군집화 기반 3차원 애니메이션 데이터 압축 기법

Grouping-based 3D Animation Data Compression Method

최영진, Young-Jin Choi*,여두환, Du-Hwan Yeo**, 김형석, HyungSeok Kim***, 김지인, Jee-In Kim****

요약 가상현실 및 증강 현실 응용 특히 디지털 패션과 같은 응용분야에서 아바타에 여러 가지 형태의 의복을 입히고 장신구들을 장식하여 보다 현실감이 나는 3차원 애니메이션을 제작한 다음 인터넷을 통하여 개인용 컴퓨터는 물론 핸드폰 같은 휴대용 정보단말기에 표현하려는 요구가 증가하고 있다. 하지만 3차원 모델을 실감나게 표현하는데 필요한 데이터의 용량이 매우 크고 네트워크의 속도 문제와 단말기에 장착된 메모리 장치의 제한적인 성능 문제등으로 인하여, 3차원 영상의 효과적인 전송 및 실감나는 표현은 매우 어렵다. 본 논문에서는 3차원 애니메이션에서 사용되는 데이터를 압축하는데, 3차원 모델 데이터를 군집화하여 애니메이션에 필요한 저장 공간을 줄여서 휴대용 정보단말기에서도 3차원 모델을 자연스럽게 렌더링할 수 있게 하는 기법을 제안하고자 한다. 제안된 기법은 각종 디포머블 오브젝트에도 응용될 수 있으며, 기존의 정적인 형상 압축 기술과 연계하여 좀 더 높은 압축을 가능하게 할 것으로 기대된다.

Abstract The needs for visualizing interactive multimedia contents on portable devices with realistic three dimensional shapes are increasing as new ubiquitous services are coming into reality. Especially in digital fashion applications with virtual reality technologies for clothes of various forms on different avatars, it is required to provide very high quality visual models over mobile networks. Due to limited network bandwidths and memory spaces of portable devices, it is very difficult to transmit visual data effectively and render realistic appearance of three dimensional images. In this thesis, we propose a compression method to reduce three dimensional data for digital fashion applications. The three dimensional model includes animation of avatar which require very large amounts of data over time. Our proposed method utilizes temporal and spatial coherence of animation data, to reduce the amount. By grouping vertices from three dimensional models, the entire animation is represented by a movement path of a few representative vertices. The existing three dimensional model compression approaches can get benefits from the proposed method by reducing the compression sources through grouping. We expect that the proposed method to be applied not only to three dimensional garment animations but also to generic deformable objects.

Keyword : 3D Compression, 3D Animation,

본 논문은 서울시 산학연 협력 사업의 지원으로 수행되었으며, 본 연구에서 사용된 의류 모델은 SKC&C 의 지원을 받았으며 dolphin과 chicken 모델은 Zachi Karni 에게 제공 받았음

*주저자 : 건국대학교 신기술 융합과 e-mail: rarcoon@hotmail.com

**공동저자 : SKC&C u-Biz. 연구소 R&D 센터 e-mail: twone@skcc.com

***공동저자 : 건국대학교 인터넷미디어 공학부 e-mail: hyuskim@konkuk.ac.kr

****교신저자 : 건국대학교 신기술 융합과 e-mail: jnkm@konkuk.ac.kr

1. 서론

1.1 연구 배경

정보화 시대가 되면서 문자, 소리, 영상등 다양한 형태로 존재하는 많은 정보들이 생성되고 사용되고 있다. 이와 함께 각종 기기와 다양한 기술의 발달로 기존의 정보형태에서 새로운 3차원 정보가 생성되어 여러 분야에서 쓰이고 있다. 이러한 3차원 정보는 기존의 정보보다 더 정확한 정보를 사용자에게 알려주어 많은 분야에서 사용되고 있다.

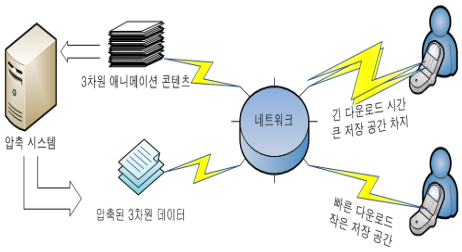
한 예로써 의류 산업, 섬유공학 및 컴퓨터공학의 발전으로 등장한 디지털 패션 기술은 인터넷에서 소비자의 신체 조건과 취향에 맞춘 의류를 주문할 수 있는 서비스를 제공하고 있다. 이러한 서비스는 모바일 통신과 결합하여, 언제 어디서나 소비자가 원하는 의류 상품을 휴대용 정보단말기를 이용하여 표현하고 있다. 뿐만 아니라 게임이나 영화 등 각종 엔터테인먼트 사업에서도 3차원 모델의 애니메이션이 사실성과 표현력을 높이기 위하여 사용된다.

[그림 1] 3차원에서 아바타가 의류를 착용한 애니메이션

이러한 정보 표현을 위한 3차원 정보의 데이터는 매우 큰 저장 공간을 필요로 한다. 온라인 환경에서 휴대용 정보 단말기기를 사용하여 3차원 정보를 확인하는 경우 네트워크 속도와 기기가 갖는 작은 저장 공간과 낮은 CPU 성능 때문에 3차원 정보의 효과적인 표현이 문제가 된다.

