모션 데이터를 이용한 3차원 아바타 얼굴 표정 제어

김 성 호† · 정 문 렬††

요 약

본 논문은 사용자로 하여금 얼굴표정들의 공간으로부터 일련의 표정을 실시간적으로 선택하게 함으로써 3차원 아바타의 얼굴 표정을 제어하는 기법을 제안하고, 해당 시스템을 구축한다. 본 시스템에서는 얼굴 모션 캡쳐 데이터로 구성된 2400여개의 표정 프레임을 이용하여 표정공간을 구성하였다. 본 기법에서는 한 표정을 표시하는 상태표현으로 얼굴특징 점들 간의 상호거리를 표시하는 거리행렬을 사용한다. 이 거리행렬의 집합을 표정공간으로 한다. 그러나 이 표정공간은 한 표정에서 다른 표정까지 이동할 때 두 표정간의 직선경로를 통해 이동할 수 있는 그런 공간이 아니다. 본 기법에서는 이 경로를 표정 데이터로부터 근사적으로 유추한다. 우선, 각 표정상태를 표현하는 거리행렬간의 거리가 일정 값 이하인 경우 두 표정을 인접해 있다고 간주한다. 임의의 두 표정 상태가 일련의 인접표정들의 집합으로 연결되어 있으면 두 표정 간에 경로가 존재한다고 간주한다. 한 표정에서 다른 표정으로 변화할 때 두 표정간의 최단경로를 통해 이동한다고 가정한다. 두 표정간의 최단거리를 구하기 위해 다이내믹 프로그래밍 기법을 이용한다. 이 거리행렬의 집합인 표정공간은 다차원 공간이다. 3차원 아바타의 얼굴 표정은 사용자가 표정공간을 항해하면서 원하는 표정을 실시간적으로 선택함으로써 제어한다. 이를 도와주기 위해 표정공간을 다차원 스케일링 기법을 이용하여 2차원 공간으로 가시화했다. 본 시스템이 어떤 효과가 있는지를 알기 위해 사용자들로 하여금 본 시스템을 사용하여 3차원 아바타의 얼굴 표정을 제어하게 해본 결과, 3차원 아바타의 실시간 얼굴 표정 제어가 필요한 각 분야에서 매우 유용하게 사용될 것으로 판단되었다.

Facial Expression Control of 3D Avatar using Motion Data

Sung-Ho Kim† · Moon-Ryul Jung††

ABSTRACT

This paper propose a method that controls facial expression of 3D avatar by having the user select a sequence of facial expressions in the space of facial expressions. And we setup its system. The space of expression is created from about 2400 frames consist of motion captured data of facial expressions. To represent the state of each expression, we use the distance matrix that represents the distances between pairs of feature points on the face. The set of distance matrices is used as the space of expressions. But this space is not such a space where one state can go to another state via the straight trajectory between them. We derive trajectories between two states from the captured set of expressions in an approximate manner. First, two states are regarded adjacent if the distance between their distance matrices is below a given threshold. Any two states are considered to have a trajectory between them if there is a sequence of adjacent states between them. It is assumed that one states goes to another state via the shortest trajectory between them. The shortest trajectories are found by dynamic programming. The space of facial expressions, as the set of distance matrices, is multidimensional. Facial expression of 3D avatar is controled in real time as the user navigates the space. To help this process, we visualized the space of expressions in 2D space by using the multidimensional scaling(MDS). To see how effective this system is, we had users control facial expressions of 3D avatar by using the system. As a result of that, users estimate that system is very useful to control facial expression of 3D avatar in real-time.

키워드 : 얼굴 모션 캡쳐(Facial Motion Capture), 얼굴 표정 제어(Facial Expression Control), 표정 상태표현(Facial State Representation), 거리 행렬(Distance Matrix), 다이내믹 프로그래밍(Dynamic Programming), 다차원 스케일링(MDS)

1. 서 론

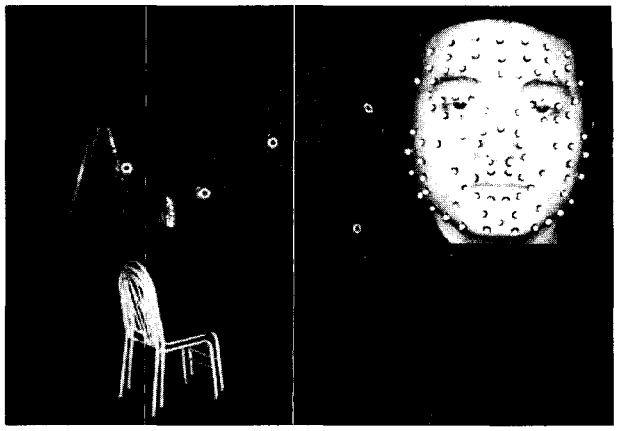
인간은 자신의 감정을 말보다는 얼굴 표정을 통해서 더 잘 표현하며, 상대방의 얼굴 표정을 보고 상대방의 현재 감정 상태를 파악한다. 그런 연유로 지금까지 3차원 컴퓨터 그래픽스 기법을 통해서 인간의 얼굴 표정을 표현할 수 있는 많은 방법이 연구[1-6]되었다. 특히 인터넷의 보급이 활발해지고 멀티미디어의 기능이 고급화됨으로써 인터넷과 같은 가상공간에서 사용자 자신을 대신하는 3차원 아바타가 등장하게 되었으며, 이제는 많은 멀티미디어 분야에서 사용되고 있다. 또한 가상공간에서 3차원 아바타를 표현할 때 가장 관심을 가지는 것은 역시 3차원 아바타의 얼굴 표정이다. 왜냐하면 인간은 표정의 변화를 보고 가장 빠르게 상대방의 감

※ 본 논문은 과학재단 특정기초 연구과제(R01-2002-000-00311-0)의 수행결과의 일부임.
† 준 회 원 : 숭의여자대학교 정보통신계열 교수
†† 정 회 원 : 서강대학교 미디어공학과 교수
논문접수 : 2004년 3월 17일, 심사완료 : 2004년 7월 1일

