Sammon 매핑을 사용한 모션 데이터의 대화식 표정 애니메이션

김 성 호†

요 약

본 논문은 다량의 고차원 얼굴 표정 모션 데이터를 2차원 공간에 분포시키고, 애니메이터가 이 공간을 항해하면서 원하는 표정들을 실시간적으로 선택함으로써 얼굴 표정 애니메이션을 생성하는 방법을 기술한다. 본 논문에서는 약 2400여개의 얼굴 표정 프레임을 이용하여 표정공간을 구성하였다. 표정공간의 생성은 임의의 두 표정간의 최단거리의 결정으로 귀결된다. 표정공간은 다양체 공간으로서 이 공간내의 두 점간의 거리는 다음과 같이 근사적으로 표현한다. 임의의 마커 간의 거리를 표시하는 거리행렬을 사용하여 각 표정의 상태를 표현하는 표정상태벡터를 정의한 후, 두 표정이 인접해 있으면, 이를 두 표정 간 최단거리(다양체 거리)에 대한 근사치로 간주한다. 그리하여 인접 표정들 간의 인접거리가 결정되면, 이들 인접거리들을 연결하여 임의의 두 표정 상태간의 최단거리를 구하는데, 이를 위해 Floyd 알고리즘을 이용한다. 다차원 공간인 표정공간을 가시화하기 위해서는 Sammon 매핑을 이용하여 2차원 평면에 투영시켰다. 얼굴 애니메이션은 사용자 인터페이스를 사용하여 애니메이터들이 2차원 공간을 항해하면서 실시간으로 생성한다.

Interactive Facial Expression Animation of Motion Data using Sammon's Mapping

Sung-Ho Kim†

ABSTRACT

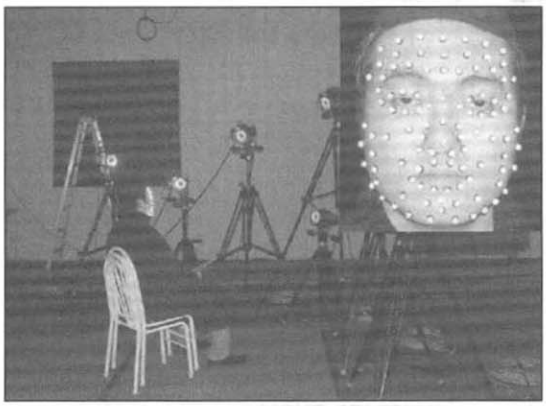
This paper describes method to distribute much high-dimensional facial expression motion data to 2 dimensional space, and method to create facial expression animation by select expressions that want by realtime as animator navigates this space. In this paper composed expression space using about 2400 facial expression frames. The creation of facial space is ended by decision of shortest distance between any two expressions. The expression space as manifold space expresses approximately distance between two points as following. After define expression state vector that express state of each expression using distance matrix which represent distance between any markers, if two expression adjoin, regard this as approximate about shortest distance between two expressions. So, if adjacency distance is decided between adjacency expressions, connect these adjacency distances and yield shortest distance between any two expression states, use Floyd algorithm for this. To materialize expression space that is high-dimensional space, project on 2 dimensions using Sammon's Mapping. Facial animation create by realtime with animators navigating 2 dimensional space using user interface.

키워드 : 얼굴 모션 캡처(Facial Motion Capture), 표정 애니메이션(Facial Animation), 표정상태벡터(Facial State Vector), 다양체 공간(Manifold Space), 다양체 거리(Manifold Distance), Floyd 알고리즘(Floyd Algorithm), 거리행렬(Distance Matrix), Sammon 매핑(Sammon's Mapping)