본 논문에서는 효과적인 표현을 위하여 3차원 애니메이션 정보의 압축 방법을 제안한다. 제안하는 방법은 3차원 정보에 대하여 움직임의 유사성을 바탕으로 그룹을 만들어 압축한다. 별도의 패턴 분류의 방법 없이 전체 프레임을 살펴봄으로써 압축을 실시하였으며 거기에 보간법을 더하여 전체적으로 높은 압축률과 낮은 에러율을 보여주고자 한다.

본 논문의 구성은 다음과 같다. 먼저 2장에서는 기존에 사용하고 있는 3차원 자료들을 살펴보고 여러 가지의 3차원 데이터 압축 기법에 대하여 기술한다. 3장에서는 3차원 모델의 저장 공간에 대한 이야기 한 후 군집화(Grouping) 이라는 방법을 소개한다. 4장에서는 3차원 자료에 적용할 압축 기법들을 설명하고 적용한다. 5장에서는 이러한 압축 결과를 비교 분석한다. 마지막으로 6장에서는 결론을 내면서 향후 발전 방향을 이야기한다.



[그림 2] 3차원 애니메이션 콘텐츠 압축 시스템의 효과

2. 관련 연구

2.1 3차원 데이터의 정점의 압축

정점의 위치를 압축하는 방법은 다음과 같다.

Multiresolution[1]방법은 사용자가 원하는 해상도에 따라 저장 공간의 크기가 동적으로 조절되어 좌표가 저장되는 방법이다. 자세히 표현할 것은 큰 자료 형에, 중요하지 않은 것은 작은 자료 형에 저장하여 효율적으로 사용한다.

Prediction[2]방법은 평행사변형의 성질을 이용하여 삼각형의 세 점으로 가상의 한 점을 계산하고 그 삼각형에 실제로 연결되어 있는 정점의 값을 빼서 그 차이 값만큼 표시하여 저장 공간의 크기를 줄이는 방법이다. 보통의 정점들은 이러한 평행사변형의 법칙으로 가까운 위치에 있기 때문에 작은 범위의 값으로도 해당 정점을 표현할 수 있다.

본 논문에서는 3차원 형상 자료를 움직임의 특징에 따라 다양한 그룹으로 만들고 Prediction 방법을 약간 변형하여 그룹의 중심점을 기준으로 시간이 바뀌었을 때 바뀐 시간의 한 정점이 전 시간에서 얼마나 차이가 발생했는지를 표시하여 데이터가 표현되는 범위를 줄였으며, 그 차이에 대하여 효율적인 저장 공간을 정하여 압축을 실시하였다.

2.2 3차원 데이터의 애니메이션 압축

Lawrence Ibarria 와 Jarek Rossignac 은 Dynapack[3]이라는 방법을 통해 3차원 데이터를 Edge-breaker를 사용하여 공간상의 정보를 확인한 후 시간과 공간에 각각 Prediction 방법을 사용하여 압축을 하였다.

Zacki Karni와 Crag Gotsman은 PCA를 통한 패턴 분류 방법과 LPC를 이용하여 시간적인 흐름에 따라 정점의 위치를 예측하는 방법을 통해 3차원 정보를 압축하였다[4]. 위의 PCA를 통해 3차원 정보가 가지고 있는 벡터 중 특징있는 벡터를 뽑아내어 자료를 압축하였으며, 추가로 선형 예측 방법인 LPC를 통해 애니메이션 정보를 압축하여 높은 압축

률을 보였다. 또한 별도의 에러 확인 법을 제시하여 자신들의 방법이 얼마나 오차가 나는지도 보여주었다.

위의 두 논문은 3차원 애니메이션 데이터를 시간과 공간에 관련해서 압축을 하여서 본 논문과 좋은 비교가 된다. Dynapack 같은 경우 상대적으로 오래된 논문으로 현재 나온 논문에 비해 낮은 성능을 보인다. 또한 Zacki Karni 와 Craig Gotsman같은 경우 PCA 의 방법이 많은 샘플들을 가지고 있어야 하는데 "dolphin" 같은 애니메이션의 경우 프레임 수가 적어서 제대로 적용되지 못한다.

본 논문은 위에서 Dynapack 보다 훨씬 좋은 압축률을 보이며 Zacki Karni 와 Craig Gotsman의 PCA 와 LPC의 방법과 비교하여 낮은 에러율을 보이고자 한다.

2.3 형상 정보의 변환에서의 Error 측정법

3차원 데이터의 오차를 확인하는 방법은 원본과 디코딩된 결과를 비교하여 차를 구하거나 얼마나 많이 변하였는지 측정하는 방법이 있다. 하지만 애니메이션에 적용하기에는 단순한 프레임의 비교로는 정확하지 못하기 때문에 본 논문에서는 다음의 방법을 사용하고자 한다. KG error 측정법은 Zacki Karni 의 논문에서 나온 방법으로 다음과 같다.

$$e = \frac{\|A - A'\|}{\|A - E(A)\|} \qquad [식\ 1]$$

여기서 || A || 는 A 의 길이 norm 값. A는 기존의 벡터들, A'는 압축후 다시 디코딩된 벡터들, E(A) 는 시간에 따른 좌표들의 평균을 나타낸다.

