수화 애니메이션을 위한 중간 프레임 생성 방법

오 정 근† · 김 상 철††

요 약

동영상 처리와 컴퓨터그래픽 기술이 발전됨으로써, 수화 단어들의 캡쳐된 동영상들을 이용해서 임의의 문장에 대한 수화 동작을 보여주는 교육시스템이 최근에 출현하고 있다. 본 논문에서는, 한 수화 단어의 마지막 프레임과 다음 수화 단어의 시작 프레임 사이에서 손 모습의 변화를 애니메이션하는 중간 프레임들을 생성하는 효율적인 방법을 제안한다. 먼저, 중간 프레임의 생성에 필요한 여러 위치와 각도를 갖는 손 모습을 미리 파악하고 이들을 실제로 캡쳐해서 데이터베이스에 저장한다. 주어진 두 수화 프레임에 대해서, 그들 사이에 존재해야 하는 위치와 각도를 갖는 손 모습들을 순서대로 결정함으로써 중간 프레임들을 생성한다.

본 논문에서 제안한 방법은 계산상 단순하고, 작은 디스크 용량을 필요로 한다. 실험에 의하면, 짧은 시간 내에 15 fps (frame per second) 속도로 재현할 수 있는 중간 프레임들을 생성하였고, 실제 재현 시에 부드러운 애니메이션 효과를 보여주었다. 두 영상사이의 중간 프레임을 생성하는 기존 기법은 상당한 계산량이 요구되거나 의도하지 않는 영상이 생성되는 단점이 있다.

A Method for Generating Inbetween Frames in Sign Language Animation

Jeong-Keun Oh† · Sang-Chul Kim††

ABSTRACT

The advanced techniques for video processing and computer graphics enables a sign language education system to appear. The system is capable of showing a sign language motion for an arbitrary sentence using the captured video clips of sign language words. In this paper, a method is suggested which generates the frames between the last frame of a word and the first frame of its following word in order to animate hand motion. In our method, we find hand locations and angles which are required for inbetween frame generation, capture and store the hand images at those locations and angles. The inbetween frames generation is simply a task of finding a sequence of hand angles and locations.

Our method is computationally simple and requires a relatively small amount of disk space. However, our experiments show that inbetween frames for the presentation at about 15 fps(frame per second) are achieved so that the smooth animation of hand motion is possible. Our method improves on previous works in which computation cost is relatively high or unnecessary images are generated.

1. 서 론

최근 멀티미디어 기술과 CAI(Computer Aided Instruction) 이론을 바탕으로 하여 교육적 효과를 높일 수 있는 시스템들이 많이 등장하고 있다. 이런 시스템들 중에는 농아(deafness and dumbness)들의 의사전달 방식인 수화(sign language)를 교육하는 시스템이 국내 신문이나 방송에 수차 보도된 바 있다. 이것은

† 정 회 원 : 종로산업학교 정보응용과 교수
†† 정 회 원 : 한국외국어대학교 컴퓨터공학과 교수
논문접수 : 1999년 10월 13일, 심사완료 : 2000년 2월 15일

농아 교육 환경을 개선하고자 하는 절실한 요구에 따라서 수행된 연구 개발의 결과라고 판단된다. 초기의 수화교육시스템은 사용자(즉, 학습자)에게 간단한 지화나 지문자를 정지화상 형태로 보여주는 방식이고, 그 후 컴퓨터 하드웨어의 속도 증가와 멀티미디어 시스템 기술의 발달로 점차 동화상을 이용한 시스템[1, 2]이 개발되었다. 동화상을 이용하는 수화교육시스템의 주요 기능은 사용자가 키보드로 입력한 문장(예를 들면, 나는 학생이다)을 수화 동작 동화상으로 보여주는 것이다.

입력된 문장에서 수화 동작 동화상을 보여주려면 크게 두 가지 기술에 대한 연구가 선행되어야 한다. 두 가지 기술이란 입력된 문장을 단어(lexicon)의 열(sequence)로 분석하고 문장의 의미를 분석하는 자연어 처리 기술과 그 열로부터 수화 동작 동화상을 생성하는 기술을 말한다. 한글 자연어 처리는 상당한 기간에 걸쳐서 연구가 진행되어 왔고 현재는 상용화 제품이 출현하는 단계에 이르렀다. 만약 문법적으로 간단한 문장들만을 처리한다면, 수화 동작의 동화상 처리에 대한 연구가 더 중요하다고 할 수 있다.