1. 서 론

인간은 자신의 감정을 말보다는 얼굴 표정을 통해서 더 잘 표현하며, 상대방의 얼굴 표정을 보고 상대방의 현재 감정상태를 파악한다. 그런 연유로 지금까지 3차원 컴퓨터 그래픽스 기법을 통해서 인간의 얼굴 표정을 표현할 수 있는 많은 방법이 연구[1-6]되었다. 최근에는 모션 캡처를 사용한 캐릭터 애니메이션이 컴퓨터 애니메이션 분야에서 각광받으면서 배우의 얼굴 모션을 캡처하여 3차원 캐릭터 애니메이션에 적용하고 있다. 캡처한 데이터를 사용하는 방법으로 크게 두 가지가 제시되었다. 첫째는 배우의 모션 데이터를 새로운 모델에 재적용하는 모션 리타겟팅 기법[5, 6]이다. 모션 리타겟팅 기법은 먼저 몸동작에 대한 것이 제시[6]되었고, 얼굴 표정에 대한 것도 제시[5]되었다. 둘째는 배우의 동작을 가능한 한 많이 캡처하여 이 동작 속에 들어있는 자세들을 획득, 자세 데이터베이스를 만든 후, 애니메이터가 특정 자세들을 선택, 연결하여 새로운 동작을 생성하는 기법이다. 이 기법도 몸동작에 대한 것이 먼저 제시[6] 되었는데, 아직 얼굴 표정에 대한 것은 제시된 것이 없다.

† 준 회 원 : 숭의여대 정보통신계열 멀티미디어콘텐츠전공 교수
논문접수 : 2004년 1월 9일, 심사완료 : 2004년 2월 23일



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

그러므로 본 논문에서는 다량의 얼굴 모션 캡쳐 데이터를 직관적인 공간에 분포시키고, 애니메이터가 적당한 공간을 항해하면서 원하는 얼굴 표정들을 실시간으로 선택하여, 표정 애니메이션을 생성하는 방법을 기술한다. 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 수개의 얼굴 모션을 연출하게 하고, 초당 60 프레임으로 캡쳐한다. 한 개의 마커는 3D 좌표 값으로 표현되므로 100개의 마커위치로 표현되는 한, 표정은 300차원의 데이터이다. 얼굴 표정 데이터는 3차원 공간상의 위치 값으로만 구성되어져 있기 때문에, 얼굴 표정들 사이의 유사성을 수치적으로 표현하기가 어렵다.얼굴 표정들 사이의 유사성은 300차원의 데이터를 2차원이나 3차원과 같은 저차원 평면상에 분포시킬 경우, 얼굴 표정들 사이의 거리 값으로 구분하기 때문에 필요하다. 그러므로 다양한 얼굴 표정들을 잘 구분할 수 있도록 하기 위해서 마커들 사이의 거리를 이용한 벡터행렬로 변경하고, 이를 Sammon 매핑[7, 8]으로 2차원 평면에 투영시킨다. 실시간 표정 애니메이션을 생성하기 위해서는 애니메이터로 하여금 투영된 2차원 공간을 항해하게 하고, 항해경로의 각 점에 해당되는 3차원 얼굴 모델을 디스플레이 한다.

2. 얼굴 표정상태 표현법

표정공간을 생성하기 위해서는 각각의 얼굴 표정상태를 수치적으로 표현해야 한다. 표정상태는 얼굴에 부착된 마커들의 위치에 의해서 결정된다. 표정상태의 표현은 표정들 간의 상대적인 거리 관계를 잘 표현하는 것이어야 하고, 이로써 얼굴 표정을 구분할 수 있어야 한다. 표정의 상태를 표현하는 가장 간단한 방법은 마커들의 위치들로 이루어진 상태벡터를 이용하는 것이다. 본 논문에서 사용한 얼굴 모션 프레임 데이터는 100개의 마커를 사용하고, 한 마커는 3D 좌표를 가지기 때문에, 표정상태벡터는 300차원이 된다. 이런 식으로 표현된 표정상태벡터를 '*위치벡터*'라고 하자. 본 논문에서는 표정상태를 표현할 때, 위치벡터를 사용하지 않는다. 대신 위치벡터의 임의의 두 마커간의 거리를 표현하는 "*거리행렬*"을 이용하여 표정상태를 표현한다. 왜냐하면 거리행렬이 위치벡터보다 얼굴 마커들의 분포상태에 대한 정보를 더 많이 표시하고 있고, 따라서 두 표정간의 거리를 보다 더 정확하게 표현할 수 있기 때문이다. 거리행렬은 위치벡터를 기반으로 다음과 같이 구한다. 즉, 하나의 위치벡터에서 마커와 마커들 사이의 거리를 식 (1)과 같이 직선거리 방법으로 계산한다.