이외에 Objective Measure for Position Interpolator[5] 이 있다. 이 방법은 원본을 기준으로 디코딩된 값이 갑자기 높아지거나 낮아질 때를 구간으로 나눈 후, 각 구간을 계산하여 얼마나 차이가 나는지 계산하는 방법이다. 이 방법은 MPEG-4 AFX에서 사용하였지만 그 접근 방법과 성능에 있어 전술된 두 방법과 본질 적인 차이는 보여주지 않는다.

본 논문에서는 Zacki Karni 의 논문과의 직접적인 비교를 위해 KG error 방법을 사용하였다.

3. 군집화 기반 압축 시스템

3.1 변형(Deformation)과 군집화(Grouping)

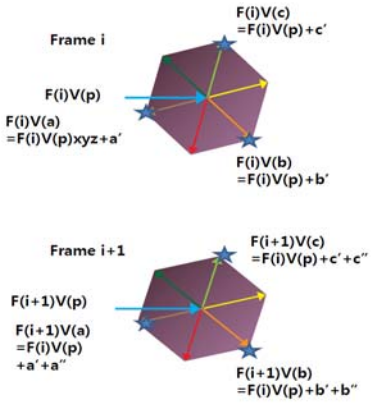
3차원 데이터가 많은 분야에 응용되면서 개발자들이 3차원 데이터를 있는 그대로 사용하지 않고 개발자의 의도에 따라 여러 가지 방법으로 변형하는 작업을 수행한다. 원래의 모습이 이동, 회전되거나 외부의 힘에 의해 변형되기도 하는데 이러한 경우를 변형이라 한다.

스키닝 메쉬 애니메이션[6]의 경우 물체의 기준이 되는 뼈를 중심으로 3차원 데이터를 변형하여 애니메이션을 표현한다. 실제 옷이 움직이는 모습을 시뮬레이션 하는 경우 변형은 비슷한 움직임을 가진 부분을 그룹지어 계산하기 때문에 자연스런 움직임을 보여준다[7].

3차원 데이터에 움직임이 유사한 정점을 모으면 유사한 움직임을 갖는 그룹이 만들어진다. 이러한 그룹 안의 정점들은 서로 비슷하기 때문에 그룹의 기준과 정점을 비교해서 약간의 차이 값만으로도 정점을 나타내는 것이 가능하다. 본 논문에서는 3차원 데이터와 이 데이터의 애니메이션을 손상시키지 않고 압축을 실시하기 위해서 이러한 변형의 유사성을 활용하였다.

3.2 접근 방법

군집화를 사용함으로써 시간과 공간에 따라 유사한 성질을 가지는 정점들을 하나의 그룹으로 묶을 수 있으며, 정점들을 그룹의 기준이 되는 정점의 움직임을 통해 동일 그룹 내의 다른 정점들을 작은 차이를 써서 나타낼 수 있다. 또한 애니메이션의 경우도 전 프레임의 값에 더 작은 차이 값을 더하여 해당 정점을 표현할 수 있다. [그림 3]에서 f(i)는 i 번째 프레임을, v(n)는 n번째 정점을 나타낸다. i 프레임에서의 a,b,c 의 값은 기준 정점에서 좌표 차이 만큼의 값을 더해 표시하며 i+1 번째 프레임에서의 a,b,c 의 값은 전 프레임에서 작은 값을 더해서 표시한다.



[그림 3] 전 프레임과 현 프레임의 작은 차이로 표현되는 그룹 내의 정점의 모습

본 논문에서는 이러한 군집화를 기반으로 3차원 데이터를 압축한다.

3차원 정보의 공간에 관련된 압축은 다음과 같다. 다음 장에 소개할 리전 그로잉[8]을 기반으로 한 군집화를 수행하고, 그 결과에 prediction방법을 수행하여 작은 범위 안에

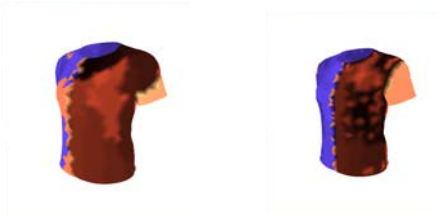
포함하게 한 후 정규화 한다. 이러한 정규화를 거쳐 기존의 자료는 보다 작은 저장 공간에 저장한다. 다음으로 시간과 관련된 압축을 실시하는 보간법을 사용하였다. [그림 4]는 이러한 시스템 구성도이다.