정을 파악하고, 아울러 자신의 감정을 실시간으로 상대방에게 보여주고 싶어 하기 때문이다. 이에 가상공간에서 3차원 아바타의 표정을 실시간적으로 사용자가 제어할 수 있는 기법이 필요하다. [6]에서는 3차원 아바타의 표정을 제어하는 기법을 제안하였는데, 서로 다른 6개의 표정을 3D 얼굴 모델을 사용하여 표현하고 표정 합성을 위한 파라미터 공간을 형성한다. 파라미터 공간은 각 표정들을 키 프레임으로 설정하고, 각 키 프레임들을 보간하여 다양한 표정들을 생성할 수 있도록 구성하였다. 그리고 사용자가 미리 준비된 Cubic 스플라인(Spline) 곡선을 파라미터 공간에서 임의로 변경하고 재생하면 보간법에 의하여 새로운 표정이 생성되게 된다. 그러나 이 기법에서는 사용자가 파라미터 공간에서 스플라인 곡선을 생성할 때 실시간적으로 표정이 보여 지지 않는다. 그런 이유로 본 논문의 목적인 가상공간에서 3차원 아바타의 표정을 실시간으로 제어하기 위한 조건을 만족하지 못한다. 그러므로 3차원 아바타의 표정을 실시간적으로 제어하기 위한 새로운 방법이 필요하며, 본 논문에서는 실시간적으로 자연스러운 표정을 제어하는 방법으로 모션 캡쳐 기법을 제안한다. 최근 [16]에서는 얼굴 모션 캡쳐 데이터를 사용한 사용과 재사용에 대하여 제안하였는데, 사용자의 최소한의 도움으로 모션 캡쳐 데이터로부터 면으로 형성된 3차원 아바타의 얼굴 메쉬를 애니메이션하기 위한 방법이다. 그러나 이 기법은 얼굴 모션 캡쳐 데이터를 새로운 얼굴에 재적용하기 위한 리타겟팅 기법을 강조하고 있기 때문에, 본 논문의 목적인 실시간 표정 제어와는 거리가 멀다.

모션 캡쳐 데이터를 사용한 표정 제어는 배우의 표정을 가능한 많이 캡쳐하여 표정 데이터베이스를 만든 후, 사용자가 특정 표정들을 실시간적으로 선택하여 보여주는 것이다. 그러므로 본 논문에서는 다량의 얼굴 모션 캡쳐 데이터를 직관적인 공간에 분포시키고, 사용자가 적당한 공간을 항해하면서 원하는 얼굴 표정들을 실시간적으로 선택하여, 3차원 아바타의 얼굴 표정을 실시간으로 제어하는 방법을 기술한다.

얼굴 표정 데이터를 사용한 3차원 아바타의 표정 제어를 위해서는 다음과 같은 사전 작업이 필요하다. 먼저 얼굴 표정을 전문적으로 연출하는 배우의 도움을 받아 (그림 1)과 같이 광학식 모션 캡쳐 시스템을 사용하여 얼굴 표정을 캡쳐한다. 표정을 캡쳐할 때, 배우는 얼굴 주 근육 부분에 작은 반사 마커 100개를 부착한다. 그런 다음 배우로 하여금 서로 다른 10개의 얼굴 모션을 연출하게 하고 초당 60 프레임으로 캡쳐한다. 모든 얼굴 모션은 처음 무표정 상태에서 출발하여 특정 표정까지 진행한 다음, 다시 무표정 상태로 되돌아오는 방법으로 진행한다. 한 개의 마커는 3개의 좌표값으로 표현되므로 100개의 마커 위치로 표현되는 한, 표정은 300차원의 데이터이다.



(그림 1) 마커 100개 부착위치 및 광학식 모션캡쳐 시스템을 사용한 얼굴모션캡쳐 장면

그리고 얼굴 모션 캡쳐 작업을 수행한 배우의 정면 사진을 이용하여 3차원 얼굴 모델링 작업을 한다. 또한 100개 마커를 3차원 아바타의 얼굴 각 부위(모션 캡쳐시 마커 부착 위치)에 연결하여 표정 재생이 가능하게 한다. 본 논문에서는 이를 미리 구축하였으며, 이를 기반으로 실험한다. 본 논문에서는 얼굴 표정 데이터에서 임의의 두 표정간의 최단경로를 결정함으로써 표정공간을 생성하고, 이 공간을 다차원 스케일링(Multidimensional Scaling, MDS)[7,8,11] 기법으로 2차원으로 투영시킨다. 3차원 아바타 얼굴 표정을 실시간으로 제어하기 위해서는 애니메이터로 하여금 투영된 2차원 공간을 항해하게 하고, 항해경로의 각 점에 해당되는 표정을 3차원 얼굴 모델을 통하여 실시간으로 디스플레이 한다.

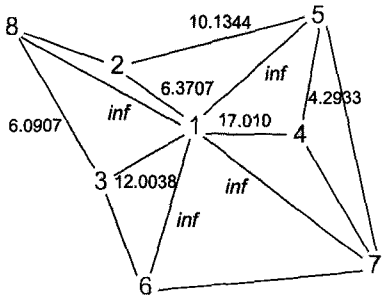
2. 얼굴 표정상태 표현법

표정공간을 생성하기 위해서는 각각의 얼굴 표정상태(Facial state)를 수치적으로 표현해야 한다. 표정상태는 얼굴에 부착된 마커 위치들에 의해 결정된다. 표정상태의 표현은 표정들 간의 상대적인 거리 관계를 잘 표현하는 것이어야 한다. 표정의 상태를 표현하는 가장 간단한 방법은 마커 위치들로 이루어진 상태벡터를 이용하는 것이다. 본 논문에서는 100개의 마커를 사용하고 한 마커는 3개의 좌표를 가지기 때문에, 표정상태벡터는 300차원이 된다. 이런 식으로 표정상태를 표현하는 방법을 '*위치벡터방식*'이라고 하자. 본 논문에서는 표정상태를 표현할 때, 위치벡터방식을 사용하지 않는다. 대신 임의의 두 마커간의 상호거리를 표현하는 "*거리행렬*"을 이용하여 표정상태를 표현한다. 거리행렬이 위치벡터보다 얼굴 마커들의 분포상태에 대한 정보를 더 많이 표시하고 있고, 따라서 두 표정간의 거리를 보다 더 정확하게 표현할 수 있다. 두 위치벡터를 비교할 때, 각 마커의 위치들 간의 거리를 구함으로써 두 표정간의 거리를 표현한다. 이때 각 마커의 표정간의 거리는 다른 마커들과 관계없이

계산한다. 이에 비해, 거리행렬로 표정상태를 표현하면, 두 표정간의 거리를 구할 때 마커들 간의 상호관계가 자연스럽게 고려된다. 물론 표정의 거리행렬은 표정의 위치벡터로부터 구할 수 있지만, 위치벡터를 사용하는 경우, 이 위치벡터에 내포되어 있는 정보를 상태간의 거리를 구하거나 할 때 사용하지 않기 때문에, 유용한 정보를 내포하고 있다는 것이 도움이 되지 않는다. 따라서 이 정보를 명시적으로 표현하는 거리행렬 방식이 더 좋은 상태표현방식이다.