수화 동작 동화상을 만드는 방법은 크게 다음 3가지로 나누어 볼 수 있다. 첫 번째는 각 문장별로 동화상 파일을 저장하는 것인데, 이것은 저장된 내용만 학습할 수 있다는 것과 방대한 동화상 파일의 크기로 인해서 제한된 수의 동화상만을 저장하는 문제점이 있다. 두 번째는 수화 동작을 3D 모델 기반의 애니메이션으로 처리하는 것인데, 지금까지의 컴퓨터 애니메이션 연구는 얼굴[3, 4]이나 전신 애니메이션[5-9]에 초점이 맞추어져 있는 반면에 손동작에 대한 연구 발표는 많지 않다. 손과 팔의 관절 및 근육의 형태를 적절히 반영한 3D 모델(여기에는 기하학적인 구조와 변형 방법이 포함)을 고안하고 자연스러운 수준의 손동작을 렌더링하기까지는 더 많은 연구가 진행되어야 된다고 생각된다[10, 11].

세 번째는 단어별 동화상 파일을 데이터베이스에 저장하고, 주어진 문장을 구성하는 단어들의 동화상 파일을 차례로 화면에 보여주는 것이다. 표준수화[12]에 정의된 기본 단어의 개수가 2,500개 미만인 점을 감안하면, 첫 번째 방법에 비해서 훨씬 작은 저장공간만으로 임의의 문장에 대한 수화 동작을 만들어 낼 수 있다는 장점이 있다. 우리가 실험해 본 바에 의하면, 이 방법은 사용자의 불편이 없을 정도의 속도 지연으로 동화상 생성을 가능하게 했다. 이상의 이유에서 동화상을 이용한 수화교육시스템의 개발에는 세 번째 방89법이 가장 적합하다.

단어들의 수화 동화상 파일들을 연결해서 한 문장의 수화 동화상을 구성할 때, 손동작의 연결이 끊어지는 현상이 발생한다. 예를 들면, "나는 사람이다"는 문장의 수화 동화상은 '나'의 수화 동화상과 '사람'의 수화 동화상을 연결해서 만들 수 있다. 참고로, 수화에서는 조사인 '는'과 '이다'는 표현하지 않아도 된다[12]. 참고로, 수화 동작 '나'의 끝 부분에서 양손은 뺨 주위에 있는 반면에 수화 동작 '사람'의 시작 부분에서 양손은 가슴 아래에 있다. 그러므로, '나'와 '사람'의 동화상을 그대로 연결하면, '나'의 끝 부분과 '사람'의 시작 부분 사이에서 손의 위치가 갑자기 바뀌게 되어서 손 움직임이 끊어지는 현상이 발생한다.

두 동화상에서 첫 번째의 마지막 프레임(frame)과 두 번째의 첫 프레임을 자연스럽게 연결하기 위해서는, 소위 중간 프레임(inbetween frame)들을 생성해야 한다. 중간 프레임 생성을 위한 일반적인 방법인 디졸빙(dissolving)[2], 워핑(warping)[13, 14], 몰핑(morphing)[15]을 수화 동화상 중간 프레임 생성에 적용하면 여러 가지 문제점이 있다. 예를 들면, 두 화상이 겹쳐지면서 두 개의 왼손 또는 오른손이 보이거나, 프레임 내용을 단순히 변형시키는 것만으로는 자연스러운 손 움직임을 나타낼 수 없거나, 이미지 변형에 상당한 계산량이 요구되기 때문에 실시간으로 처리하기 어렵다[2].

본 논문은 한 수화 단어의 끝 프레임과 다른 수화 단어의 시작 프레임간의 중간 프레임을 생성하는 효과적인 방법을 제안한다. 우리는 두 단어 사이에서 수화자의 손이 실제 이동될 때의 손과 팔의 모습을 파악하고, 이들 모습을 캡쳐(capture)해서 데이터베이스에 저장하고 이들을 이용해서 중간 프레임을 생성한다. 표준수화에 나타나는 임의의 두 수화 단어를 어느 정도 매끄럽게 연결하기 위해서는 1,000여 개의 손 모습만 있으면 되었다. 본 논문에서 제안한 방법을 이용해서 수화교육시스템 을 구축한 결과, 문장의 입력부터 화면에 수화 동작이 보일 때까지의 시간 지연을 사용자는 느끼지 못했다.