[그림 4] 군집화 기반 압축 시스템 구성도

4. 압축 시스템 구현

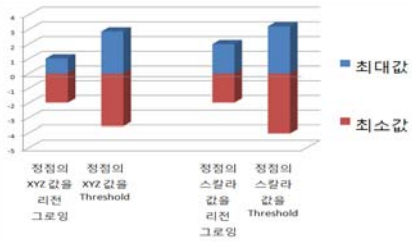
4.1 리전 그로잉 기반 군집화

리전 그로잉은 기준을 중심으로 유사한 성질을 가진 영역을 점차 키우는 방법이다. 리전 그로잉을 하는 이유는 보다 유사한 영역을 체계적으로 모아 군집화를 수행할 수 있기 때문이다. 전체 영역에서 정점의 속성에 따라 단순 Threshold를 적용할 경우 몇몇 정점은 기준 정점과 큰 차이를 내는 반면 리전 그로잉을 사용하면 기준 정점으로 그 정점을 비교할 때 작은 차이가 낸다. [그림 5]를 보면 단순 Threshold의 경우 같은 색이 띄엄띄엄 떨어져 있는 모습을, 리전 그로잉의 경우 비슷한 색이 서로 연결되어 색이 뭉쳐 있는 것을 확인 할 수 있다. 리전 그로잉의 사용은 본 논문의 그룹을 통한 압축에서 보다 작은 오차가 나타나게 한다. [그림 6]이 그 차이 값을 보여주고 있다.



(a)리전그로잉 결과 (b)단순 Threshold

[그림 5] 리전 그로잉과 단순 Threshold 결과 모습



[그림 6] 리전그로잉과 단순 Threshold를 적용한 경우의 정점 위치차이의 최대, 최소값

이러한 리전 그로잉을 수행하기 위해서는 정점의 특징을 뽑아서 비교해야 한다. 정점의 특징을 뽑아내기 위해서 다음의 3가지 방법을 사용하였다. 정점의 움직임을 벡터로 보고 벡터의 길이를 비교하는 방법, 좌표계에서의 정점의 움직임을 비교하는 방법, 퓨리에 변환을 통해 전체 프레임 동안의 정점의 움직임을 상수로 표시, 비교하는 방법을 사용하였다.

4.1.1 벡터의 크기 값을 기준

벡터의 크기 값은 그 벡터가 어느 정도 움직이는지를 나타내는 운동량이기 때문에 각 정점의 벡터의 크기 값은 정점의 운동량이 다른 정점들에 비해 얼마나 차이가 나는지 알려준다. 또한 움직임의 방향이 어느 정도 다르더라도 전체적인 운동량이 비슷하면 비슷한 값을 가지기 때문에 기준 정점과 비교되는 정점의 운동 방향이 다르더라도 운동량이 비슷한 경우에 그룹에 선택될 수 있다.

4.1.2 X, Y, Z 좌표 값을 기준

프레임 별로 나온 정점의 XYZ좌표의 차이를 다른 정점들과 비교하여 두 개의 좌표축의 값이 비슷하더라도 한가지 좌표축에서 다른 값이 있으면 그룹에 포함시키지 않고 X,Y,Z축의 값을 모두 만족시켜야 그룹에 포함시킨다.

각 좌표의 값을 비교하는 이 방법은 전체 프레임동안 한 프레임이라도 한 좌표 축이라도 틀리면 다른 그룹에 들어가기 때문에 자세하고 정확하게 그룹을 나눌 수 있다.

4.1.3 퓨리에 변환을 기준

퓨리에 변환은 연속된 값을 주파수로 보고 이런 주파수를 상수로 표시하는 방법이다[9]. 즉 퓨리에 변환은 각 프레임별로 나온 정점의 좌표별 차이 값을 주파수로 보고 n차 퓨리에 상수로 변환시킨다.

퓨리에 상수는 주어진 배열 주파수적 성질을 나타내며 상수의 차수 n의 값이 작을수록 주파수의 전체적인 모습을, n 의 숫자가 커질수록 주파수의 작은 부분을 나타낸다. 퓨리에 상수의 주파수적 특징을 이용하여 정점들의 움직임과 특징이 다른 정점과 얼마나 유사한지 측정할 수 있다. 본 논문에서는 3차 퓨리에 상수 까지 사용하여 정점의 움직임과 특징을 비교하였다.

4.2 저장 공간의 변화를 통한 압축

4.2.1 인코딩

3차원 데이터가 리전 그로잉을 통해 여러 그룹으로 나뉜 다음 각 그룹의 정점들은 기준 정점과 비교하여 작은 차이를 가지며 이 차이를 이용해서 데이터가 압축된다.

step 1 : 각 정점들마다 시간을 기준으로 해서 앞 프레임과 다음 프레임의 차를 구한다. 전 프레임과 현 프레임의 차이 값 Vn^i 는 [식 2]과 같다.