3. 표정공간의 생성

표정상태를 거리행렬로 표현하면 표정간의 직선거리는 두 거리행렬간의 직선거리로 표현된다. 거리행렬로 표현된 표정상태들의 공간은 임의의 두 거리행렬간의 거리를 결정함으로써 결정된다. 본 논문에서는 거리행렬을 하나의 벡터로 보고, 이 벡터간의 직선거리를 거리행렬간의 직선거리로 사용한다. 즉 임의의 두 마커간의 거리가 서로 비슷한 두 표정은 서로 인접한 표정으로 간주한다. 표정공간은 임의의 두 표정간의 거리를 두 표정거리 행렬간의 직선거리로 정의할 수 있는 벡터공간이 아니다. 한 표정에서 다른 표정으로 옮아가는 과정은 얼굴의 여러 가지 제약조건으로 말미암아, 복잡한 경로를 거치게 되기 때문이다. 표정공간은 구면과 같은 다양체(Manifold) 공간인 것이다. 다양체 공간상에서의 거리는 두 점간의 거리를 한 점에서 이 공간을 벗어나지 않으면서 다른 점까지 도달하는 최단경로의 길이로 정의[10]한다.



(그림 2) 플로이드 알고리즘을 위한 그래프 생성방법 : 모두 8개의 표정상태가 주어졌다고 보고, 1번 표정상태를 기준으로 2~4번 표정상태는 1번 표정상태와 인접해 있고, 5~8번 표정상태는 인접해 있지 않다. 인접표정이 아닌 두 표정상태간의 거리는 무한대(inf)로 주어져 있는데, 이것은 한 표정상태에서 다른 표정상태까지 바로는 이동할 수 없다는 것을 의미한다. 원의 반지름은 인접거리 한계 값을 나타낸다.

본 논문에서는 이 다양체 공간을 근사적으로 표현한다. 이를 위해 먼저, 두 거리행렬간의 직선거리가 일정 값 이하인 경우 이 직선거리가 두 표정간의 최단거리에 대한 근사치라고 간주한다. 이 조건을 만족하는 두 표정을 "*인접표정*"이라고 하는데, 임의의 표정에 대한 인접표정들은 (그림 2)에서 보는 것처럼 결정한다. 여기서 표정상태들의 각 경로간의 값은 표정상태들 사이의 거리를 의미한다. 인접표정이 주어지면 한 표정에서 다른 표정까지 바로 이동할 수 있다고 본다. 두 표정이 인접해 있지 않은 경우에 한 표정에서 다른 표정으로 바로 이동할 수 없고, 이들 사이의 거리는 무한대(inf)로 설정하며 그 사이에 있는 인접한 표정들을 통해서만 이동할 수 있다고 가정한다. 즉, (그림 2)의 1번 표정상태에서 8번 표정상태까지의 최단거리는 8번 표정상태가 1번 표정상태의 인접표정이 아니기 때문에, 2번 표정상태를 거쳐서 8번 표정상태로 진행되어야만 한다. 왜냐하면 1번 표정상태를 중심으로 특정 값을 반지름으로 가지는 가상의 원을 형성하고 원 안에 포함된 표정상태들까지는 직선거리를 사용하고, 원 밖의 표정상태들까지는 원 안의 표정상태들 사이의 직선거리를 거쳐서 가는 최단거리를 구하기 때문이다. 그러나 인접표정을 결정하는 임계값 즉, 각 표정상태를 중심으로 가상의 원을 형성하기 위한 반지름 값은 정해진 특정 값이 아니기 때문에 미리 알기는 쉽지 않다. 왜냐하면 인접거리 임계값은 임의의 표정에서 다른 임의의 표정으로 이동하는 데 필요한 충분한 수의 인접표정들이 나오도록 설정되어야 하기 때문이다. 따라서 이는 실험을 통해서 좋은 결과를 내는 최적의 한계 값을 결정해야 한다. 인접표정들이 결정되면, 인접하지 않은 두 표정상태간의 거리는 그 사이에 있는 인접 표정들 간의 거리들을 합하여 구한다. 이를 위해, 시간복잡도 $O(n^3)$을 가지며 최단거리를 구하는 알고리즘인 플로이드(Floyd) 알고리즘(다이내믹 프로그래밍 기법)[9, 10]을 이용한다. 이렇게 임의의 두 표정간의 최단거리가 구해지면 해당 다양체 공간이 결정된다. 본 논문에서는 모두 2,400여개의 얼굴표정을 사용하여 다양체 공간을 형성하였으며, 이는 2,400여개의 얼굴 표정들 사이의 최단거리들의 집합이다.

4. 다차원 스케일링(MDS)

앞에서 생성한 얼굴표정의 다양체 공간은 300차원 공간이다. 따라서 이 공간을 사용자가 행해하면서 원하는 표정을 선택하여 얼굴 표정을 제어할 수는 없다. 그러므로 원래 표정공간의 구조를 근사적으로 표현하는 2차원 또는 3차원으로 공간을 구하여 이 공간을 항해하는 방법을 사용한다. 본 논문에서는 2차원 공간을 생성한다. 그리고 이를 위해 다차원 스케일링(MDS) 기법[7, 8, 11-13]을 사용한다. MDS는 고차원 데이터들 사이의 거리가 주어지면, 이 거리들의 분포를 대변하는 좌표들의 집합을 구하는 방법이다. 이때 이 좌표들의 차원은 필요에 따라 미리 정하는데, 본 연구에서는 구한 좌표들을 시각적으로 표현해야 되기 때문에, 2차원 좌표를 사용했다.

