동작 전처리 기법을 활용한 실시간 군중 애니메이션 시스템

(A Real-time System of Crowd Animation with Motion Pre-processing Method)

안 정 현 † 원 광 연 ††
(Junghyun Ahn) (Kwangyun Wohn)

요 약 군중 애니메이션을 위한 연구분야는 동작의 사실성 제공과 속도향상의 두 가지 측면으로 구분된다. 최근 엔터테인먼트 산업의 발달로 군중 애니메이션은 디지털 콘텐츠 제작의 필수요소로 자리 잡았으며, 애니메이션의 속도향상을 위한 노력이 계속되고 있다. 이러한 추세에 맞춰, 본 논문에서는 군중 애니메이션의 속도저하에 영향을 미치는 요인들을 실험을 통해 검증하고, 실험 분석을 토대로 전처리 시스템을 설계한다. 동작 전처리 모듈은 제시된 애니메이션 시스템의 핵심 모듈로서 관절의 개수를 동작의 상세도에 따라 다양한 레벨(level-of-detail)의 동작들로 재구성한다. 실제 애니메이션 과정에서는 카메라와 가상환경 내부에 존재하는 캐릭터의 관계를 토대로 적절한 레벨의 동작을 적용하여 전체 시스템의 속도를 향상시킨다. 동작전처리 방법(motion level-of-detail)은 동작분석과 동작단순화 과정으로 분류되는데, 비선형 최적화 방법의 하나인 SQP를 활용하여 다양한 상세도 레벨의 동작들을 계산한다. 본 논문에 제시한 전처리 방법은 동작편집 등 캐릭터 애니메이션의 다양한 분야에 유용하게 활용될 수 있다.

키워드 : 동작 전처리, 동작 단순화 기법, 군중 애니메이션, 비선형 최적화

Abstract Research field on crowd animation can be classified into two major categories. One is to offer realism of the crowd motion and the other is to improve speed of the animation. For the last decade, a lot of research on realism and behavior of crowd have been presented. But lately, research on improving speed seems like more interesting. Therefore, in this paper, we conducted an experiment to analyze what is the main bottleneck of crowd animation. As the result, we find out one of the most important bottleneck is the number of joints transformed in each animation frame. In order to resolve this problem we propose a novel level-of-detail technique 'motion level-of-detail', which is a joint-reduction technique operated in the pre-processing time. We used a non-linear optimization, SQP (sequential quadric programming), to generate the low detailed motions

Key words : Motion simplification, Level-of-detail, Crowd animation, Nonlinear optimization

1. 서 론

디지털 콘텐츠 제작에 있어 군중 애니메이션은 최근 중요한 연구분야 중 하나로 자리잡았다. 수년 전부터 군중 애니메이션 관련연구들이 세계적으로 다수 소개되었는데, 동작의 사실성을 제공하는 연구와 속도향상을 위한 연구, 두 가지로 분류할 수 있다. 본 논문은 군중 동작의 사실성 보다 속도향상 측면에 초점을 맞춘 군중 시스템 설계에 대해 기술하려 한다.

군중 애니메이션의 속도향상을 위한 기존 연구로는 가상캐릭터 위쪽 반구 영역에 카메라를 임의로 배치시켜 각 카메라로부터 동작의 각 프레임을 이미지로 저장하고, 실시간에 캐릭터와 카메라의 위치관계에 따라 적절한 이미지를 연속적으로 교체해주는 임포스터(impostor) [1-3]방법이 군중 애니메이션의 속도를 가장 빠르게 할 수 있는 기술로 평가된다. 그러나, 아래와 같은 세 가지 문제점을 발생시킨다.

1. 사실성(Realism): 카메라의 움직임이 빠르거나 캐릭터에 근접할 경우 동작의 사실성이 떨어짐
2. 저장공간(Space): 이미지를 저장할 수 있는 다량의

· 본 연구는 정보통신부, 정보통신연구진흥원 정보통신선도기반기술개발(2006) 사업의 연구비 지원으로 수행하였습니다.
† 학생회원 : 한국과학기술원 전자전산학과
chocchoggi@vr.kaist.ac.kr
†† 종신회원 : 한국과학기술원 문화기술대학원 원장
wohn@kaist.ac.kr
논문접수 : 2006년 10월 24일
심사완료 : 2006년 12월 11일

메모리 공간 확보의 필요성

3. 상호작용(Interaction): 사용자와 애니메이션 시스템간 상호작용 및 동작편집의 어려움

이러한 문제점을 해결하기 위해 임포스터 방법과 같은 2레벨(기존 3D, 이미지 대체) 애니메이션을 사용하기보다, 관절의 수를 줄인 동작들을 전처리 과정에 미리 계산하여 동작의 상세도를 다수의 레벨로 구성할 수 있다면, 사실성과 실시간성을 동시에 만족 할 수 있는 군중 애니메이션 시스템을 설계할 수 있다. 이처럼 상세도가 점차적으로 줄어드는 동작 데이타들을 전처리 과정에 미리 계산해 두는 경우, 관절의 개수만 줄어든 3차원 애니메이션이기 때문에 동작의 사실성을 유지할 수 있으며, 카메라 샘플링을 통한 이미지 저장 단계가 사라져 많은 량의 데이타를 저장하지 않아도 된다. 또한, 관절을 이용한 애니메이션 방식이 그대로 유지되므로 사용자의 동작편집과 상호작용이 가능하다.