$i =$ 프레임 번호 , $n =$ 정점의 번호 [식 2]

$\|f(i)v(n)\| =$ 현재 프레임의 정점의 값

$\|f(i-1)v(n)\| =$ 전 프레임의 정점의 값

$$V_n{}^i = \|f(i)v(n)\| - \|f(i-1)v(n)\|$$

step 2 : 나누어진 그룹에서 Threshold 값에 가장 가까운 정점을 기준 정점으로 정하고 그룹 내의 다른 정점들과 비교한다. 현재 프레임이 i, 기준 정점이 p, 해당 정점이 n일 때, 기준 정점 Vp^i, 해당 정점Vn^i 에 대하여 군집화에 쓰일 값 Rn^i 는 기준 정점과 해당 정점의 값을 뺀 차이로 구한다. 리전 그로잉을 통해 계산 되어 매우 작은 값이 나온다.

$$\{ R_n{}^i = V_p{}^i - V_n{}^i \} \quad [식 3]$$

4.2.2 저장 상태

보통 3차원의 데이터를 표현할 때 각 좌표 별로 4바이트의 Float 형을 사용한다. Float 는 2^{32} 승까지 표현이 가능하며 소수점 6자리까지의 표현을 할 수 있다. 하지만 실험에서 리전 그로잉을 통해 생성된 값은 작은 범위 값이 사용되므로 큰 크기의 Float 는 저장 공간의 낭비를 가져온다. 본 논문에서는 군집화한 결과를 가급적이면 작은 범위의 저장 공간에 저장하여 저장 공간의 낭비를 막아 압축을 하고자 한다. 이런 방식으로2^{32}에 저장했던 기존의 방법에 비해 2^n 승의 저장 공간에 데이터를 저장함으로써 저장 공간의 크기를 줄일 수 있게 된다. 그 과정은 다음과 같다.

step 1 : 군집화의 결과 값 Rni의 최대값(MAX)과 최소값(MIN)을 구한다.

step 2 : 최대값과 최소값을 이용해서 정규화 한다. 이 정규화는 Rn^i의 값을 0과 1사이의 값으로 만든다.

$$R_n{}^i = (R_n{}^i - MIN) * (MAX - MIN) \quad [식 4]$$

$$then \ 0 \leq R_n{}^i \leq 1$$

step 3 : Rni의 값에 저장 공간 2n 의 값을 곱하여 최대 0에서 2n-1사이의 값을 가지게 한다.

저장공간의 비트수 $= N$ [식 5]

$$R_n{}^i = R_n{}^i * 2^N$$

$$then \ 0 \leq R_n{}^i \leq 2^n - 1$$

step 4 : 기준 정점은 다른 정점들과의 비교 때문에 Float 형으로 ,그룹의 나머지 정점들은 2^n 의 크기를 가진 자료 형으로 저장한다.

4.2.3 디코딩

디코딩은 모든 정점들이 Float로 표시된 시작 프레임과 군집화 인코딩을 거쳐 나온 값을 통해 수행된다.

step 1 : 시작 프레임의 정점의 위치 값을 저장한다.

step 2 : 기준 정점은 다른 변형 없이 그대로 전송되기에 Vp^i의 값을 알 수 있으며, Rn^i의 값은 인코딩된 값이다.

step 3 : Vn^i 의 값은 현재 프레임과 전 프레임의 차이이며 전 프레임은 이미 알고 있다. 따라서 현재 프레임에서 정점n의 좌표 값을 구할 수 있다. [식 6]가 위의 내용이다.

$$V_n{}^i = V_p{}^i - R_n{}^i \quad [식 6]$$

and

$$V_n{}^i = (\|f(i)v(n)\| - \|f(i-1)v(n)\|)$$

$$then \ \|f(i)v(n)\| = V_p{}^i - R_n{}^i + \|f(i-1)v(n)\|$$

4.3 보간법을 통한 압축

4.2에서 한 프레임에서 각 정점들의 위치를 압축하는 방법을 보았다. 이와 같은 방법은 시간과 공간 중 공간을 압축하는 방법에 속한다. 여기에 보간법[5]을 통하여 시간과 관련된 내용을 압축하고자 한다.

군집화를 통한 압축 방법은 각 그룹의 기준 점이 있기 때문에 그 기준점을 중심으로 다른 정점들을 표시한다. 여기에 더욱 압축률을 높이기 위해 모든 시간마다 정점들을 표시하지 않고 일정 간격을 두고 프레임을 표시하는 방법을 통해 보간법을 구현하였다. 각 그룹마다 기준 점이 있기 때문에 보간을 통해서 애니메이션 정보를 표시하는 프레임의 수가 줄어들어도 큰 에러가 발생하지 않는다.

[식 7]은 이러한 보간법을 수행하는 코드를 보여준다. 보간되는 구역의 첫 프레임과 마지막 프레임을 알고 있을 때 이 두 프레임의 상대적인 영향력을 계산하여 중간 프레임의 값을 구하게 되는 것이다.