본 논문의 구성은 다음과 같다. 제2장에서는 중간 프레임 생성에 관한 기존의 연구 내용을 살펴보고, 제3장에서는 수화 애니메이션에 필요한 수화 동작에서의 손의 형태, 위치 등을 선별하는 방법을 논하며 제4장에서는 중간 프레임을 생성하기 위한 알고리즘을 다루

기로 한다. 제5장에서는 실험 결과를 보여주고 마지막 장에서는 결론을 서술한다.

2. 관련연구

중간 프레임 생성은 최근 활발히 연구되고 있는 동화상 코딩(coding)의 한 분야로 볼 수 있다. 낮은 비율의 동화상 코딩(low bit rate video coding)을 위해서 프레임들을 서브샘플링(subsampling)한 후에 전송하면, 수신자 측에서는 중간 프레임을 자동으로 생성함으로서 이미지의 주관적인 화질을 높일 수 있다. 동화상 코딩에서 중간 프레임 생성을 위해서는 블록기반(block-based) 방법, 화소 재귀적(bel-recursive) 방법, 영상 워핑(image warping) 방법 등이 제안되어 있다.

블록 기반 방법은 가장 간단한 것으로써, 영상을 여러 블록으로 나누고 각 블록의 움직임 벡터를 추정한 후에, 그 벡터를 선형 인터폴레이션(linear interpolation)하는 것이다[16-19]. 블록들은 전체 영상을 사각형으로 나눈 것인데, 만약 시각적으로 중요한 영상 특징(features)들이 여러 블록에 걸쳐져 있고 이들 블록의 움직임 벡터가 다르면 생성된 중간 프레임에는 이상한 현상(artifact)이 생기게 된다. 이런 단점 때문에 화소 재귀적인 방법이 제안되었다. 이 방법에서는 각 픽셀들에게 고유한 움직임 벡터를 부여하고 이들 벡터를 선형 인터폴레이션하는 것으로써, 계산량이 많으며 움직임 벡터 추정이 객체의 윤곽선에서는 실패할 가능성이 많은 단점이 있다[20].

위의 두 방법을 수화 프레임들간의 중간 프레임을 생성하는 데 적용하는 것은 한계가 있다. 이들 방법은 선형적인 객체들의 움직임을 가정하지만 손과 팔의 실제 움직임은 그렇지 않는 경우가 많고, 또한 손과 팔의 움직임이 2차원적이지 않은 경우(예를 들면, 한 프레임은 손등을 보이고 다른 한 프레임은 손바닥을 보임)에는 적용이 불가능하다.

영상 워핑은 먼저 두 영상들에 특징 선들을 명시하면, 이들 특징 선을 이용해서 각 픽셀의 움직임을 표현하는 공간적 변형(spatial transformation)을 구하는 것이다[13]. 한 프레임에서 다른 프레임의 모양(shape)으로 변형시킴으로써 구해지는 중간 프레임들과 반대 방향으로 변형시킴으로써 구해지는 중간 프레임들을 평균함으로써 최종 중간 프레임들을 생성하게 된다. 영상 워핑은 블록 기반 방법을 개선하기 위해서 제안되어 왔다고 볼 수 있다[16]. 이 방법은 이미지 변형에 상당한 계산 량이 필요하기 때문에 현재의 데스크탑 컴퓨터에서는 특별한 하드웨어 지원이 없이 실시간으로 처리하기 힘들다. 또한, 2차원적이지 않은 손과 팔의 움직임은 2차원적 공간적 변형만으로는 표현할 수가 없다.

영화의 특수효과에 자주 사용되는 방법인 디졸빙(dissolving)과 몰핑(morphing)도 중간 프레임 생성에 활용될 수 있다. 디졸빙은 한 영상의 끝 부분과 다른 영상의 처음 부분을 중첩(overlap)하는 기법인데, 색의 강도(Intensity)를 변화시키는 웨이팅 값을 두 영상간에 다르게 부여함으로써 움직임 효과를 얻을 수 있다[2]. 디졸빙은 객체들의 실제 움직임을 포착하지 않고 다만 이미지 중첩에 의한 시각적인 효과를 얻는 것으로 실제감이 떨어지며, 이를 이용해서 생성된 중간 프레임에서는 이미지 중첩 현상이 분명히 느껴지는 단점이 있다. 몰핑은 워핑과 디졸빙을 동시에 수행하여 중간 단계의 이미지를 생성하는 기법으로써, 워핑과 디졸빙이 가지고 있는 단점을 그대로 가지고 있다. 최근에 3차원 몰핑에 대한 연구가 발표되고 있지만 사실감과 처리시간상의 문제점 때문에 동화상의 중간 프레임 생성에 사용하기에는 적당하지 않다고 판단된다.