관절의 수를 줄이는 골격구조 단순화(bone LoD) 방법[4-6]도 애니메이션 분야에서 다수 발표되었다. 하지만, 이러한 기존연구의 주된 관심사는 물리기반 시뮬레이션 시간을 줄이는 방법으로 많이 활용되었다. 단순화 방법도 골격구조 자체에 국한 되었으며 동작의 속성은 고려되지 않았다. 게임에서도 애니메이션 속도향상을 위해 관절 수를 여러 단계로 나누는 경우가 있지만, 애니메이터가 수작업으로 많은 시간을 할애하여 가장 적절한 동작레벨을 구성한다. 가장 유사한 연구분야로, 동작의 속도향상을 위해 동작 상세도를 계산하는 방법[7,8]이 있는데, 단순화된 골격의 변환행렬을 메쉬 각 점의 위치, 노멀(normal)로 최적화 하기 때문에 보다 정확한 단순화 결과를 도출할 수 있지만, 전처리 시간이 많이 소요되고 단순화 동작의 재사용이 어렵다. 본 논문에서는, 단순화의 정확도에 비해 전처리 시간을 줄이고 재사용이 가능한 동작 상세도 방법을 소개하고 제안하는 군중 애니메이션 시스템에 어떻게 적용되었는지를 보인다.

본 논문의 구성은 다음과 같다. 우선, 속도의 향상을 위해 관절제거가 가지는 의미와 중요성을 다음 절에 기술한다. 제2장에는 제시하는 군중 시스템의 구성과 각 모듈을 기술한다. 제3, 4장에는 동작 전처리 기법을 동작 분석과 단순화로 분류하여 자세히 기술한다. 마지막으로, 5, 6장에는 실험결과를 보이고 결론을 맺는다.

1.1 관절제거의 중요성

군중 애니메이션에 사용되는 개개의 가상캐릭터는 계층적 관절구조로 이루어지며, 시간에 따른 관절의 변환으로 캐릭터의 움직임을 생성한다. 사실적인 동작을 제공하기 위해 스키닝(skinning)[9,10] 기술이 기본 요소로 널리 활용되고 있다. GPU의 개념이 없었던 과거에는 메쉬(mesh)의 복잡도가 렌더링(rendering) 성능을 좌우하는 주된 요인이었다. 이는 스키닝 과정을 모두 CPU상에서 계산하고 새로 계산된 메쉬를 그래픽 하드웨어로 다시 보내는 과정을 반복하기 때문이다. 하지만, 그래픽스 하드웨어 기술의 발전으로 스키닝 계산이 GPU상에서 가능하게 되었고 CPU는 캐릭터의 초기자세(rest pose)만을 디스플레이 리스트에 저장하여 전달하는 것이 가능하게 되었다. 이에 따라, 군중 애니메이션의 속도향상을 위해 관절 개수를 줄이는 문제의 중요성이 점점 부각되고 있다. 본 절에서는 실험을 통해 관절과 메쉬의 복잡도가 군중 애니메이션의 속도저하에 미치는 영향을 보이려 한다. 이를 위해 그림 1과 같이, 3.0GHz CPU와 GeForce 7800 GTX 하드웨어 환경 하에서 10,000개의 가상 캐릭터에 대해 관절 개수와 메쉬 다각형 개수를 조절하여 애니메이션 속도를 측정하였다.

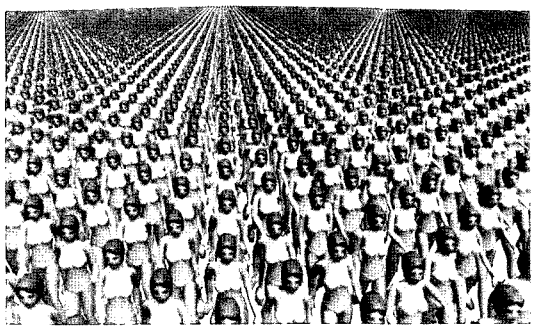


그림 1 가상캐릭터 10,000개의 애니메이션 실험장면 - 상세도가 가장 높은 경우 각각의 캐릭터는 40개의 관절(joints)과 1310개의 다각형(polygons) 그리고 658개의 점(vertices)으로 이루어진다.

실험은 다음과 같이 수행하였다. 개개의 가상캐릭터를 관절 8단계, 다각형 8단계, 총 64개 상세도로 분류하여 각 상세도에 대해 10,000 캐릭터의 애니메이션 속도(fps)를 측정하였다. 추가로 군중 애니메이션의 속도를 가장 높일 수 있는 기술로 평가되는 빌보딩(billboarding) 기법을 구현하여 속도를 측정하였다. 실험의 정확성을 높이기 위해 관절과 다각형의 단순화율 (상세도가 가장 높은 캐릭터에 대해 현재 상세도의 관절개수 또는 다각형의 개수 백분율)을 100, 82, 70, 58, 45, 33, 20, 8%로 일치 시켰다. 실험결과는 그림 2와 같다.