표정상태를 나타내는 n개의 다차원 다양체 공간상의 점들을 $x_1, \cdots, x_n$, 임의의 표정상태 x_i와 x_j 사이의 다양체 거리를 d_{ij}, $i, j = 1, \cdots, n$이라고 하자. d_{ij}로 구성된 거리 행렬을 $\{d_{ij}\}$로 표시한다. 행렬 $D = \{d_{ij}\}$는 대각 원소들이 0인 대칭행렬이다. 이를 MDS 이론에서 보통 비유사성 대칭행렬이라고 한다. 다차원 표정공간을 근사적으로 나타내는 p차원 공간상의 점들을 $y_1, \cdots, y_n$라 하자. 각 y_i를 상태(Configuration) 벡터라고 한다. 구성 벡터들은 시각화하여 보여주는 것을 목적으로 하는 경우가 많으므로 p는 보통 2 또는 3이다. p차원 점 y_i와 y_j간의 거리를 δ_{ij}이라고 하자. 일반적으로 MDS를 적용하면, p차원 거리들의 집합 $\{\delta_{ij}\}$가 다차원 다양체 거리들의 집합 D와 가장 근사한 분포를 가지도록 하는 p차원 평면상의 점들의 집합 $Y = \{y_i\}$을 구할 수 있다. 집합 Y는 $n \times p$행렬로 표현할 수 있는데, 이를 상태(Configuration) 행렬이라고 한다.

MDS는 최적화 문제로 귀결된다. MDS에서 많이 사용된 최적화 함수는 크게 세 가지로 분류할 수 있다. 첫째 최적화 함수는 거리의 최소 제곱 함수 $STRESS(y_1, \cdots, y_n) = \sum_{i,j=1}^{n} (d_{ij} - \delta_{ij})^2$이고, 둘째 최적화 함수는 거리 제곱의 최소 제곱함수 $STRESS(y_1, \cdots, y_n) = \sum_{i,j=1}^{n} (d_{ij}^2 - \delta_{ij}^2)^2$이다. 여기서 $STRESS$는 'STandard REsidual Sum of Squares'의 약어이다. 이와 같은 최적화 문제는 변수 $\{y_i\}$의 초기 좌표 값(Initial coordinates)을 필요로 한다. 일반적으로 변수 $\{y_i\}$의 초기 좌표 값은 임의로 설정하거나 주성분 분석(Principal Component Analysis, PCA)를 적용하여 구한다. 변수 $\{y_i\}$의 초기 좌표 값을 입력으로 사용한 최적화는 최적화 함수가 최소가 될 때까지 집합 $\{y_i\}$를 반복적으로 갱신한다.

셋째 최적화 함수는 본 논문에서 사용한 것으로서 첫째 및 둘째 최적화 함수와 같이 비유사성 대칭행렬 D가 주어질 때, 행렬 D와 가능한 한 가까운 거리행렬 $\{\delta_{ij}\}$을 구하는 것이 아니라, D를 변환한 행렬 $\{\tau(d_{ij}^2)\}$과 가장 가까운 행렬 $B = \{b_{ij}\}$을 구한다음, 이 행렬 B로부터 우리가 원하는 p차원 평면상의 점들의 집합 $Y = \{y_i\}$을 구하는 방법이다. 이 방법의 최적화 함수는 식 (1)과 같이 표현되는데, STRAIN 함수[7, 12-14]라고 한다. STRAIN은 'STandardized Residual sum of squares between the quantities b_{ij} And the INner products'을 의미하는 약어이다.

$$STRAIN(B) = \sum_{i,j=1}^{n} (b_{ij} - \tau(d_{ij}^2))^2 \qquad (1)$$

여기서 $B = \{b_{ij}\}$는 행렬의 계수(rank)가 p이하인 양의 준정부호(positive semidefinite)행렬이다. 그리고 $\tau(d_{ij}^2) = -\frac{1}{2} H d_{ij}^2 H$이며, 행렬 $H = I_n - \frac{J}{n}$로서, I_n는 크기 n의 항등 행렬, J는 모든 요소가 1인 $n \times n$ 정방행렬이다. 식 (1)을 최소화하는 행렬 B을 구하고, 이로부터 원하는 집합 $Y = \{y_i\}$를 구하는 것은 잘 개발된 이론에 근거하고 있다. 이 이론은 Mardia[14]와 Torgerson[7]에 의해서 개발되었다. 이 이론은 다음과 같이 두 부분으로 이루어져 있다. 첫 부분은 식 (1)을 최소화하는 행렬 B를 구하는 방법이다. 둘째 부분은 행렬 B로부터 원하는 집합 $\{y_i\}$를 얻는 방법이다. 식 (1)을 최소화하는 행렬을 $B^{\star}$라 하자. 최적화 행렬 $B^{\star}$는 행렬 $B^0 = \tau(d_{ij}^2)$의 고유치(Eigenvalue) 및 고유벡터(Eigenvector)를 이용하여 구한다. 행렬 B^0의 고유치를 $\lambda_1 \geq \cdots \geq \lambda_n > 0$이라고 하고($B^0$가 양의 준정부호(positive semidefinite)이므로 모든 고유치들이 0보다 크거나 같음), 이에 대응하는 고유벡터를 $v_1, \cdots, v_n$라고 하자. 그리고 $\lambda_i^+ = \max(\lambda_i, 0)$, $i = 1, \cdots, p$ 및 $\lambda_i^+ = 0$, $i = p+1, \cdots, n$라 할 때, 식 (1)의 전역 최소화 값(Global minimizer)은 식 (2)로 표현된다.

$$B^{\star} = \sum_{i=1}^{n} \lambda_i^+ v_i v_i' = \sum_{i=1}^{p} \lambda_i^+ v_i v_i' \qquad (2)$$

집합 $\{y_i\}$는 식 (1)의 최적화 행렬인 식 (2)의 행렬 $B^{\star}$을 이용하여 구한다. 행렬 $B^{\star}$는 다음 식 (3)과 같이 표현할 수 있는데, 이때 Y가 우리가 원하는 점들의 집합 $\{y_1, \cdots, y_n\}$이다.