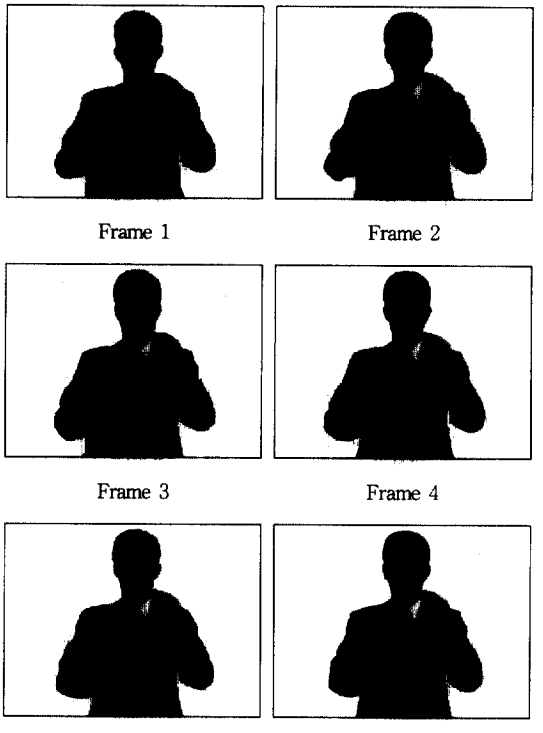
3. 문제의 정의 및 제안된 방법의 전반적인 기술

본 논문에서 해결하고자 하는 문제를 정의하면 다음과 같다. 한 수화 단어를 표현하는 수화 동작의 끝 프레임과 다른 수화 단어를 표현하는 수화 동작의 시작 프레임이 주어졌을 때, 이들 사이를 연결하는 중간 프레임들을 생성하고자 한다.

앞으로, 중간 프레임 생성에서 한 수화 동작의 끝 프레임을 **원천 프레임**, 다른 수화 동작의 시작 프레임을 **목적 프레임**이라고 한다.

본 논문에서 제안하는 방법의 핵심은, 임의의 원천 프레임에서 목적 프레임으로의 변화를 애니메이션 하는 데 필요한 손 모습들을 모두 파악하여 데이터베이스하고, 이들을 그대로 이용하는 것이다. 특정한 손 모습은 손의 위치, 손의 형태, 손목 관절의 각도 등으로 명시할 수 있다[21]. 원천 프레임과 목적 프레임 사이의 중간 프레임들은 데이터베이스에 저장된 손 모습들을 순서대로 찾아내어서 생성하면 된다. 예를 들어, (그림 1)의 Frame 1과 6과 같은 원천 및 목적 프레임

이 주어졌다고 가정하자. 두 프레임 사이의 중간 프레임으로 Frame 2부터 5까지를 생각할 수 있는데, 우리의 방법은 이들의 손 모습을 데이터베이스로부터 검출하는 것이다.



(그림 1) 중간프레임 생성의 예

수화에서의 팔은 의사전달에 사용되지 않기 때문에, 팔은 손의 동작에 가장 적합한 형태를 취하면 된다[12]. 그러므로, 우리는 중간 프레임 생성을 위한 손 모습들을 고려할 때 팔의 모습과 움직임은 고려하지 않는다.

우리는 우선적으로 표준수화에 나오는 단어들 중에서 초등학교 교과서에 나오는 단어를 중심으로 1,500여개를 발췌하였고, 본 논문은 이들 단어를 사용했을 때의 중간 프레임 생성 방법을 제안한다. 하지만, 우리는 본 연구가 더 많은 수화 단어를 고려했을 때에도 적용될 만큼 일반적이라고 주장한다.

4. 중간 프레임 생성용 손 모습 데이터베이스

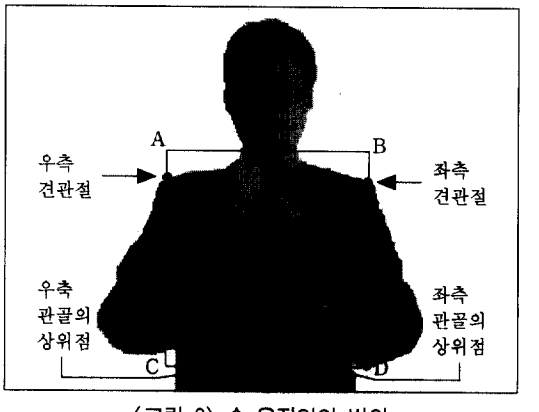
본 장에서는, 중간 프레임 생성용 데이터베이스에 저장되어야 하는 손 모습들과 데이터의 양을 줄이기 위한 효과적인 방법을 서술한다.