실험결과 관절의 개수 축에서 본 속도 향상이 다각형 개수 축 보다 빠르게 향상되는 것을 확인 할 수 있다. 이는 앞서 기술하였듯이 스키닝 애니메이션을 버텍스 쉐이더(vertex shader) 내부에서 계산하고 애니메이션 실행 시 디스플레이 리스트에 저장된 정적인 초기자세만을 그래픽 하드웨어로 넘겨주기 때문이다. 또한, 3개

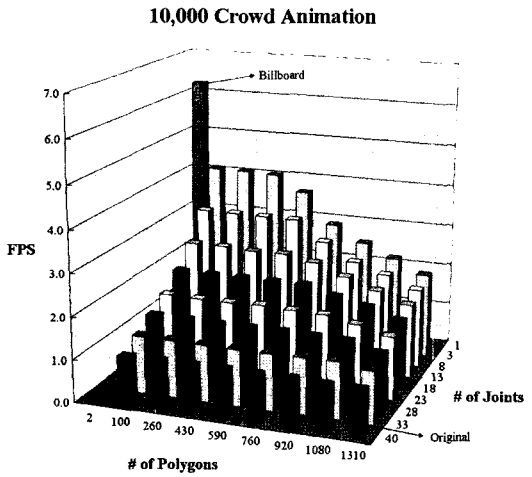


그림 2 관절의 개수 감소와 메쉬 다각형의 개수 감소가 군중 애니메이션 속도에 미치는 영향

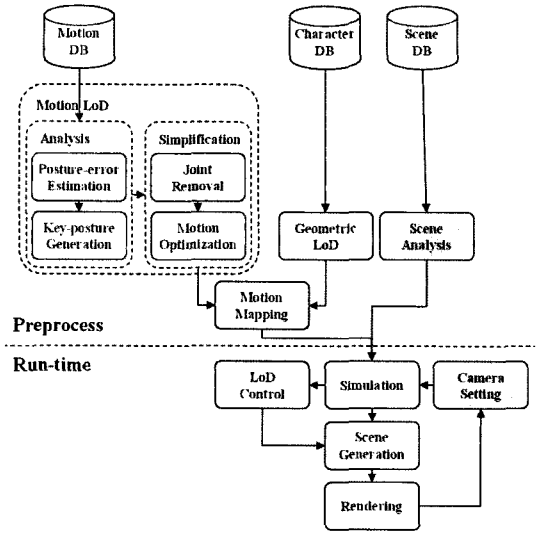


그림 3 군중 애니메이션 시스템 구성도

의 관절과 100개의 다각형으로 구성된 가장 낮은 상세도의 경우 가장 높은 상세도 애니메이션 보다 속도가 4~5배 정도 향상된 것을 확인할 수 있고, 빌보딩 기법만을 이용한 애니메이션에 크게 뒤지지 않는 수준의 속도를 보여준다. 본 실험 결과 군중 애니메이션의 속도향상과 사실성 제공을 위해 관절 제거의 역할이 크게 작용할 것으로 예상된다.

2. 군중 시스템 설계 및 구성

실시간 군중 애니메이션을 위한 전체 시스템 구성은 그림 3과 같고 전처리 과정, 애니메이션 실행 과정으로 분류된다. 애니메이션 실행 전에 필요한 모든 계산을 전처리 과정에 수행하고 분석된 데이타들을 애니메이션 실행 단계에서 활용한다. 본 장에서는 시스템 각 모듈에 대해 기술한다.

2.1 전처리 과정

전처리 과정은 군중 애니메이션에 활용되는 동작과 메쉬의 상세도를 생성하고 각각을 매핑(mapping)한다. 또한, 런타임에 사실적인 애니메이션을 제공하기 위해 배경장면 분석을 수행한다.

동작 전처리(Motion LoD): 동작 전처리 과정은 본 논문의 핵심 모듈로 모션의 관절을 순차적으로 제거하여 다양한 상세도 레벨의 동작을 생성한다. 이 과정은 동작 분석과 동작 단순화 과정으로 분류된다. 동작 분석에서는 관절의 자세간 오차를 계산하는 방법을 정의하고 이를 활용해 동작 전체를 대표하는 자세를 생성한다. 이는 관절이 제거 되었을 때 기존 동작과 최대한 유사한 자세를 유지하기 위함이다. 동작 단순화 과정은 관절 제거 후 골격구조가 다른 동작에 대해 기존 동작과 가장 유사하도록 최적화된 동작을 계산한다. 동작 전처리 방법은 제3, 4장에서 상세히 기술하도록 한다.

메쉬 전처리(Geometric LoD): 가상 캐릭터의 메쉬 단순화를 위해 본 시스템에서는 점진적 메쉬 단순화 기법(progressive mesh)[11,12]을 활용하였다. 가장 낮은 상세도는 임포스터 방법의 이미지로 대체하되 카메라의 위치 별 동작 샘플링은 수행하지 않았다.

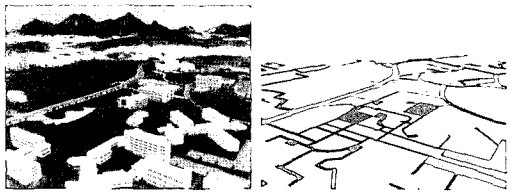
동작 매핑(Motion Mapping): 본 시스템은 캐릭터의 동작과 캐릭터 메쉬를 독립적으로 단순화하기 때문에 각각의 상세도를 생성한 후 메쉬에 동작을 입히는 과정이 필요하다. 이를 위해 동작과 캐릭터의 중심을 일치시키고 대응되는 각 관절의 회전량 차이를 계산하여 최종적으로 가상 캐릭터의 움직임을 계산한다.

장면 분석(Scene Analysis): 캐릭터 활동가능 영역과 높이정보를 전처리 과정에 계산한다. 이를 위해, 장면 위에서 바라 본 깊이 맵(depth map)을 생성한다. 분석 과정은 그림 4와 같다.