$$D_{ii,c} = \sum_{i=1}^{n} \left(\sum_{j=1}^{99} \sum_{k=j+1}^{100} \sqrt{(M_{i,j1:j2} - M_{i,k1:k2})^2} \right) \quad (1)$$

여기서 M은 위치벡터들의 집합, D는 거리행렬들의 집합, i는 약 2400여개의 위치벡터의 수, ii는 약 2400여개의 거리행렬의 수를 의미한다. j_1, j_2 및 k_1, k_2는 모션 프레임 데이터에서 프레임별 각 마커들의 3D 좌표 값으로서 마커와 마커 사이의 거리를 구하기 위해서 사용된다. j는 위치벡터별 100개 마커에서 마커와 마커 사이의 거리를 계산하기 위한 기준 마커를 의미한다. k는 위치벡터별 100개 마커 j에 의해서 비교되어지는 마커를 의미한다. c는 마커 100개에 대한 마커들 사이의 거리 계산 회수로 $100\times(100-1)/2 = 4950$을 의미한다.

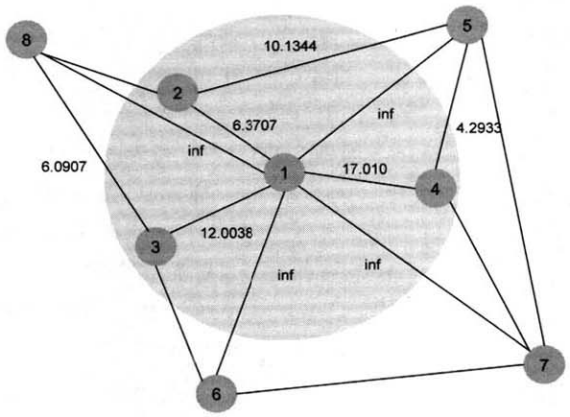
3. 표정공간의 생성

표정상태를 거리행렬로 표현하면 표정간의 직선거리는 두 거리행렬간의 직선거리로 표현된다. 거리행렬로 표현된 표정상태들의 공간은 임의의 두 거리행렬간의 거리를 결정함으로써 결정된다. 본 논문에서는 거리행렬의 각 행을 죽 나열하여 그 결과를 하나의 벡터로 간주하고, 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 구면처럼 다양체 공간인 것이다. 다양체 공간상에서의 거리는 두 점간의 거리를 한 점에서 이 공간을 벗어나지 않으면서 다른 점까지 도달하는 최단경로의 길이로 정의[10]한다.

본 논문에서는 이 다양체 공간을 근사적으로 표현한다. 이를 위해 먼저, 두 거리행렬간의 직선거리가 일정 값 이하

인 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. 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, 그 사이에 있는 인접한 표정들을 통해서만 이동할 수 있다고 가정한다.



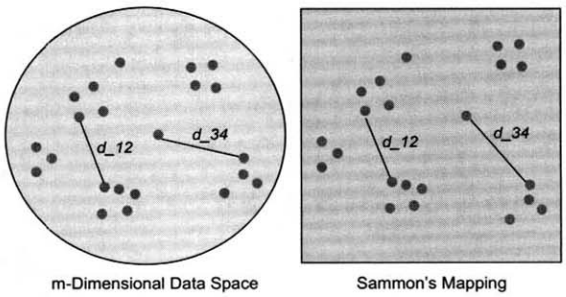
(그림 2) Floyd 알고리즘을 위한 그래프 생성방법 : 1번을 기준으로 2, 3, 4번은 인접표정, 5, 6 7, 8번은 비 인접표정(무한대), 원의 반지름은 인접거리 한계 값