$h =$ 기준점에서 떨어진 거리 [식 7]
$framenumber =$ 보간으로 표시하는 프레임 간격

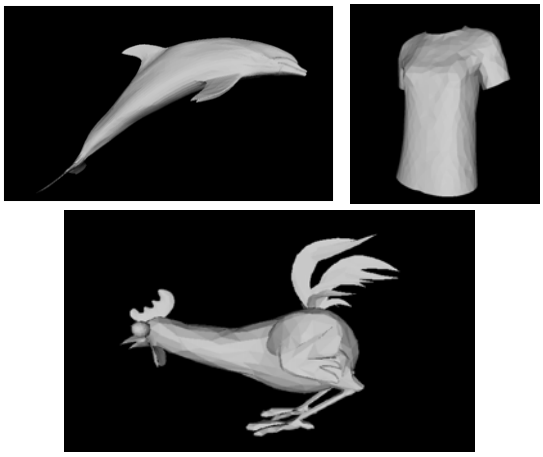
$$f(i+h)v(n) = f(i)v(n) * (framenumber - h)/framenumber + f(i+framenumber)v(n) * (h)/framenumber$$

5. 결과 비교

5.1 실험 데이터

본 논문에서는 펜티엄 QuadCore 6600 CPU와 2GB 메모리, 그리고 GeForce 8600의 PC에서 실험을 수행하였다.

그리고 실험 모델로 3개의 실험 모델을 비교하였다. “cloth"(924정점, 1729페이스, 324프레임)와 "dolphin"(6179정점, 12337페이스, 101프레임), "chicken" (2916정점, 5454 페이스, 399 프레임)데이터이다. 이 데이터는 애니메이션의 특징을 비교할 수 있다는 것에 큰 의미가 있다. "cloth"의 경우 직선 움직임을 한후 회전을 하며, “dolphin"은 한 방향으로 웨이브를 타며 이동을 한다. ” chicken"은 직선 움직임을 보인 후 크게 놀라 몸이 펴지며 다시 움직인다.



[그림 7] 시계 방향으로 “dolphin”, “cloth”, “chicken” animation data

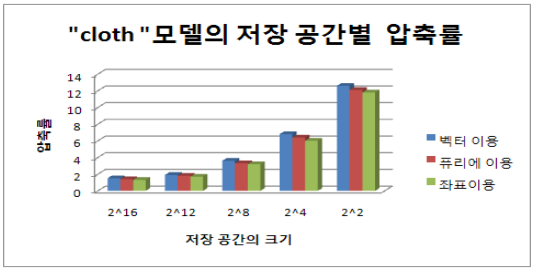
5.2 군집화 결과 분석

군집화를 통해 3차원 모델 데이터를 여러 그룹으로 나누어 압축하였고 그 결과는 [표 1],[그림 7]과 같다.

퓨리에 변환의 경우 웨이브로 움직이는 "dolphin"애니메이션에서 큰 자료 범위를 나타냈다. 퓨리에 변환의 경우 모델 데이터에 따라 큰 성능 차이를 보이며 중간 이하의 성능을 보였다. 퓨리에 변환이 전체적인 움직임을 확인하는데 같은 정점의 움직임을 보이는 모든 정점들이 많으면 정점들이 뭉쳐서 그룹의 수가 적어지고 군집화 결과의 범위가 커지게 된 경우가 있기 때문이다.

좌표의 벡터 크기를 비교하는 방법은 큰 에러를 보이나 그룹의 수가 많아 압축률이 높은 경우를 보인다. 그러나 “chicken"의 경우 좌표 비교와 좌표의 벡터 값을 비교하는 경우 유사한 그룹 수를 나타낼 때 좌표 비교의 경우 군집화 결과의 범위가 더 작다. 이 경우는 벡터 값을 비교한 경우 결과의 범위는 크고 그룹의 개수가 같기 때문이다 .

좌표를 비교한 군집화 방법은 낮은 에러율과 항상 어느정도 이상의 압축률을 나타낸다. 매우 높은 압축률을 낼수는 없지만 항상 안정적인 군집화를 이루며 군집화의 결과 범위가 일정하다. 본 논문에서는 이 방법을 기준으로 3가지 애니메이션 데이터에 보간법을 실시하였다.



[그림 7] “cloth" 모델의 저장 공간별 압축률

5.3 보간법을 적용한 결과 분석

다음으로 시간과 관련된 압축을 하기 위하여 보간법을 통해 압축을 실시하고 그 결과를 Zacki Karni 와 Craig Gotsman의 논문과 비교해 보았다.

먼저 보간법을 통하여 압축을 한 경우 XYZ의 좌표를 비교하며 각 저장 공간마다 각 프레임별로 나온 결과가 [그림 8]이다. 8비트의 저장공간을 이용하고 보간법을 사용하지 않은 경우 약 3.6배의 압축률이 나오지만 보간법을 사용하면 2배가량씩 압축률이 증가하여 2프레임씩 읽었을때는 6.9배, 4프레임씩 읽었을 때는 12.6배, 8프레임씩 읽었을때는 22.3배의 압축률을 보이고 있다.