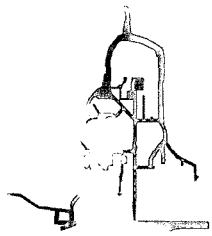
2.2 애니메이션 실행 과정

애니메이션 실행 시에는 우선 카메라의 위치와 방향을 설정한다. 시뮬레이션 모듈에서는 앞서 전처리 과정에서 계산된 데이타와 카메라 설정 정보를 입력 받아 캐릭터의 위치와 높이를 계산하고 애니메이션 할 동작 프레임을 결정한다. 시뮬레이션 모듈에서 계산된 정보는 상세도 제어를 통해 적합한 레벨의 캐릭터를 결정하며 장면 생성 과정에서 자세변환, 스키닝 후 최종적으로 렌더링 단계에 돌입한다. 렌더링 후 카메라 설정으로 돌아가 앞서 기술한 애니메이션 실행 과정을 반복한다.

상세도 제어(LoD Control): 시뮬레이션 과정에 계



(a) 배경 모델 (b) 캐릭터 활동영역 추출



(c) 깊이 맵 생성

그림 4 장면분석 모듈의 깊이 맵 생성 결과 - 흑색에 가까울수록 배경지형의 고도가 높고 백색에 가까울수록 지형의 높이가 낮다. 순백색의 경우 캐릭터의 활동이 불가능한 영역으로 인식된다.

산된 정보는 상세도 제어 모듈에서 캐릭터의 상세도를 결정하기 위한 수단으로 활용된다. 상세도를 낮추어도 시각의 질을 유지하기 위해 가상 캐릭터의 화면상 크기(size), 속도(velocity), 이심률(eccentricity)을 고려했다[8]. 상세도가 가장 낮은 경우 임포스터 방법과 같이 이미지로 대체한다. 또한 영상의 맺힐 캐릭터의 크기가 보이지 않을 정도로 작거나 보이는 영역 밖에 놓일 경우(view frustum culling) 애니메이션 과정에 계산되지 않도록 제거된다.

3. 동작 분석

본 논문의 실시간 군중 애니메이션 시스템은 동작 전처리 모듈을 제공한다. 앞서 기술하였듯이, 동작 전처리 과정에는 관절의 수를 줄이면서 동작의 상세도를 낮춘다. 관절을 제거하기 앞서 전체동작을 대표하는 자세를 생성해 단순화된 동작의 사실성을 높일 수 있다. 이러한 대표자세(key posture)를 계산하기 위해 각 관절의 자세간 오차를 최소화하는 식의 정의가 필요하다. 이를 위해 관절의 자세 군집화(joint posture clustering)[7] 방법에 사용한 오차 식을 활용한다. 자세 군집화 방법은 현재 동작에 포함된 자세들 중 대표자세를 추출하기 때문에 다른 오차 식에 비해 추출된 자세의 사실성이 높다.

3.1 관절의 자세간 오차 정의

자세간 오차는 같은 관절의 서로 다른 시간에서의 자세차이를 물리적인 척도로 계산한 방법이다. 관절 j의 시간 t 자세와 시간 t_{ref} 자세간 오차를 계산하기 위해 그림 5에 도시된 변수들을 활용한다. 관절 j의 지역 좌표계(local coordinate)를 중심으로 시간 t, t_{ref}에서의 하위관절들로 변수들이 설정된다. 자세간 오차는 식 (1)에서 보듯이, 관절의 회전량인 $\theta_j(t_{ref}, t)$와 가중치인 $r_j(t)$의 곱으로 자세간 오차를 정의하는데, 이는 두 자세간 하위관절들의 궤적영역을 의미한다.

$$E_j(t_{ref}, t) = \frac{\left\| r_j(t)\theta_j(t_{ref}, t) \right\|}{r_{max}\,\pi} \tag{1}$$

자세간 오차 식 $E_j(t_{ref}, t)$에서 관절의 가중치 $r_j(t)$는 식 (2)와 같이 정의된다. 관절 j가 하위관절이 많을 경우, 약간의 움직임에도 하위관절들이 많은 영향을 받고 큰 오차가 작용한다. 따라서, 가중치는 관절 j와 각 하위관절 c사이의 거리인 $l_{j,c}(t)$의 합으로 계산된다.

$$r_j(t) = \sum_c l_{j,c}(t) \tag{2}$$

관절의 회전량 $\theta_j(t_{ref}, t)$는 식 (3)과 같이 정의된다. 시간 t에서 관절 j의 하위관절 벡터를 $\mathbf{v}_j(t)$와 시간 t_{ref}의 vj(tref) 사이 각으로 계산된다. 관절의 세그먼트 길이는 변하지 않으므로 아래와 같이 정의된다.

$$\theta_j(t_{ref}, t) = \arccos\left[\frac{\mathbf{v}_j(t_{ref}) \cdot \mathbf{v}_j(t)}{\| l_{j,c}(t) \|^2} \right] \tag{3}$$

마지막으로 자세간 오차를 표준화(normalize)하는 상수인 $r_{max}\pi$는 계층구조, 길이구조가 다른 동작들 사이의 오차를 정량적으로 표현하기 위해 계산된다. 모션캡처된 동작이나, 가상 캐릭터의 초기자세는 일반적으로 양팔을 벌리고 선 T-자세를 활용한다. 따라서, r_{max}는 계층구조 최상위 관절의 $r_j(t)$ 즉, $r_{root}(0)$을 적용한다. 동작의 전처리 과정이 수행되기 전 프레임 0은 T-자세로 설정한다.