그러나 인접표정을 결정하는 한계거리를 미리 알기는 쉽지 않다. 따라서 이는 실험을 통해서 좋은 결과를 내는 한계 값을 정한다. 인접표정을 결정하는 인접거리 임계 값은 실험에 의해 최적의 값을 결정해야 한다. 다만, 인접거리 임계 값은 임의의 표정에서 다른 임의의 표정으로 이동하는 데 필요한 충분한 수의 인접표정들이 나오도록 설정되어야 한다. 인접표정들이 결정되면, 인접하지 않은 두 표정 상태간의 거리는 그 사이에 있는 인접 표정들 간의 거리들을 합하여 구한다. 이를 위해, 최단거리를 구하는 알고리즘인 Floyd 알고리즘(다이내믹 프로그래밍 기법)[10]을 이용한다. 이렇게 임의의 두 표정간의 최단거리가 구해지면 해당 다양체 공간이 결정된다. 본 논문에서는 약 2400여개의 얼굴표정을 사용하여 다양체 공간을 형성하였다.

4. Sammon 매핑

앞에서 생성한 얼굴표정의 다양체 공간은 300차원 공간이다. 따라서 이 공간을 애니메이터가 행해하면서 원하는 표정을 선택할 수는 없다. 그러므로 원래 표정공간의 구조를 근사적으로 표현하는 2차원 또는 3차원으로 공간을 구하여 이 공간을 항해하는 방법을 사용한다. 본 논문에서는 이를 위해 Sammon 매핑 기법[7, 8]을 사용한다. Sammon 매핑은 고차원 데이터들 사이의 거리가 주어지면, 이 거리들의 분포를 보존하면서 2차원이나 3차원과 같은 저차원으로 축소된 좌표들의 집합을 구하는 방법 중의 하나이다. 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 거리행렬을 $\{d_{ij}\}$로 표시한다. 행렬 $D = \{d_{ij}\}$는 대각 원소들이 0인 대칭행렬이다. 다차원 표정공간을 근사적으로 나타내는 2차원 평면상의 점들을 $y_1, \cdots, y_n$ 2차원 점 y_i와 y_j간의 거리를 δ_{ij}이라고 하자. 일반적으로 Sammon 매핑을 적용하면, 2차원 거리들의 집합 $\{\delta_{ij}\}$가 다차원 다양체 거리들의 집합 D와 가장 근사한 분포를 가지도록 하는 2차원 평면상의 점들의 집합 $\{y_i\}$을 구할 수 있다.



(그림 3) 좌 : 원형 m차원 데이터 공간, 우 : 2차원 데이터 공간을 위한 Sammon 매핑

Sammon 매핑은 최적화 문제로 귀결되며, 최적화 함수는 다음 식 (2)와 같다.

$$STRESS(y_1, \cdots, y_n) = \frac{1}{\sum_{i,j=1}^{n} d_{ij}} \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{(d_{ij} - \delta_{ij})^2}{d_{ij}} \quad (2)$$

여기서 *STRESS*는 'STandard REsidual Sum of Squares'의 약어이다. 이와 같은 최적화 문제는 변수 $\{y_i\}$의 초기 좌표 값을 필요로 한다. 일반적으로 변수 $\{y_i\}$의 초기 좌표 값은 임의로 설정하거나 Principal Component Analysis(PCA)[11]을 적용하여 구하는데, 본 논문에서는 임의로 설정한다.

식 (2)의 최적화 함수는 (그림 3)과 같이 기울기를 이용한 확률론적인 접근법을 통하여 최소화되어진다. 축소된 차원 공간의 차원의 수를 p라고 하면, 반복 회수 t에서의 거리 δ_{ij}는 다음 식 (3)과 같다.

$$\delta_{ij}(t) = \sqrt{\sum_{k=1}^{p} [y_{ik}(t) - y_{jk}(t)]^2} \quad (3)$$

여기서 y_{ik}는 2차원 평면상에 존재하는 점 y_i의 k번째 좌표 값이고, y_{jk}는 2차원 평면상에 존재하는 점 y_j의 k번째 좌표 값이다. 변수 $\{y_i\}$의 최적화는 다음 식 (4)와 같이 최적화 함수가 최소가 될 때까지 집합 $\{y_i\}$를 반복적으로 갱신한다.

