test와 PRS로 얻은 결과로, 양쪽모두 에지 개수가 61799개로, 분포모양 역시 차이가 없다.

6. 결론

본 논문은 모션그래프의 효과적 생성 방법을 제시하였다. 모든 모션의 모든 프레임간 비용계산 대신, 연결가능성이 큰 에지를 선별하고 이에 대해서만 실제 모션 간 비용계산을 함으로써 효율을 높였다. 이때, 가능성 있는 에지 선별을 위하여, 모션의 차원을 낮추어 2차원 상에서 차이를 계산하고, 범위탐색 방식을 사용함으로써 성능을 높였다. 2차원에서는 n차원에 비해 거리계산과 범위탐색이 간단하고 빠르기 때문에 많은 시간단축 효과를 볼 수 있었다.

본 연구의 방법은 linear dimension reduction 방식인 PCA를 사용하여 2차원 평면에 투영시켰다. 투영 방법으로 nonlinear dimension reduction 방식을 사용하면, 투영오차가 줄어들어 선별되는 에지들을 좀 최적에 가깝게 얻을 수가 있다. 그러나 nonlinear 방식은 투영과정에 많은 계산을 요구하기 때문에 전체적인 그래픽 생성 시간을 심각하게 증가시키는 문제가 있다.

등가대척점(anti-podal equivalence) 문제로 인하여 선별되어야 할 에지들이 누락되는 문제에 대해서는, 이런 노드들은 두 개의 대척점을 가지는 두 개의 노드로 만들어서 그래프를 만든 후, 이 두 노드를 다시 하나로 병합하는 방법으로 해결할 수 있다. 그러나, 이런 문제를 발생시키는 노드들이 어떤 조건의 노드들인지 효율적으로 찾아내는 연구가 필요하다.

감사의 글

본 연구는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 육성 지원 사업의 연구 결과로 수행되었습니다.

참고문헌

[1] Alla Safonova , Jessica K. Hodgins and Nancy S. Pollard, Synthesizing Physically Realistic Human Motion in Low-Dimensional, Behavior-Specific Spaces. *Proceedings SIGGRAPH 2004*

[2] C. Karen Liu and Zoran Popović. Synthesis of Complex Dynamic Character Motion from Simple Animations. *Proceedings SIGGRAPH 2002*

[3] Jehee Lee, Jinxiang Chai and Paul S. A. Reitsma, Interactive control of Avatars Animated with Human Motion Data, *Proceedings SIGGRAPH 2002*

[4] Katherine Pullen, Christoph Bregler. Motion Capture Assisted Animation: Texturing and Synthesis. *Proceedings SIGGRAPH 2002*

[5] Leslie Ikemoto and David A. Forsyth. Enriching a motion collection by transplanting limbs. *The Eurographics Association 2004*

[6] Lucas Kovar, Michael Gleicher and FrédéricPighin. Motion graphs. *Proceedings SIGGRAPH 2002.*

[7] Michael Garland, Andrew Willmott, Paul S. Heckbert. Hierarchical face clustering on polygonal surfaces. *Proceedings of the 2001 symposium on Interactive 3D graphics 2001* [8] Okan Arikan and D. A. Forsyth. Interactive motion generation from examples. *Proceedings SIGGRAPH 2002*

[9] Yan Li, Tianshu Wang, and Heung-Yeung Shum. Motion texture: A two-level statistical model for character motion synthesis. *Proceedings SIGGRAPH 2002*