3.2 동작의 대표자세 생성

앞 절에서 기술한 자세간 오차 $E_j(t_{ref}, t)$를 토대로 동작을 대표하는 각 관절의 대표 프레임을 추출 할 수 있다. 동작의 대표자세는 다음 장에 기술할 동작 단순화

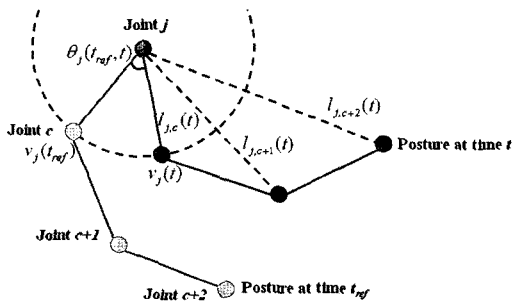
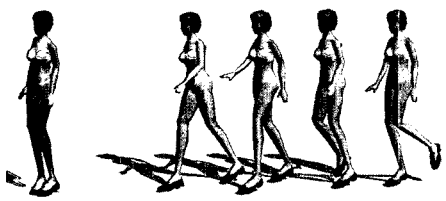


그림 5 관절 j의 시간 t_{ref} 자세와 t 자세의 차이를 계산하기 위해 필요한 속성 정의



Key posture Walking motion

그림 6 (좌) 식 (4)를 통해 걷기 동작의 대표자세를 생성한 결과; (우) 걷기동작의 모습

과정에서 관절을 제거하기 전 가상캐릭터의 초기자세를 변형하는데 활용되며, 이를 통해 기존 동작과 유사한 단순화 결과를 생성하는데 핵심적인 역할을 한다. 각 관절의 대표 프레임은 식 (4)와 같이 추출된다. 관절 j에 대해 동작 프레임을 표현하는 모든 시간 t에 대해 특정 시간 t_{ref} 와의 자세간 오차들의 합을 구한다. 계산결과 중 최소 오차를 가지는 프레임 t_{ref}가 관절 j를 대표하는 프레임이 된다. 최종적으로 동작의 대표자세를 생성하기 위해 각 관절의 대표 프레임의 회전값을 계층구조 순서에 따라 변환한다. 상수 n은 전체 프레임 수를 나타낸다. 동작의 대표자세를 생성한 결과는 그림 6과 같다. 동작의 대표자세 생성 후에는 프레임 0의 T-자세를 대표자세로 대체한다.

$$\arg\min_{t_{ref}}\left[\sum_{t=1}^{n} \mathrm{E}_j(t_{ref}, t)\right] \tag{4}$$

4. 동작 단순화

동작 단순화는 관절제거 과정에 의해 관절구조의 속성이 바뀐 단순화된 동작 자세와 이에 대응되는 기존 동작 자세의 차이를 최소화하는 최적화[13,14] 방법이다. 이를 위해, 앞서 기술한 동작의 대표자세는 가상 캐릭터의 초기자세로 활용되어 단순화된 관절의 오차를 줄이고, 관절제거 후 남아있는 관절들의 각 자세 별 위치를 최적화하여 상세도가 낮은 동작들을 생성한다.

4.1 관절제거

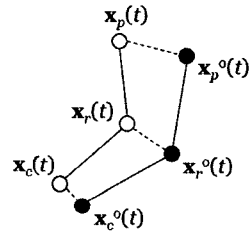
관절제거 과정에는 관절의 제거 우선순위를 결정한다. 관절의 속성 중 동작에 영향을 주는 가장 큰 요소는 회전량과 계층구조상의 위치이다. 즉, 동작의 변화가 적은 관절 또는 계층구조 하위에 있는 관절 일수록 제거 우선순위가 높아진다[8]. 우선순위가 결정되면 제거될 관절 수에 의해 제거 관절 목록을 추출할 수 있다.

대표자세를 기준으로 제거될 관절의 부모-자식관절 사이 거리와 회전각은 다음절에 기술할 최적화 과정에 활용된다. 이는 제거될 관절의 지역좌표계 상의 변환이 항상 일정하도록 하기 위함이고 일정해야 관절 제거가 가능하다. 관절제거는 최적화 과정이 마무리 된 후 수행된다. 관절제거 시 메쉬의 스키닝 정보에 남아있는 제거 관절들은 해당관절의 부모관절로 바뀐다.

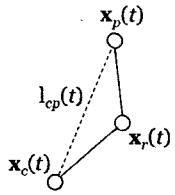
4.2 동작 최적화

동작 최적화에서는 관절제거 과정에서 설정된 정보를 이용하여 기존 동작과 가장 유사한 자세를 계산한다. 이를 위해, 본 논문에서는 비선형 최적화 방법의 하나인 SQP(sequential quadric programming)[13]를 이용해 각 관절의 위치, 길이 그리고 회전각이 기존동작과 유사하도록 최적화 하였다. 식 (5)에서 보듯이 목적함수 E는 관절의 위치 오차 E_d, 길이 오차 E_s 그리고 회전각 오차 E_a로의 합으로 구성된다. 상수 α, β, γ는 각 오차의 가중치를 조절한다. 각 오차가 계산되는 세부변수들은 그림 7에 상세히 도시 및 기술하였다.