$$y_{ik}(t+1) = y_{ik}(t) - \alpha \triangle_{ik}(t) \tag{4}$$

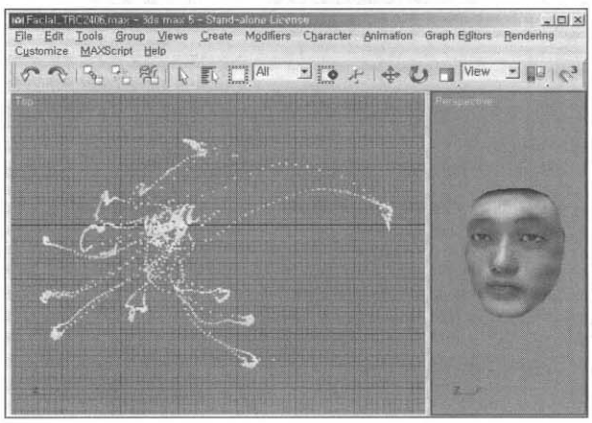
여기서 α는 반복 스텝 단계로서 Sammon은 0.3 혹은 0.4를 제안하였으나, 본 논문에서는 Kohonen[9]의 제안에 따라 0.2를 기본 값으로 가진다. 그리고 $\triangle_{ik}(t)$는 다음 식 (5)와 같다.

$$\triangle_{ik}(t) = \sum_{i,j=1}^{n} [y_{ik}(t) - y_{jk}(t) - \frac{(d_{ij} - \delta_{ij}(t))}{(y_{ik}(t) - y_{jk}(t))} \frac{\delta_{ij}^2(t)}{d_{ij}}] \tag{5}$$

본 논문에서는 2차원 평면상의 점들의 집합 $\{y_i\}$을 구하기 위해 Matlab V6.5를 사용하였다. Sammon 매핑 기법을 기반으로 한 Matlab 함수 Sammon에 표정상태벡터들로 구성된 집합 $\{x_i\}$, 차원의 수 2 및 대칭행렬 D를 입력으로 사용하여 2차원 평면상의 점들의 집합 $\{y_i\}$를 구하였다.

5. 사용자 인터페이스 및 실험

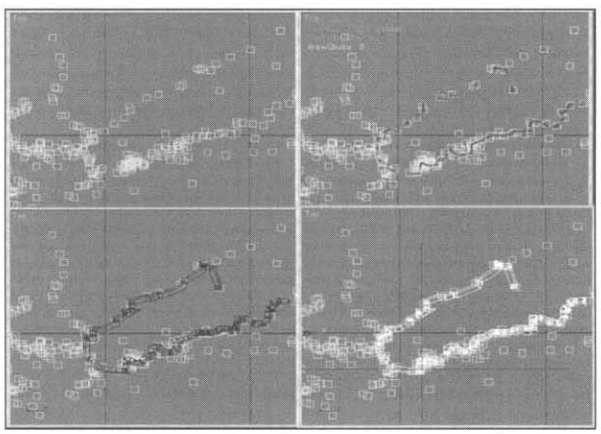
본 논문에서는 약 2400여개의 표정으로 구성된 표정공간을 생성한 후, 이를 Sammon 매핑을 통해 2차원 공간으로 투영하고, 애니메이터로 하여금 이 공간을 항해하면서 얼굴 애니메이션을 생성하게 하였다. 이를 위한 사용자 인터페이스는 (그림 4)와 같다.



(그림 4) 사용자 인터페이스(좌 : 항해공간과 우 : 3차원 얼굴 모델), 애니메이터가 항해공간에 분포된 각 얼굴 표정상태를 대표하는 작은 점을 마우스로 선택하면서 항해를 하면, 선택된 점에 해당되는 표정이 3차원 얼굴 모델에 나타난다.

(그림 4)의 왼쪽은 약 2400여개의 표정상태벡터를 2차원 평면에 분포시킨 항해공간이고, 오른쪽의 3D 얼굴 모델은 애니메이터가 항해 공간을 항해할 때 실시간으로 얼굴표정을 보여주게 한다. 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.