4.1 손의 위치, 각도, 형태

임의의 두 수화 단어에 대해서 한 수화 단어의 끝과 다른 수화 단어의 시작을 연결해 주는 손 모습들을 파악하려면, 우리가 고려 중인 모든 수화 단어들의 시작과 끝에 나타나는 손 모습을 분석하여야 할 것이다. 본 장에서는 중간 프레임 생성을 위해서 필요한 손 모습들을 손의 위치, 형태, 각도 면에서 서술하고자 한다.

4.1.1 위치 선별

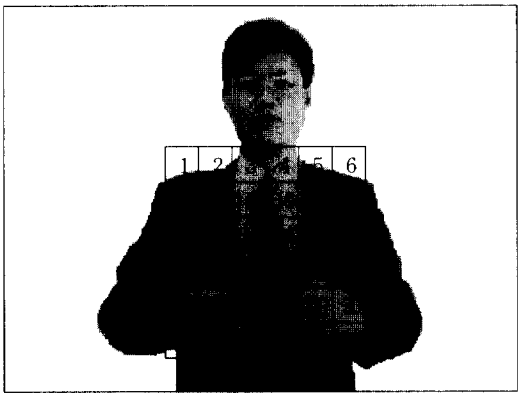
수화자의 손은 몸통의 앞, 뒤, 옆으로 반팔길이 내에서 주로 움직인다[22]. 본 논문에서 고려 중인 초급수화교육용 수화단어들에 있어서, 수화자의 손은 대부분 좌우로는 좌우 견관절(Shoulder Joint) 사이를, 상하로는 턱 아래의 목 부분부터 관골(Hip Bone)의 상위 점 사이를 움직인다[12]. (그림 2)는 수화자의 손이 움직이는 범위를 보여준다. 그 범위는 네 개의 점 A, B, C, D로 정의되는 사각형인데, 수화 동작의 시작 또는 끝에서 수화자의 손은 그 사각형 내에 존재하게 된다.



(그림 2) 손 움직임의 범위

다음으로 데이터베이스에 저장할 손 모습의 구체적인 손 위치를 결정하자. 원천 프레임과 목적 프레임에 나타나는 손이 각각 (그림 2)의 A, B, C, D 사각형 내에 존재한다면, 중간 프레임 생성을 위해서 손이 움직여야 하는 가장 긴 거리는 AD 또는 BC 대각선이다. 우리의 실험에 의하면, 수화자가 그 대각선을 따라서 실제로 손을 움직이면 보통 0.4초 정도 걸렸다. 또한,

우리가 실험을 통해서 확인한 사실은, 실제 수화 동작을 15 fps(frame per second) 정도로만 캡쳐하여 저장하면, 재현 시의 손 움직임은 상당히 매끈하다는 것이다. 그래서, 우리는 15 fps의 프레임 비율로 중간 프레임들을 생성하고자 한다. 앞의 0.4초는 손의 움직임 애니메이션을 위해서는 6프레임(15 fps x 0.4 sec = 6임)이 필요하다는 것을 의미한다. 이를 위해서 우리는 그 대각선을 따라서 등간격으로 6개 지점을 정하였다. 직선 AB와 CD는 손이 수평 및 수직으로 움직일 때의 최대 거리이다. 이 거리는 대각선보다 다소 짧지만 그 차이는 크기 않기 때문에 구현의 편의상 AB와 CD도 등간격으로 6개 지점으로 나누었다. 결론적으로, 중간 프레임 생성용으로 캡쳐할 손의 위치는 손 움직임의 범위를 나타내는 사각형을 가로, 세로로 각각 6등분함으로써 구한다.