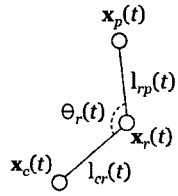
$$\mathrm{E} = \alpha \mathrm{E}_d + \beta \mathrm{E}_s + \gamma \mathrm{E}_a \tag{5}$$



Distance error (E_d)



String error (E_s)



Angle error (E_a)

그림 7 동작 최적화의 목적함수 정의 - $x_p(t)$, $x_r(t)$, $x_c(t)$: 단순화된 계층구조의 부모관절, 제거될 관절, 자식관절 위치벡터; $x_p^o(t)$, $x_r^o(t)$, $x_c^o(t)$: 기존동작의 위치벡터; $l_{cp}(t)$, $l_{cr}(t)$, $l_{rp}(t)$: 두 관절 사이의 길이 (자식-부모, 자식-제거, 제거-부모); $\Theta_r(t)$: 제거될 관절의 회전각(동작 프레임 t)

식 (6)에 정의된 E_d는 목적함수의 위치오차로 상수 n은 기존동작 관절의 개수이다. 상수 f는 동작의 전체 프레임 수다. 루트관절(j=1)의 경우 기존동작의 루트와 위치가 같도록 제약조건(constraint)에 포함시켰다. 위치오차 식에서는 모든 동작 프레임에 대해 단순화된 관절위

치와 대응하는 기존동작 관절위치 차이의 합으로 계산된다.

$$E_d = \sum_{t=1}^{f}\sum_{j=2}^{n} \| \mathbf{x}_j(t) - \mathbf{x}_j^o(t) \|^2 \quad (6)$$

식 (7)에 정의된 E_s는 목적함수의 길이오차로 상수 m은 제거되지 않을 관절의 개수이다. 이러한 관절들로 구성된 계층구조의 각 관절 길이는 동작분석 시 생성된 동작 대표자세에서 $l_{cp}(0)$로 계산된다. 루트관절(c=1)은 부모가 존재하지 않아 길이오차를 계산하지 않는다.

$$E_s = \sum_{t=1}^{f}\sum_{c=2}^{m} [\| \mathbf{x}_p(t) - \mathbf{x}_c(t) \| - l_{cp}(0)]^2 \quad (7)$$

식 (8)에 정의된 E_a는 목적함수의 회전각 오차로 상수 n-m은 제거될 관절의 개수이다. 앞서 관절제거 단계에서 기술하였듯이 제거될 관절은 대표 프레임의 회전각 $\theta_r(0)$을 모든 프레임에 일정하게 유지시켜야 한다. 즉, 회전각 오차는 제거될 관절들의 각만 고려하면 된다.

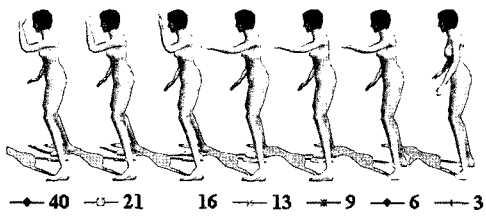
$$E_a = \sum_{t=1}^{f}\sum_{r=1}^{n-m} \left[\frac{[\mathbf{x}_c(t) - \mathbf{x}_r(t)] \cdot [\mathbf{x}_p(t) - \mathbf{x}_r(t)]}{l_{cr}(t)\, l_{rp}(t)} - \cos[\theta_r(0)] \right]^2 \quad (8)$$

최종적으로 SQP를 수행한 결과는 위치, 길이, 회전각 오차를 최소화하는 모든 관절, 모든 프레임의 $x_j(t)$이다. 최적화 결과로부터 동작 데이타를 관절의 위치에 따라 재구성하고 캐릭터에 동작을 입힌 후 제거될 관절을 실제로 제거한다. 캐릭터 메쉬의 초기자세는 동작의 대표자세와 같게 설정되었고 제거될 관절의 회전변환은 일정하게 유지되므로 바로 제거 가능하다.

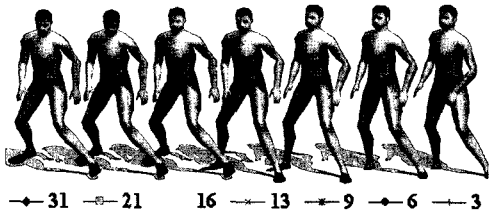
5. 실험 결과

본 논문에 제시한 군중 애니메이션 시스템의 동작 전처리 결과는 그림 8과 같다. 춤(dance), 돌기(turn), 걷기(walk) 동작에 대해 실험을 수행하였다. 동작분석으로 인해 상세도가 낮은 동작도 기존동작과 유사한 것을 확인할 수 있다. 가장 좌측에 있는 자세가 100% 상세도인 기존동작의 자세이고 우측으로 갈수록 관절의 개수가 점점 줄어든 낮은 상세도 자세이다. 상세도가 낮아질수록 관절의 제약이 많아져 기존동작과 점점 다른 자세로 최적화 되는 것을 볼 수 있다.