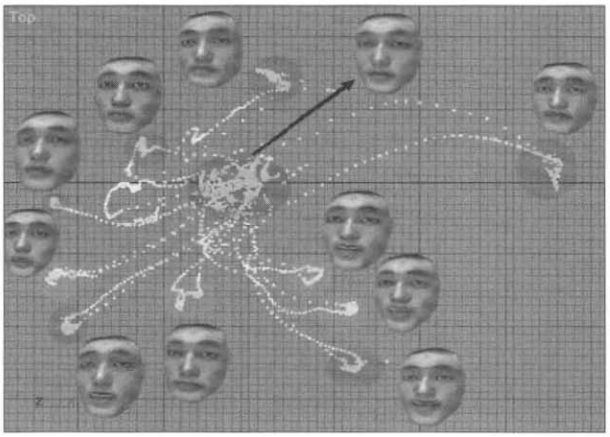
(그림 5)는 Sammon 매핑에 의해서 생성된 약 2400여개의 2차원 표정공간을 애니메이터가 항해하는 과정과 항해경로를 표시한 것이다. 애니메이터는 마우스를 사용하여 항해공간을 항해하고 동시에 3차원 얼굴 모델에 적용된 얼굴표정을 보게 되는데, 이때에는 표정 하나하나를 항해과정과 항해속도에 따라 확인하면서 보게 된다. 애니메이터의 항해과정이 끝나면 애니메이터의 항해경로에 해당되는 얼굴 표정들을 연속적으로 보고 얼굴 표정의 변화를 확인할 필요가 있다. 이때에는 애니메이터의 항해경로를 처음부터 끝까지 자동으로 반복 항해하여 연속된 얼굴 표정의 변화를 3차원 얼굴 모델이 보여주게 된다.



(그림 5) 좌상 : 2차원 표정공간의 점들, 우상 : 항해과정, 빨간색 점들이 항해 경로, 좌하 : 항해 경로를 순서대로 연결, 우하 : 적십자는 반복 재생시 재생중인 현재 프레임

(그림 5) 우하의 적십자는 이때의 항해 경로를 따라가는 현재의 경로 위치를 가리켜주기 위한 지시자이다. 본 사용자 인터페이스는 애니메이터가 (그림 5)와 같은 항해경로를 다수 개 생성하여 서로 다른 색상으로 표현할 수 있으며, 애니메이터가 항해한 경로들 중 임의의 경로 하나를 선택하여 재생하고 확인하여 수정할 수 있다.

(그림 6)은 애니메이터가 모든 2차원 표정공간을 항해하면서 마우스로 선택한 경로들 상에 있는 대표적인 얼굴표정들을 표시한 것으로서, 원래 얼굴모델은 (그림 4)와 같이



(그림 6) 애니메이터가 모든 표정공간을 항해하면서 마우스로 선택한 경로들 상에 있는 대표적인 표정들을 설명의 편리를 위해 항해공간에 표시함

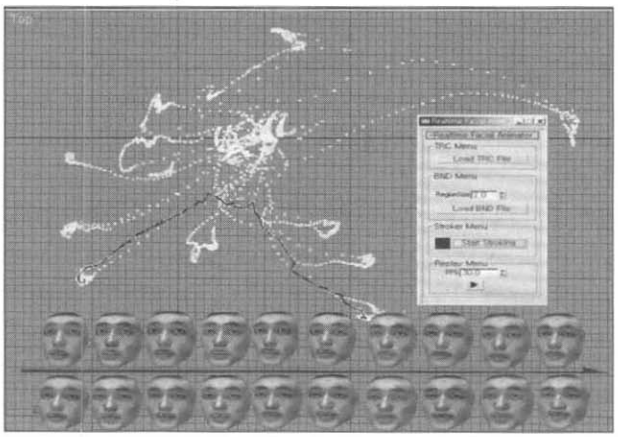
사용자 인터페이스의 오른쪽 창에 실시간으로 표시되지만, 설명의 편리를 위해 항해공간에 표시하였다. 대표적인 얼굴 표정들 사이의 선형 분포들은 이들의 순차적인 중간 표정들로 구성되어져 있다. 그리고 (그림 6)은 거리행렬 방식에 의해 표현된 표정상태를 사용한 결과로서 애니메이션을 생성하기에 매우 효율적으로 분포되어져 있었다.