$$B^{\star} = YY' \qquad (3)$$

행렬 $B^{\star}$는 다음 식 (4)와 같이 인수분해 하여 표현할 수 있는데,

$$B^{\star} = \Gamma \Lambda \Gamma^T \qquad (4)$$

여기서 $\Lambda = diag(\lambda_1^+, \cdots, \lambda_p^+)$로서 행렬 $B^{\star}$의 고유치들의 대각선 행렬이다. 그리고 $\Gamma = (v_1, \cdots, v_p)$로서 고유치에 대응되는 고유벡터들의 행렬이다. 그러므로 p차원 평면상의 점들의 집합 Y는 식 (1)~식 (4)에서 다음 식 (5)와 같이 구한다.

$$Y = \Gamma \Lambda^{\frac{1}{2}} = \{ v_i \sqrt{\lambda_i^+} \}, \; i = 1, \cdots, p \qquad (5)$$

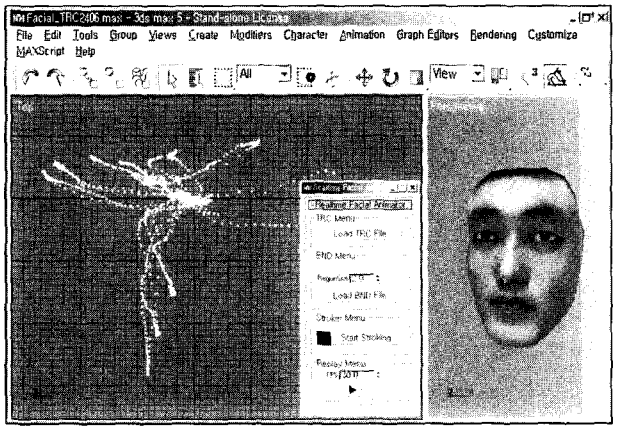
2차원 평면상의 점들의 집합 $\{y_i\}$는 식 (5)에서 $p = 2$로 설정하여 구한다.

본 논문에서는 2차원 평면상의 점들의 집합 $\{y_i\}$을 구하기 위해 Matlab V6.5를 사용하였다. Matlab의 cmdscale(Classical

Multidimensional Scale) 함수에 비유사성 대칭행렬 D와 차원의 수 2를 입력으로 사용하여 MDS를 구하고, 이를 집합 $\{y_i\}$로 사용하였다.

5. 사용자 인터페이스 및 실험

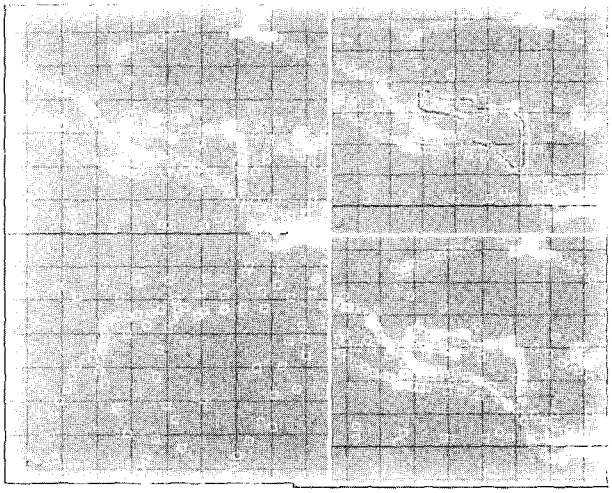
2,400여개의 표정으로 구성된 표정공간을 생성한 후에 이를 MDS를 통해 2차원 공간으로 투영하고, 사용자로 하여금 이 공간을 항해하면서 3차원 아바타의 얼굴 표정을 제어하였다. 이를 위한 사용자 인터페이스는 (그림 3)과 같으며, 3D Studio MAX R5.1의 MAXScript로 구현하였다.



(그림 3) 사용자 인터페이스(좌 : 항해공간과 스크립트 메뉴, 우 : 3차원 얼굴 모델), 사용자가 2차원 항해공간에 분포된 각 얼굴 표정상태를 대표하는 작은 점을 마우스로 선택하면서 항해를 하면, 선택된 점에 해당되는 표정이 3차원 얼굴 모델에 보여진다.

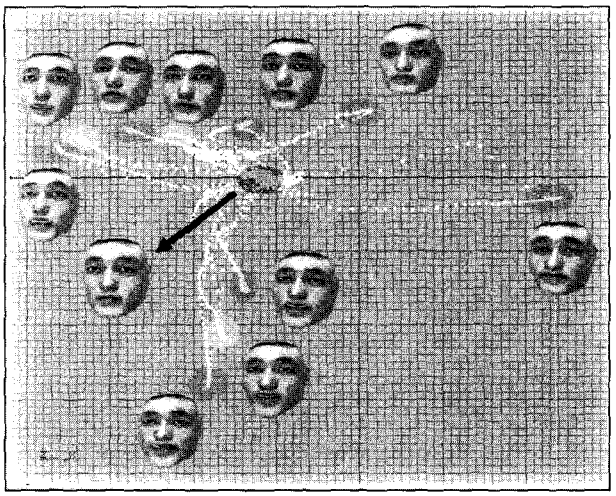
본 논문에서는 10개의 서로 다른 얼굴 모션 데이터를 사용하였으며, 모든 데이터는 무표정 상태에서 출발하여 특정 표정까지 진행한 다음, 다시 무표정 상태로 되돌아오는 과정으로 캡쳐된 데이터를 사용하였다. 사용자 인터페이스에서 항해공간의 점들의 분포는 무표정 상태인 한 점을 중심으로 방사형 분포를 이루고 있다. 이것은 10개의 서로 다른 모션 데이터들이 동일한 무표정 상태에서 출발하여 특정 표정으로 진행해갈수록 서로 다른 모션 데이터에 속한 표정들 사이의 거리가 멀어지기 때문이다.

(그림 4)는 MDS에 의해서 생성된 2,400여개의 2차원 표정공간을 3차원 아바타의 얼굴 표정을 제어하기 위하여 사용자가 항해한 경로를 표시한 것이다. 사용자는 마우스를 사용하여 항해 공간을 항해하고 동시에 3차원 아바타의 얼굴 모델에 적용된 얼굴 표정을 보게 된다. 사용자의 항해 과정이 끝나면 사용자의 항해 궤도에 해당되는 얼굴 표정들을 연속적으로 보고 얼굴 표정의 변화를 확인할 필요가 있다. 이때에는 사용자의 항해 궤도를 처음부터 끝까지 자동으로 반복 항해하여 연속된 얼굴 표정의 변화를 3차원 얼굴 모델이 보여주게 된다. (그림 4)우하의 적십자는 이때의 항해 경로를 따라가는 현재의 경로 위치를 가리켜주기 위한 지시자이다.