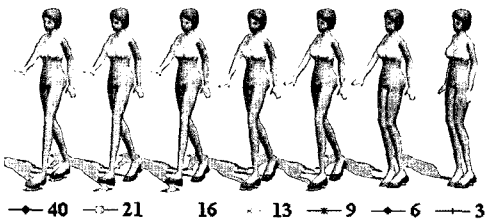
본 논문에 제시한 동작 전처리 기법의 장단점을 살피기 위해 각 동작에 대해 세 가지 동작 상세도 기법을 적용하여 단순화 결과를 정량적으로 분석하였다(표 1 참조). 비교 대상으로 삼은 동작 상세도 기법은 본 논문에 제시한 동작분석 후 SQP로 최적화한 방법(SQP with analysis), 동작분석 없이 최적화한 방법(SQP) 그리고 캐릭터에 입혀진 동작에 대해 메쉬 점 위치, 노멀 값들로 관절변환 행렬을 최소자승(least-square) 근사로 최적화한 방법(VPNO)[8]이다. 각 상세도 방법의 성능



(a) Dance 동작



(b) Turn 동작



(c) Walk 동작

그림 8 세 개의 동작에 대한 관절 개수 별 동작 전처리 결과자세(우측으로 갈수록 상세도 낮은 자세)

을 실험하기 위해 전처리 시간과 전처리 결과의 정확성을 비교하였다. 정확성은 기존동작 메쉬 점들의 위치, 노멀과 대응되는 단순화 동작 점들의 위치, 노멀 차이로 비교하였다. 실험 결과, 정확성은 VPNO에 비해 약간 떨어지지만, 전처리 시간에서 훨씬 우수한 것으로 평가된다. 또한 동작분석에 의해 단순화 된 동작의 정확성을 높일 수 있다.

실시간 군중애니메이션 시스템을 이용한 렌더링 결과는 그림 9와 같다. 카메라로부터 멀리 있는 캐릭터는 낮은 상세도로 애니메이션 되고 가까이 있는 캐릭터는 높은 상세도로 애니메이션 된다. 적절한 상세도 레벨은 앞서 기술한 상세도 제어 모듈에서 결정된다. 속도향상 결과는 다음과 같다. 동작 단순화만 적용한 경우 3~4배, 동작, 메쉬 단순화를 적용한 경우 5~6배, 화면상에 보이지 않는 캐릭터까지 제거하면 평균적으로 10배 이상의 속도향상을 보이는 것으로 확인되었다.

6. 결 론

본 논문에서는 애니메이션 속도향상에 초점을 맞춘

표 1 다양한 동작전처리 기법에 대한 동작 전처리 시간(pre time) 및 정확도(vtx error) 비교 실험 결과 - SQP: 동작분석 없이 동작 최적화를 수행한 결과; SQP with analysis: 본 논문에 제시한 방법; VPNO: [8]에 제시한 방법; Pre time: 전처리 시간; Vtx pos err: 기존동작과 단순화된 동작의 프레임당 평균 위치오차; Vtx nor err: 프레임당 평균 노멀오차

	DANCE Bones: 40, Frames: 822, Vertices: 661				TURN Bones: 31, Frames: 173, Vertices: 559				WALK Bones: 40, Frames: 285, Vertices: 658			
	Bones	Pre time (sec)	Vtx pos err (err sum / fr)	Vtx nor err (err sum / fr)	Bones	Pre time (sec)	Vtx pos err (err sum / fr)	Vtx nor err (err sum / fr)	Bones	Pre time (sec)	Vtx pos err (err sum / fr)	Vtx nor err (err sum / fr)
SQP	16	42.44	8.234	72.285	16	19.22	5.250	44.169	16	46.67	7.558	58.346
	9	307.72	81.752	181.961	9	79.25	93.688	197.804	9	353.09	61.736	171.901
	3	12.59	173.017	302.695	3	2.50	171.941	294.206	3	4.53	179.191	326.399
SQP with analysis	16	41.81	4.391	23.026	16	15.70	4.016	30.205	16	7.89	5.139	23.987
	9	18.53	42.990	99.952	9	59.48	40.279	111.823	9	304.19	26.152	73.442
	3	18.80	110.395	228.633	3	2.70	71.165	165.905	3	5.17	80.279	168.921
VPNO	16	2435.33	2.354	11.771	16	369.86	1.957	8.861	16	1211.06	1.470	11.341
	9	659.97	9.622	53.529	9	113.86	16.812	73.179	9	336.66	16.693	72.187
	3	15.22	96.496	223.465	3	2.80	55.605	141.661	3	5.70	63.847	137.355