거리행렬 방식에 의해 표현된 표정상태를 사용한 실험에서는 인접표정을 결정하는 인접거리 임계 값을 실험에 의해 최적의 값을 결정해야 한다. 본 논문에서는 인계 값을 정할 때, 다음과 같은 기준을 적용하였다. 인접거리 임계 값에 따라 300차원 다양체 공간에서의 표정 최단거리 분포와 2차원 공간에서의 표정 최단거리 분포 사이의 상관도가 달라진다. 이때, 가능하면 두 분포 사이의 상관도가 높게 나오는 인접거리 임계 값을 사용하는 것이 좋다. 2차원 공간은 다차원 공간을 근사적으로 표현하는 것이므로 두 공간 사이의 상관도가 높을수록 좋다. 이 기준을 적용하기 위해 다차원 최단거리 분포와 2차원 최단거리 분포사이의 상관도는 피어슨(Pearson)의 상관계수, r[12]을 이용하여 구하였다.다양체 공간상의 임의의 표정상태 x_i와 x_j 사이의 다양체 거리를 하나의 벡터로 표현하고, 이를 V_d라고 하자. 2차원 평면에 투영된 임의의 점 y_i와 y_j 사이의 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 임계 값을 찾는다. 실험 결과 가장 높은 상관관계는 $r = 0.9629$이며, 이때의 인접거리 임계 값은 300mm이다. (그림 6)은 이 인접거리를 이용하여 구성한 2차원 최단 거리 분포이다. 두 공간간의 상관도를 매우 높게 만드는 인접거리 임계 값(280mm에서 310mm까지)을 사용했을 때, 임의의 두 표정 간의 경로가 존재하지 않는 경우는 없었다. 그러나 두 공간간의 상관도를 낮게 만드는 상관계수($r = 0.9001$ 이하)일 때의 인접거리 임계 값(45mm 이하)을 사용할 때는 경로가 존재하지 않는 표정들이 많아진다. 이것은 인접거리 임계 값을 작게 잡으면 임의의 두 표정사이에 인접표정들을 거쳐서 가는 최단 경로를 구할 때, 최단경로가 존재하지 않는 경우가 많이 생기기 때문이다.

본 논문에서는 인접거리 임계 값을 300mm로 주었을 때 생성되는 2차원 거리 분포(그림 6)를 사용하여 애니메이션을 생성하였다. 일반적인 애니메이션이나 동영상은 약 30fps의 렌더링 속도를 가지고 있다. 그러므로 실시간 얼굴 표정 애니메이션을 생성하기 위해서는 적당한 렌더링 속도가 필요하다. 본 논문에서는 (그림 7)과 같이 사용자 인터페이스의 스크립트 메뉴에서 애니메이터가 렌더링 속도의 값을 마음대로 설정하고 실행할 수 있도록 하였다.



(그림 7) 애니메이션의 렌더링 속도를 조정하기 위한 사용자 인터페이스의 스크립트 메뉴와 애니메이터가 마우스로 선택한 특정 경로(파랑색) 상에 있는 표정들을 설명의 편리를 위해 항해공간에 연속적으로 표시함

또한 (그림 7)은 애니메이터가 표정공간을 항해하면서 마우스로 선택한 항해경로(파랑색) 상의 표정들을 30fps의 렌더링 속도로 렌더링한 결과를 순서대로 나열한 것이다.

본 논문에서 자세히 언급하지는 않았지만 위치벡터방식으로 표정상태를 표현하는 경우, 위에서 언급한 결과들이 나오지 않았다. 즉, 두 공간간의 상관도를 가장 높게 만드는 인접거리 임계 값을 사용했을 때, 임의의 두 표정 간의 경로가 존재하지 않는 경우가 있었다.