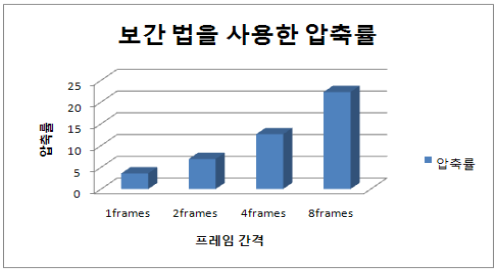
군집화특징		cloth	dolphin	chicken
좌표비교	위치변화 최대값	1.02	1.26	0.893
	위치변화 최소값	−1.09	−1.28	−0.852
	그룹 수	94	118	46
벡터 크기	위치변화 최대값	2.04	2.27	1.15
	위치변화최소값	−2.08	−2.34	−0.85
	그룹 수	60	75	45
퓨리에변환	위치변화 최대값	1.46	3.15	0.739
	위치변화 최소값	−1.844	−3.6	−1.483
	그룹 수	86	16	113

[표 1] 군집화 특징에 따른 결과치의 범위와 그룹의 개수

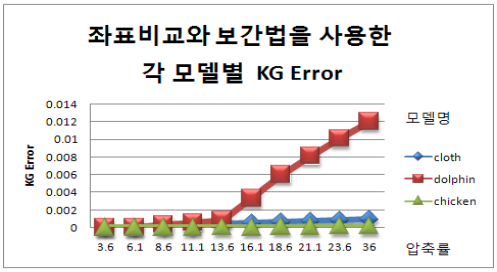
또한 3개의 애니메이션 데이터를 대상으로 KG Error 이 가장 낮은 방법인 정점의 좌표 값을 이용하는 방법을 적용하여 2의 16,8,4,2 배의 저장 공간에 데이터를 압축하여 압축률과 KG Error을 살펴보았다.

웨이브로 움직이는 "dolphin"의 KG Error 이 가장 크게 나왔다. 나머지 모델에서는 극히 작은 에러만이 나오는 것을 확인하였다. [그림 9]에서 " cloth" 와 "chicken"이 매우 작은 에러를 보인다. 또한 Zacki Karni 와 Craig Gotsman의 논문에서는 짧은 프레임 동안의 데이터에서는 나쁜 압축률이 나왔지만 본 논문에서는 프레임의 길이에 상관 없은 압축률이 나온다.

이와 같은 결과를 바탕으로 기존의 Zacki Karni 와 Craig Gotsman 의 논문과 비교하면 압축률에서 뛰어나며 상대적으로 에러가 작은 것을 확인할 수 있다. [표 2]은 Zacki Karni 와 Craig Gotsman의 압축결과이고 Zacki Karni 와 Craig Gotsman의 방법과 본 논문의 압축률과 에러를 각각 비교한 것은 [그림 10]이다.



[그림 8] 보간법을 사용한 압축률

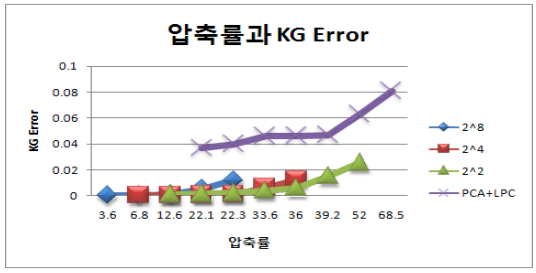


[그림 9] 좌표비교와 보간법을 사용한 모델별 KG Error

방법	압축률	KG Error
PCA+LPC	22.1	0.037
	33.6	0.046
	68.5	0.081
군집화 + 보간법	12.6	0.000757
	22.3	0.001869
	36	0.005687
	52	0.025379

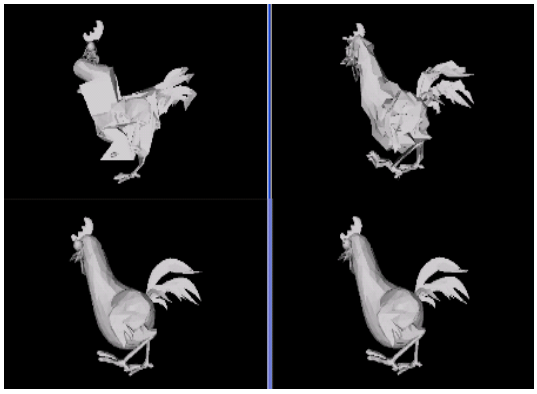
[표 2] Zacki Karni의 압축 결과와 2^2의 저장 공간에 보간법을 적용한 "chicken" animation 의 압축 결과 비교

이와 같은 결과를 통해 본 논문에서 구현한 리전 그로잉을 통한 군집화, 새로운 저장 공간에 군집화된 정점들을 저장, 저장된 프레임들의 보간을 통한 압축 방법이 우수하다는 것을 확인할 수 있다.



[그림 10] "dolphin"모델의 저장공간별 압축률과 KG Error

[그림 11] 좌측의 원본과 우측의 8비트로 좌표비교 압축한 "chicken" animation 10,150,330 frame의 모습



[그림 12] 2, 4, 8, 16 비트로 좌표비교 압축한 "chicken" animation 150 frame의 모습

6. 결론

많은 기술의 발전으로 각종 휴대용 기기들이 발전하면서 많은 분야에서 3차원 모델을 사용하고 있으며 이런 자료들은 큰 크기의 자료를 가지게 된다. 이런 대용량의 자료를 사용하기 위해서는 데이터 전송속도가 확보되어야 함은 물론 휴대용 정보 단말 기기에서 많은 저장 공간을 필요로 하게 된다. 따라서 효율적인 3차원 데이터 압축 방법의 필요성이 대두하였다.