(그림 4) 좌 : 2차원 표정공간의 점들, 우상 : 항해과정. 빨간색 점들이 항해 경로를 표시, 우하 : 적십자는 항해 경로의 재생 시 재생중인 현재 프레임을 표시

사용자는 2차원 표정공간에 존재하는 한 표정에서 다른 표정으로 항해할 때 인접 표정으로만 항해를 해야 한다. 그러나 만약 사용자가 인접하지 않은 표정으로 항해를 하고자 한다면 이를 허용하지 말아야 하는 것이 원칙인데, 그렇게 막지 않아도 그렇게 할 사용자는 많지 않을 것이다.



(그림 5) 사용자가 모든 2차원 표정공간을 항해하면서 마우스로 선택한 경로들 상에 있는 대표적인 얼굴표정들을 설명의 편리를 위해 항해공간에 표시함

(그림 5)는 사용자가 2차원 표정공간을 모두 항해하면서 마우스로 선택한 경로들 상에 있는 대표적인 얼굴표정들을 표시한 것으로서, 원래 얼굴모델은 (그림 3)에서와 같이 사용자 인터페이스의 오른쪽 창에 표시되지만, 설명의 편리를 위해 항해공간에 표시하였다. 그리고 (그림 5)는 거리행렬 방식에 의해 표현된 표정상태를 사용한 결과로서 3차원 아바타의 얼굴 표정을 실시간으로 제어하기에 매우 효율적으로 분포되어져 있다. 거리행렬 방식에 의해 표현된 표정상태

를 사용한 실험에서는 인접표정을 결정하는 인접거리 임계값을 실험에 의해 최적의 값을 결정해야 한다.

본 논문에서는 임계값을 정할 때, 다음과 같은 두 가지 기준을 적용하였다. 첫째, 인접거리 임계값에 따라 300차원 다양체 공간에서의 표정 최단거리 분포와 2차원 공간에서의 표정 최단거리 분포 사이의 상관도가 달라진다. 이때 이왕이면 두 분포 사이의 상관도가 높게 나오는 인접거리 임계값을 쓰는 것이 좋다. 2차원 공간은 다차원 공간을 근사적으로 표현하는 것이므로 두 공간 사이의 상관도가 높을수록 좋다. 둘째, 상관도를 높게 만드는 임계값들 중에서 보다 안정적인 임계값을 사용하는 것이 좋다. 본 논문에서는 주어진 임계값이 안정적이라는 것을 그 임계값을 사용할 때의 2차원 최단거리 분포와 그 주변의 임계값을 사용할 때의 2차원 최단거리 분포가 크게 다르지 않다는 의미로 사용한다. 2차원 최단거리 분포는 사용자가 시각적으로 직접 확인할 수 있다. 인접거리 임계값을 조금만 변경해도 2차원 최단 거리 분포가 많이 바뀐다는 것은 대응되는 다차원 최단 거리 분포도 많이 바뀐다는 것을 의미한다. 이것은 이미 우리가 두 공간간의 상관도를 높게 만드는 임계값들만 고려하고 있기 때문이다. 인접거리 임계값이 안정적이지 않으면 그 값이 실제의 인접거리를 대변한다는 것을 믿기가 힘들다. 실제의 인접거리는 우리가 정의한 의미의 안정성을 가질 것으로 예상되기 때문이다.

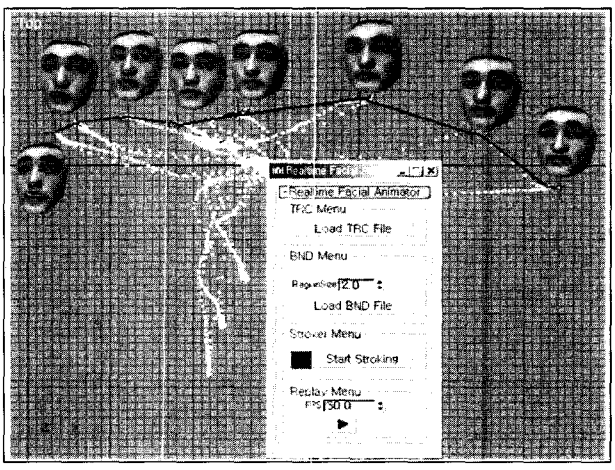
첫째 기준을 적용하기 위하여 다차원 최단거리 분포와 2차원 최단거리 분포사이의 상관계수는 피어슨(Pearson)의 상관계수, r[15]을 이용하여 구하였다. 다양체 공간상의 임의의 표정상태 x_i와 x_j사이의 다양체 거리를 하나의 벡터로 표현하고, 이를 V_d라고 하자. 2차원 평면에 투영된 임의의 점 y_i와 y_j사이의 최단거리를 하나의 벡터로 표현하고, 이를 V_y라고 하자. 상관계수 r은 $-1 \le r \le 1$의 범위 값을 가지며, 벡터 V_d와 벡터 V_y을 이용하여 다음 식 (6)과 같이 계산되어진다.

$$r = \frac{\sum_{i=1}^{n} (v_{di} - \overline{V_d})(v_{yi} - \overline{V_y})}{\sqrt{\sum_{i=1}^{n} (v_{di} - \overline{V_d})^2 \cdot \sum_{i=1}^{n} (v_{yi} - \overline{V_y})^2}} \quad (6)$$

여기서 $v_{di} \in V_d$, $v_{yi} \in V_y$, $\overline{V_d}$는 V_d의 평균, $\overline{V_y}$는 V_y의 평균을 의미한다. 일반적으로 상관계수 r은 $r \ge 0.90$일 경우 매우 높은 상관관계를 가지며, $r = 1.0$일 때에는 일치한다. 본 논문에서는 가장 높은 상관관계를 가지는 인접거리 임계값을 찾는다. 실험 결과 가장 높은 상관관계는 $r = 0.9647$이며, 이때의 인접거리 임계값은 230㎜이다. (그림 5)는 이 인접거리를 이용하여 구성한 2차원 최단 거리 분포이다.