(그림 3) 손의 위치

(그림 3)에서 각각의 작은 사각형은 하나의 손의 위치를 나타내는 것으로서, 특정 사각형의 위치에 손을 두려면 손의 중점을 그 사각형의 중간에 두면 된다. 앞으로 그 사각형을 "셀"이라고 부른다. 우리는 손의 중점은 "손목의 중점"으로 정의하는데, 그 이유는 팔의 위치가 결국 손의 위치를 결정하는 것이고, 팔의 위치를 가장 잘 나타내는 것이 손의 구성 요소인 손가락, 손등, 손목 중에서 손목이기 때문이다. 손가락이나 손등은 구부린 정도에 따라서 다른 형태를 가지며, 그것으로 인해서 이들을 2차원 영상으로 캡쳐하였을 시에 이들의 중심점이 달라지기 때문이다. (그림 3)과 같이 총 36개의 셀에 편의상 일련번호를 부여한다.

손 움직임의 범위를 나타내는 사각형을 가로, 세로로 6등분하지 않고, 데이터의 양을 줄이기 위해서 5등분 이하로 하거나 또는 좀더 매끄러운 움직임 애니메이션을 기대하면서 7등분 이상으로 할 수도 있을 것이다. 이들에 따라서 캡쳐한 손 모습들로부터 생성한 중간 프레임들의 애니메이션 효과를 비교하면 <표 1>과 같다. <표 1>에서 '셀 수'는 손의 움직임 범위를 나타내는 사각형을 가로 및 세로로 몇 개의 셀로 나누는 지를 말하고, '중간 프레임 재현 시간'은 초당 15 프레임으로 재현했을 때에 손이 최대 거리인 대각선을 따라서 움직이는데 걸리는 시간을 말한다.

〈표 1〉 손 위치의 개수에 대한 비교

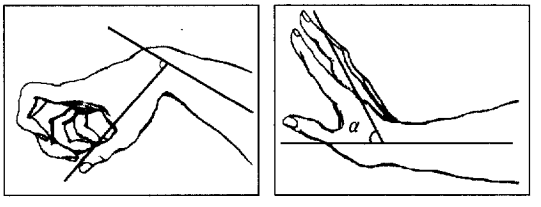
셀 수	중간 프레임 재현 시간	상하 또는 좌우운동에서의 애니메이션 효과	대각선 운동에서의 애니메이션 효과
5 x 5	0.33초	매끄럽지 않게 진행됨	필름이 끊기는 듯한 현상이 발생함
6 x 6	0.4초	매끄럽게 진행됨	매끄럽게 진행됨
7 x 7	0.47초	매끄러우나 중간프레임이 너무 오래 지속됨	매끄러우나 중간프레임이 오래 지속됨

해부학적 관점에서 보면 손목의 위치를 결정하는 요인들은 팔 상단부와 팔 하단부 사이의 각도, 팔 하단부의 회전각도, 팔 상단부의 회전각도, 팔 하단부와 몸통 사이의 각도 등이 있다. 여기서, 팔 상단부는 어깨와 팔꿈치 사이를, 팔 하단부는 팔꿈치와 손목사이를 말한다. 우리가 필요로 하는 것은 손의 위치이므로 본 논문에서는 이러한 요인들의 개별적인 작용을 별도로 고려하지 않는다. 단, 중간 프레임 생성시에 손은 어떤 위치에 있더라도 손과 몸통 사이의 거리는 약 20~25cm를 유지하고, 팔꿈치는 몸통에 근접시킨 상태로 자연스럽게 늘어뜨려져 있다고 가정한다. 이 가정은 우리가 관찰한 수화자의 동작 습관에 따른 것으로써 특정 위치의 손 모습을 결정하는 또 다른 요인도 된다.

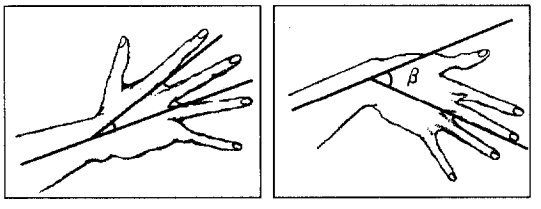
4.1.2 손목 관절의 각도 선별

같은 위치에 있더라도, 손목 관절의 회전 각도에 따라서 손의 모습은 달라진다. 먼저, 손목 관절의 각도를 결정하는 손의 운동을 살펴보기로 하자. 손의 운동에 있어서 피치(pitch)는 손목을 상하로 굽히는 것이고, 요(yaw)는 손 전체를 새끼손가락 쪽이나 엄지손가락 쪽을 향하여 옆으로 움직이는 것이고, 롤(roll)은 팔 하단부에 나란히 있는 두 뼈인 척골(Ulna)과 요골(Radius)

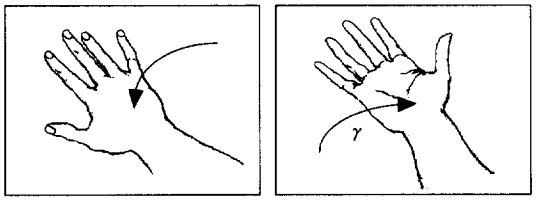
의 운동에 의하여 손을 회전시킴으로써 손등이나 손바닥을 앞으로 향하게 하는 것을 말한다[21]. 이들 움직임을 도식화하면 (그림 4)와 같다.