본 논문에서는 더 좋은 압축 효율과 더 적은 손실을 목표로 3차원 모델 데이터를 압축하는 방법을 소개하였다. 모든 프레임마다 각각의 정점을 벡터의 크기 값, XYZ별 좌표 값 그리고 퓨리에 변환을 이용하여 그룹을 지었다. 그리고 이런 그룹을 바탕으로 정점의 데이터를 보다 작은 자료 형으로 변환하여 압축을 하였다. 그리고 보간법을 이용하여 시간에 대한 압축을 실시하였다. 이와 같은 군집화를 통하여 군집이 생성되었을 때 정점의 좌표를 비교하는 방법이 에러는 낮지만 압축률도 낮으며 정점의 벡터 크기를 비교한 방법이 에러도 높고 압축률도 높은 것을 확인하였다. 퓨리에 변환의 경우 모델에 따라 군집이 너무 크게 생성되는 경우 오히려 에러가 커지는 것을 확인하였다.

시간 축에서는 프레임 별 좌표의 차를, 공간 축에서는 같은 그룹에 속해있는 기준 정점과 비교하였다. 시간과 공간 양쪽을 다 처리하는 압축을 하여 별도의 손실이 없이 저장 크기의 변환을 통해 전체데이터의 22배까지 압축하는 결과를 보였다. 관련 연구에서 언급한 Dynapack 보다 월등히 좋은 압축률을 보이며 Zacki Karni 와 Craig Gotsman의 PCA+LPC논문과 비교하여 낮은 에러율을 보인다. 기존의 3차원 애니메이션을 압축 하는 논문과 비교하여 높은 압축률에서 낮은 오차를 보여서 매우 뛰어난 성능을 가지고 있다는 것을 보여주고 있다.

향후 과제로써 본 논문에서 사용한 애니메이션 데이터 이외의 다양한 자료에 압축을 실시하여 확장성을 인정받고 다양한 분야에서 사용될 수 있어야 하겠다. 또한 기존의 3차원 자료에서 연결관계와 관련된 압축을 추가하여 보다 높은 압축률을 낼 수 있도록 발전하여야겠다. 그리고 본 시스템이 오차를 보정하고 압축을 하는데 3~8초, 디코딩을 하는데 2초 정도가 걸린다. 모바일에서 사용될 경우 압축을 푸는 시간이 더 많이 걸릴 것이기에 이런 부분에 대한 수정도 있어야 하겠다.

또한 정점의 저장 공간을 줄이고 보간법을 사용하여도 PCA+LPC의 최대 압축률의 성능을 보일수가 없었다. 저장 공간을 2바이트로 하고 8프레임마다 보간을 하였을 경우 52배의 압축률을 보인다. 물론 이러한 압축률에서 작은 PCA+LPC에 비해 작은 KG Error을 보인다.

이제 본격적으로 다가오는 유비쿼터스 컴퓨팅 환경을 맞아 많은 3차원 컨텐츠들이 생성되고 있다. 이와 같은 압축 방법을 활용하게 되면, 큰 크기의 3차원 데이터가 전송 시간의 부담이나 메모리 공간의 부담이 없이 사용될 것이다. 3차원 콘텐츠가 사용자들에게 보다 편하고 자연스럽게 접근하고 널리 쓰이게 될 것으로 기대한다.

참고 문헌

[1] M. Eck, T. Derose, T. Duchamp, H. Hoppe, M. Lounsbery AND W. Stuetzle, "Multiresolution analysis of arbitrary meshes", In R. Cook, editor, SIGGRAPH 95 Conference Proceedings, pp. 173-182. 1995.

[2] M. Isenburg AND P. Alliez, "Compressing polygon mesh geometry withparallelogram prediction", In Visualization' 02 Conference Proceedings, pp. 141-146, 2002.

[3] L. Ibarria AND J. Rossignac. "Dynapack: space-time compression of the 3D animations of triangle meshes with fixed connectivity", ACM Symp. Computer Animation, pp 126-135, 2003.

[4] Z. Karni AND C. Gotsman. "Compression of soft-body animation sequences", Computers and Graphics, pp. 25-34, 2004.

[5] Information Technology, Coding of Audio-Visual Objects, Part 11: SL Extension and Multi-User Worlds, ISO/IEC 14496-11:2003/Amd.1:2003(E).

[6] D. L. James AND C. D. Twigg, "Skinning mesh animations", ACM Transactions on Graphics Vol 24, no 3, pp. 399-407, 2005.

[7] L. Kavan, R. McDonnell, S. Dobbyn, J. Ž ra AND C. O'Sullivan, "Skinning arbitrary deformations", In Proceedings of the 2007 Symposium on interactive 3D Graphics and Games, pp. 53-60, 2007.

[8] R. C. Gonzalez AND R. E. Woods, Digital Image Processing, Addison Wesley,Reading, MA, 1992.

[9] W. H.Press, S. A. Teukolsky, W. T. Vetterling, AND B. P. Flannery, Numerical Recipes in C,The Art of Sci. Computing, Cambridge U. P., 1992.