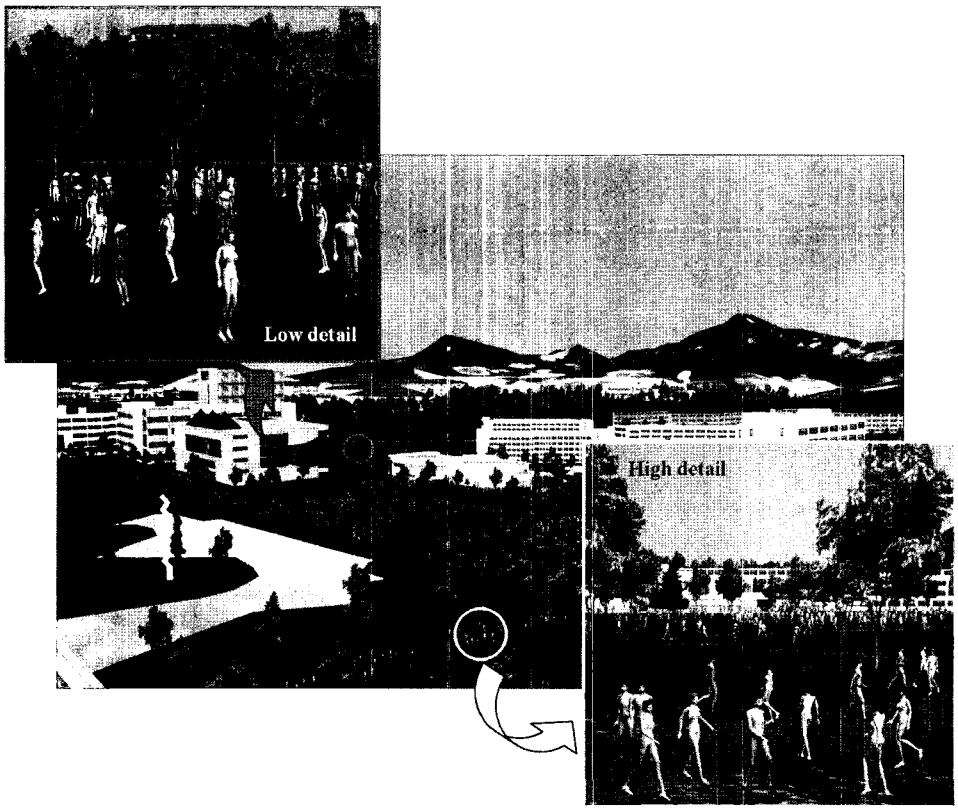


그림 9 총 3000개의 가상캐릭터로 구성된 군중 애니메이션 장면(좌측상단: 멀리 보이는 군중들이 낮은 상세도로 애니메이션 되는 장면; 우측하단: 가까이 보이는 군중들이 높은 상세도로 애니메이션 되는 장면)

군중 애니메이션 시스템을 소개하고 전체 시스템의 핵심 모듈인 동작 전처리 방법에 대해 상세히 기술하였다. 전처리 과정에서 동작 분석은 단순화 된 동작의 사실성을 극대화 하였다. 동작 단순화 과정에서는 애니메이션 속도향상에 중요한 역할을 하는 관절을 제거하고 관절의 계층구조를 단순화 하였다. 또한, 단순화 된 관절구조에 비선형 최적화 방법을 이용해 기존동작과 유사한 단순화 동작을 생성하였다. 이러한 동작 전처리 방법은

현재 널리 사용되는 임포스터 방법에 비해 전체적인 애니메이션 속도향상은 떨어지지만, 동작의 사실성을 증가시키고, 전처리 결과물로 필요한 메모리를 절약할 수 있을 뿐만 아니라 사용자와 상호작용이 가능하여 런타임에 군중의 동작을 제어할 수 있다.

참 고 문 헌

[1] A. Aubel, R. Boulic and D. Thalmann, "Real-time display of virtual humans: Level of details and impostors," IEEE Transactions on Circuits and Systems for Video Technology, Special Issue on 3D Video Technology, 10(2): 207 - 217, 2000.

[2] F. Tecchia, C. Loscos and Y. Chrysanthou, "Image Based Crowd Rendering," IEEE CG&A March/April, pp.36-43, 2002.

[3] S. Dobbyn, J. Hamill, K. O'Conor and C. O'Sullivan, "Geopostors: A real-time geometry / impostor crowd rendering system," I3D 2005.

[4] Z. Popovic and A. Witkin, "Physically based motion transformation," ACM SIGGRAPH 99, pp.11-20, 1999.

[5] S. Redon, N. Galoppo and M. Lin, "Adaptive Dynamics of Articulated Bodies," ACM SIGGRAPH 2005.

[6] J. Beaudoin and J. Keyser, "Simulation Levels of Detail for Plant Motions," SCA 2004.

[7] J. Ahn and K. Wohn, "Motion Level-of-Detail: A Simplification Method on Crowd Scene," CASA 2004.

[8] J. Ahn, S. Oh and K. Wohn, "Optimized Motion Simplification for Real-time Crowd Animation," Journal of Computer Animation and Virtual Worlds 17(3-4): 155-165, 2006.

[9] J. Lewis, M. Cordner and N. Fong, "Pose Space Deformations: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation," ACM SIGGRAPH 2000.

[10] D. James and C. Twigg, "Skinning Mesh Animations," ACM SIGGRAPH 2005, pp.399-407, 2005.

[11] H. Hoppe, "Progressive Mesh," ACM SIGGRAPH 96, pp.99-108, 1996.

[12] M. Garland and P. Heckbert, "Mesh Simplification with Quadric Error Metrics," ACM SIGGRAPH 97, pp.209-216, 1997.

[13] M. Gleicher, "Retargetting Motion to New Characters," ACM SIGGRAPH 98, pp.33-42, 1998.

[14] K. Choi and H. Ko, "Online Motion Retargetting," Journal of Visualization and Computer Animation 11(5): pp.223-235, 2000.

안 정 현

1998년 한국과학기술원 전산학과(학사) 2000년 한국과학기술원 전산학과(석사) 2006년 현재 한국과학기술원 전산학과 박사과정. 관심분야는 군중애니메이션, 모션캡처, 가상현실

원 광 연

1974년 서울대학교(학사). 1984년 Univ. of Maryland 전산학(박사). 1986년 Univ. of Pennsylvania 교수. 1991년 한국과학기술원 전산학과 교수. 2005년 한국과학기술원 문화기술대학원 원장. 관심분야는 문화기술, 가상현실, 컴퓨터 그래픽스, 비젼