둘째 기준을 적용하기 위하여 두 공간간의 가장 높은 상관계수를 가지는 인접거리 임계값 230㎜ 주변 임계값(220㎜에서 240㎜까지)을 조금씩 변경하면서 2차원 최단 거리 분포를 생성하였다. 그리고 이를 차례대로 디스플레이하면서 거리분포의 변화를 확인하였다. 그 결과 인접거리 임계값이 220㎜에서 230㎜까지 증가할 때 최단거리 분포에서 인접 표정들 사이의 거리가 거의 유사하거나 미세한 차이로 점점 가까워짐을 확인할 수 있었다. 또한 인접거리 임계값이 230㎜에서 240㎜까지 증가할 때, 인접표정들 사이의 거리가 거의 유사하거나 매우 미세하게 멀어지는 것을 확인하였다. 이것은 인접거리 임계값 230㎜를 전후하여 2차원 거리 분포도가 크게 변하지 않는다는 것이다. 따라서 안정적인 임계값들 중에서 두 공간간의 상관도를 가장 높게 만드는 임계값 230㎜을 사용하기로 하였다. 한편, 두 공간간의 상관도를 매우 높게 만드는 인접거리 임계값(220㎜에서 240㎜까지)을 사용했을 때, 임의의 두 표정 간의 경로가 존재하지 않는 경우는 없었다. 그러나 두 공간간의 상관도를 낮게 만드는 상관계수(r=0.9040 이하)일 때의 인접거리 임계값(63㎜ 이하)을 사용할 때는 경로가 존재하지 않는 표정들이 많아진다. 이것은 인접거리 임계값을 작게 잡으면 임의의 두 표정사이에 인접표정들을 거쳐서 가는 최단 경로를 구할 때, 최단경로가 존재하지 않는 경우가 많이 생기기 때문이다.

본 논문에서는 인접거리 임계값을 230㎜로 주었을 때 생성되는 2차원 거리 분포(그림 5)를 사용하여 3차원 아바타의 얼굴 표정을 제어하였다. 본 논문에서 자세히 언급하지는 않았지만 위치벡터방식으로 표정상태를 표현하는 경우, 위에서 언급한 결과들이 나오지 않았다. 즉, 두 공간간의 상관도를 가장 높게 만드는 인접거리 임계값을 사용했을 때, 임의의 두 표정 간의 경로가 존재하지 않는 경우가 있었다. 또한 이 때문에 안정적인 인접거리 임계값을 찾는 것도 무의미하였다.



(그림 6) 표정 제어를 위해 사용자의 항해 경로(파랑색)에 따라 나타나는 실시간 렌더링을 거친 3차원 아바타의 실시간 얼굴 표정 변화를 설명의 편리를 위해 항해공간에 표시함. 메뉴는 본 논문을 위해 설계 및 구현된 것임

(그림 6)은 2,400여개의 얼굴 표정들이 분포된 2차원 표정 공간에서 3차원 아바타의 얼굴 표정 제어를 위해서 마우스를 사용하여 임의의 방향으로 항해경로를 실시간적으로 생성할 때, 3차원 아바타의 얼굴 표정 변화를 실시간으로 보여주고 있다. 3차원 아바타의 얼굴 표정 제어는 (그림 6)의 스크립트 메뉴 'Start Stroking'을 사용하여 항해공간을 항해하는 사용자에 의해서 실시간적으로 수행되어진다. 즉, (그림 6)과 같이 하나의 표정에서 다른 표정으로 표정의 변화를 줄 때, 인접한 표정들을 항해하면 유사한 표정들이 실시간적으로 나타나고 거리가 멀어질수록 특정 표정으로 변화가 실시간적으로 진행된다. 그러나 인접한 표정을 거치지 않고 특정 거리만큼 점프를 하여 실시간 항해를 할 경우에는 매우 큰 표정의 변화가 있는 서로 다른 표정을 실시간적으로 보여준다. 또한 (그림 6)의 스크립트 메뉴 'Replay Menu'을 사용하여 사용자가 재생 속도(fps)를 마음대로 설정함으로써 항해경로에 해당되는 표정들의 실시간 재생을 반복적으로 수행하여 확인할 수 있도록 하였다.

본 논문은 본 시스템이 어떤 효과가 있는지를 알기 위해 애니메이션 업계에서 재직 중인 약 30명의 사용자들로 하여금 약 30분 동안 20회 이상씩 본 시스템을 사용하여 3차원 아바타의 표정 제어를 수행하게 하였다. 그 결과 실험에 참여한 사용자들의 70%정도는 표정의 변화가 빠르고 급격한 표정들의 제어뿐만 아니라 표정의 변화가 정밀하면서 유사한 표정들을 거쳐서 다른 표정으로의 변화를 제어할 수 있는 효과가 있다고 판단하였다. 그러나 약 30%정도의 사용자들은 모든 표정들이 점으로만 표현되어져 있어서 대표적인 표정들의 위치를 기억하고 있어야 한다는 문제점이 있다고 하였으나, 수번의 사용 경험을 통하여 표정들의 분포를 알 수 있었다고 하였다. 이 문제는 향후 사용자 인터페이스를 보완하여 본 시스템을 처음 사용하는 사용자라도 표정들의 분포를 쉽게 파악할 수 있도록 해결해야한다.

6. 결 론

본 논문에서는 다량의 얼굴 모션 데이터들을 적당한 공간에 분포시키고, 사용자가 이 공간을 항해하면서 원하는 얼굴 표정들을 실시간적으로 선택하여 3차원 아바타의 얼굴 표정을 실시간적으로 제어하는 방법을 기술하였다. 2,400여개의 얼굴 표정 프레임을 사용하여 항해 공간을 구성하였으며, 이 공간을 구성하기 위해서 임의의 두 얼굴 표정간의 최단거리인 다양체 거리를 계산하였다. 다양체 거리(최단거리)를 구하기 위해 사용된 플로이드 알고리즘은 시간복잡도 $O(n^3)$을 가지는데, 최단거리를 구하는 알고리즘은 시간복잡도 $O(n^2)$을 가지는 다익스트라 알고리즘도 있다. 그러나 다익스트라 알고리즘은 한 점에서 출발하여 각 정점까지의 최단거리를 구하지만, 플로이드 알고리즘은 모든 정점에서 출발하여 출발한 정점을 제외한 모든 정점까지의 최단거리를 구하는 알고리즘이다. 시간 복잡도만을 가지고 비교를 하면 당연히 다익스트라 알고리즘이 효율적이라고 볼 수 있으나, 본 논문의 목적상 다익스트라 알고리즘은 적합하지 않다. 왜냐하면, 2,400여개의 얼굴 표정들 사이의 거리를 모두 계산하여야 하기 때문이다. 또한 플로이드 알고리즘은 시간복잡도에 비해 계산 속도가 매우 빠르다.