6. 결 론

본 논문에서는 다량의 얼굴 모션 데이터들을 적당한 공간에 분포시키고, 애니메이터가 이 공간을 항해하면서 원하는 얼굴 표정들을 실시간으로 선택하고 디스플레이 하는 방법을 기술하였다. 약 2400여개의 얼굴 표정 프레임을 사용하여 항해 공간을 구성하였으며, 이 공간을 구성하기 위해서는 임의의 두 표정간의 최단거리인 다양체 거리를 계산하였다. 실시간 얼굴 표정 애니메이션을 생성하기 위해서 개발한 사용자 인터페이스는 표정상태 표현법을 위한 위치 벡터 및 거리행렬 방식을 실험적으로 확인하는데 유용하게 사용되었다. 사용자 인터페이스는 애니메이터가 생성하고자 하는 얼굴 표정 애니메이션을 직관적인 공간을 자유자재로 항해하면서 실시간으로 생성, 확인, 수정 및 재생성이 가능하기 때문에 유용하고 효율적이라는 것을 애니메이터로 하여금 확인할 수 있었다. 그러나 본 사용자 인터페이스는 애니메이터들에게 사용하도록 하여 테스트해본 결과 한 가지 단점이 있었다. 즉, 2차원 공간 내에 분포된 얼굴 표정 프레임들이 어떤 표정들인지 한 눈에 파악할 수 없기 때문에 몇 번의 사용 경험을 요구한다는 것이다. 왜냐하면 수천 개의 얼굴 표정상태를 2차원 공간에 분포시키기가 쉽지 않기 때문에, (그림 6)과 같이 작은 사각형의 점 한 개가 하나의 얼굴 표정상태를 대표하도록 하여 2차원 공간에 분포시켰기 때문이다.

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, 1997.

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

[3] Y. Lee, D. Terzopoulos and K. Waters, "Realistic modeling for facial animation," Proc. ACM SIGGRAPH '95 Conf., pp.55-62, 1995.

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

[5] Cyriaque Kouadio, Pierre Poulin, and Pierre Lachapelle, "Real-time facial animation based upon a bank of 3D facial expressions," Proc. Computer Animation '98, June, 1998.

[6] Jehee Lee, Jinxiang Chai, Paul Reitsma, Jessica Hodgins, and Nancy Pollard, "Interactive Control of Avatars Animated with Human Motion Data," ACM Transactions on Graphics(SIGGRAPH 2002), Vol.21, No.3, pp.491-500, July, 2002.

[7] M. E. Tipping, "Topographic mappings and feed-forward neural networks," PhD thesis, Aston University, Birmingham, UK, February, 1996.

[8] Sammon, J. W. Jr., "A nonlinear mapping for data structure analysis," IEEE Transactions on Computers, Vol.C-18, No.5, pp.401-409, 1969.

[9] T. Kohonen, J. Kangas, J. Laaksonen and K. Torkkola, "Lvq pak : A program package for the correct application of learning vector quantization algorithms," in Proc. Intl, Joint Conf, Neural Networks, pp.725-730, 1992.

[10] R. W. Floyd, "Algorithm '97 : Shortest Path," CACM, Vol.5, p.345, 1962.

[11] Hotelling, H., "Analysis of a complex of statistical variables into principal components," Journal of Educational Psychology, 24, pp.417-441, 1933.

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

[13] 박연출, 오해석, "벡터기반의 캐리커처 자동생성에 관한 연구", 정보처리학회논문지B, 제10-B권 제6호, 2003.

[14] 이옥경, 박연출, 김성호, 오해석, "얼굴 특징 추출을 위한 캐리커처 자동 생성", HCI 2001, 2001.

김 성 호

e-mail : kimsh1204@hotmail.com

1989년 상지대학교 이공과대학 전산학과 (학사)

1998년 숭실대학교 일반대학원 컴퓨터학과 (공학석사)

2001년 숭실대학교 일반대학원 컴퓨터학과 (박사수료)

1997년~1999년 숭실대학교 전자계산원 시간강사

1999년 한국방송통신대 방송정보학과 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, Web 3D, 컴퓨터 비젼, 영상처리, 멀티미디어 등