〈피치 운동〉



〈요 운동〉



〈롤 운동〉

(그림 4) 손의 운동

피치 운동의 각도를 α, 요 운동의 각도를 β, 롤 운동의 각도를 γ라고 할 때, $\alpha = 0°$는 손목을 아래나 위로 전혀 굽히지 않은 경우, $\beta = 0°$는 손 전체를 새끼손가락 방향이나 엄지손가락 방향으로 전혀 굽히지 않은 경우, $\gamma = 0°$는 손등이 완전히 위쪽으로 향한 경우이다. (그림 3)과 같이, 양의 α은 손목을 위로 굽히는 경우, 양의 β는 새끼손가락 방향으로 움직이는 경우, 양의 γ은 손바닥을 위로 향하게 움직이는 경우의 각도를 말한다.

우리가 분석한 바에 의하면, 수화 단어의 끝 동작과 시작 동작에서의 손 모습은 주로 양의 각도를 갖는 피치, 요 및 롤 운동으로 손목 관절이 회전되어 있다. 이들의 운동학적인 범위는 $0° \le \alpha \le 60°$, $0° \le \beta \le 45°$, $0° \le \gamma \le 180°$이다. 우리가 고려 중인 수화단어들의 손동작을 [12]에 따라서 분석해 보면, 이들 각도의 범위는 보다 제한적으로서, $0° \le \alpha \le 30°$, $0° \le \beta \le 30°$, $45° \le \gamma \le 135°$임을 알 수 있다.

우리의 목적은 15 fps 속도로 중간 프레임들을 재현하는 것이기 때문에 중간 프레임의 재현에 걸리는 시간은 최대 0.4초이고, 많은 경우는 이보다 짧은 시간이 걸리게 된다. 대부분의 경우에 사용자는 이만큼의 짧은 시간 동안에 화면에 나타나는 손 모습을 명확하게 파악하지는 못한다. 또한 중간 프레임의 주된 용도는 어떤 의미를 전달하는 것이 아니라 단순하게 손 모습만을 바꾸는 것이기 때문에, 손목 관절의 각도 변화를 미세하게 나누지 않아도 손 모습의 변화를 부드럽게 애니메이션할 수 있다. 피치 각도로는 범위의 양쪽 끝 값을, 요 각도와 롤 각도로는 범위의 양쪽 끝 값과 중간 값을 각각 선택하였다. 즉, 피치 각도로는 0°, 30°를, 요 각도로는 0°, 15°, 30°를, 롤 각도로는 45°, 90°, 135°를 선택하였다. 이와 같은 각도 선택이면 캡쳐할 손 모습의 개수를 최소화하면서 대체적으로 부드러운 수화 애니메이션을 가능하게 했다. 일반적으로 사용자는 컴퓨터 화면에서 피치 각도의 변화보다는 요 각도와 롤 각도의 변화에 더 민감하기 때문에, 요 각도와 롤 각도를 더 미세하게 나눈 것이다.

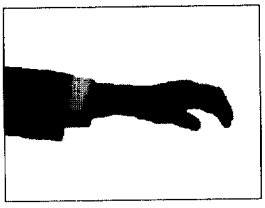
(그림 5)는 피치, 요, 롤 각도 별로 손의 모습을 보여주고 있다. (그림 5)의 마지막 그림은 왼손의 피치, 요, 롤 운동이 복합적으로 이루어진 한 가지의 경우를 보여 준다.

서로 다른 피치, 요, 롤 각도의 조합으로 정의되는 손의 모습은 18개가 있을 수 있다. 그들을 구분하기 위해서 <표 2>와 같은 ID를 부여하는데 다음 장에서 사용된다. ID의 prefix는 L 또는 R인데, L은 왼손, R은 오른손을 나타낸다.