3차원 아바타의 얼굴 표정을 실시간으로 제어하기 위해서 개발한 사용자 인터페이스는 표정공간을 위한 최적의 다양체 거리 계산방법을 실험적으로 확인하는데 유용하게 사용되었다.

지금까지 얼굴 모션 캡쳐 데이터를 사용한 연구들은 모션 캡쳐 데이터의 재사용과 관련된 리타겟팅 기법에 한정된 것이 대부분이며, 본 논문의 주제인 실시간 표정 제어와 관련된 연구는 없었다. 그만큼 얼굴 모션 캡쳐 데이터와 관련된 연구는 세계적으로 많은 연구가 진행되지 않았다. 그러므로 본 논문의 연구 결과는 기존의 관련 연구와 비교할 때, 모션 데이터를 사용한 3차원 아바타 얼굴 표정의 실시간 제어가 가능하다는 것이 가장 큰 특징이다. 예를 들어 채팅, 메신저 및 아바타를 사용한 각종 인터넷 응용 프로그램에서 상대방과 의사소통 등의 커뮤니티를 하는 도중에 자신의 감정을 3차원 아바타를 사용하여 실시간적으로 실감나게 표현해줄 필요가 있는데, 이런 경우 본 논문의 연구 결과는 매우 적합하다. 그러므로 본 논문의 연구결과는 3차원 아바타의 실시간 얼굴 표정 제어가 필요한 각종 응용 프로그램 분야에서 매우 유용하게 사용될 것이다.

참 고 문 헌

[1] Demetri Terzopoulos, Barbara Mones-Hattal, Beth Hofer, Frederic Parke, Doug Sweetland, Keith Waters, "Facial animation : Past, present and future," Panel, SIGGRAPH97.

[2] Frederic I. Parke, Keith Waters. "Computer facial animation," A. K. Peters, 1996.

[3] Brian Guenter, Cindy Grimm, Daniel Wood, Henrique Malvar, and Frederic Pighin. "Making Faces," In SIGGRAPH 98 Conference Proceedings. ACM SIGGRAPH, July, 1998.

[4] Won-Sook Lee, Prem Kalra, Nadia Magnenat Thalmann, "Model based face reconstruction for animation," Proc. MMM'97 (World Scientific Press), Singapore, pp.323-338, 1997.

[5] Cyriaque Kouadio, Pierre Poulin and Pierre Lachapelle, "Real-time facial animation based upon a bank of 3D facial expressions," Proc. Computer Animation 1998, June, 1998.

[6] Wonseok Chae, Yejin Kim, Sung Yong Shin, "An Example-based Approach to Text-driven Speech Animation

with Emotional Expressions," EUROGRAPHICS 2003 Vol.22, No.3.

[7] W. S. Torgerson. Multidimensional scaling : I. theory and method. Psychometrica., 17, pp.401-419, 1952.

[8] Young, F. W. and Hamer, R. M. "Multidimensional Scaling : History, Theory and Applications," Erlbaum, New York.

[9] R. W. Floyd, "Algorithm 97 : Shortest Path," CACM, Vol.5, pp.345, 1962.

[10] Foster, I., "Designing and Building Parallel Programs," Addison-Wesley, 1995.

[11] T. Cox and M. Cox. "Multidimensional Scaling," Chapman & Hall, London, 1994.

[12] Wolfgang Hardle, Leopold Simar, "Applied Multivariate Statistical Analysis," Springer Verlag, pp.373-392, 2003.

[13] P. Tarazaga and M. W. Trosset, "An Approximate Solution to the Metric SSTRESS Problem in Multidimensional Scaling," Computing Science and Statistics, 30, pp.292-295, 1998.

[14] K. V. Mardia, "Some properties of classical multi-dimensional scaling," Communications in Statistics-Theory and Methods, A7, pp.1233-1241, 1978.

[15] Uprendra Shardanand, "Social information filtering for music recommendation," Master's thesis, MIT, 1994.

[16] Sanchez Lorenzo, M., J. D. Edge, S. King and S. Maddock, "Use and Re-use of Facial Motion Capture Data," Proc. Vision, Video and Graphics 2003, University of Bath, pp. 135-142, July, 2003.

김 성 호

e-mail : kimsh1204@hotmail.com

1996년 상지대학교 이공과대학 전산학과 (학사)

1998년 숭실대학교 일반대학원 컴퓨터학과 (공학석사)

2001년 숭실대학교 일반대학원컴퓨터학과 (박사수료)

1997년~1999년 숭실대학교 전자계산원 시간강사

1999년~2000년 숭실대학교 컴퓨터학부 시간강사

1999년~2002년 숭의여대 인터넷정보과 시간강사

숭의여대 컴퓨터게임과 시간강사

2002년~현재 숭의여대 정보통신계열 멀티미디어콘텐츠전공 겸임교수

관심분야 : 컴퓨터 그래픽스, 모션 캡쳐 애니메이션, 가상현실, Web3D, 멀티미디어 등

정 문 렬

e-mail : moon@mail.sogang.ac.kr

1980년 서울대학교 계산통계학과 졸업 (학사)

1982년 KAIST 전산학과(공학석사)

1992년 펜실베니아 대학교 전산학과 (공학박사)

1992년~1994년 일본 구주공과대학교 정보공학부 교수

1994년~1999년 숭실대학교 컴퓨터학부 교수

2000년~현재 서강대학교 영상대학원 미디어공학과 교수

관심분야 : 컴퓨터 그래픽스, 디지털 방송 등