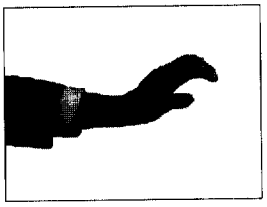
4.1.3 손의 형태 선별

본 논문에서 손의 형태란 손가락을 어떻게 하고 있느냐를 말하는데, 예를 들면, 주먹을 쥐고 있는 지, 특정 손가락을 세우고 있는 지 등을 말한다. 이와 같이 손의 형태는 상당히 다양하기 때문에, 수화자의 모든 가능한 손 형태를 캡쳐해서 데이터베이스에 저장할 수는 없을 것이다.

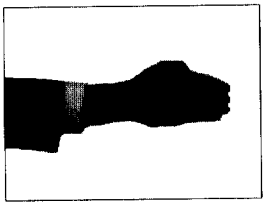
앞에서 언급한 대로, 중간 프레임들은 어떤 의미를 전달하는 것이 아니라 단순하게 손 모습을 바꾸는 용



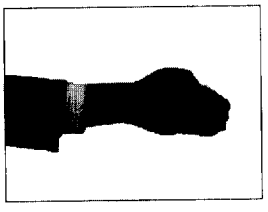
〈피치 0°〉



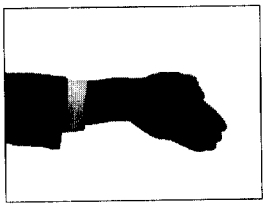
〈피치 30°〉



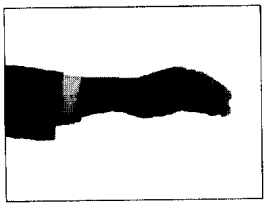
〈요 0°〉



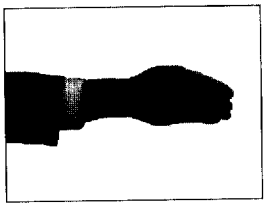
〈요 15°〉



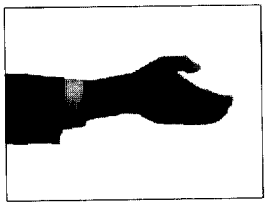
〈요 30°〉



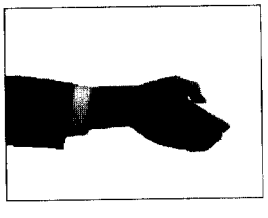
〈롤 45°〉



〈롤 90°〉



〈롤 135°〉



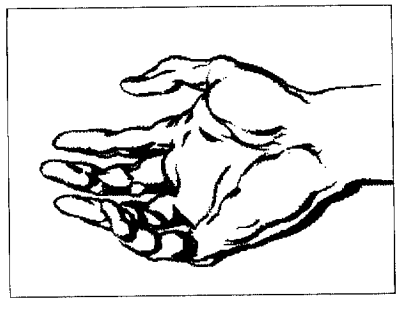
〈피치 30°, 요 30°, 롤 90°〉

(그림 5)

도이다. 정확한 손 형태의 변화를 애니메이션하는 대신에 우리는 (그림 6)과 같이 다섯 손가락 모두를 1/4 정도로 구부린 형태를 선택하였다. 즉, 이러한 형태를 갖는 손 모습만을 캡쳐해서 중간 프레임의 생성에 사용하였다.

〈표 2〉 손의 모습 ID

ID		피치 각도	요 각도	롤 각도
오른손의 경우	왼손의 경우			
R-1	L-1	30	0	45
R-2	L-2	30	0	90
R-3	L-3	30	0	135
R-4	L-4	30	15	45
R-5	L-5	30	15	90
R-6	L-6	30	15	135
R-7	L-7	30	30	45
R-8	L-8	30	30	90
R-9	L-9	30	30	135
R-10	L-10	0	0	45
R-11	L-11	0	0	90
R-12	L-12	0	0	135
R-13	L-13	0	15	45
R-14	L-14	0	15	90
R-15	L-15	0	15	135
R-16	L-16	0	30	45
R-17	L-17	0	30	90
R-18	L-18	0	30	135



(그림 6) 손 형태의 예

4.2 손 모습의 선택

손의 위치는 36가지이고, 손의 각도가 18가지임을 감안하면, 한쪽 손의 경우에 중간 프레임 생성을 위해서 데이터베이스에 저장해야 하는 손 모습은 1296(1296 = 36 x 18)가지이다. 중간프레임 생성에 거의 사용되지 않는 손 모습은 데이터베이스에서 삭제했다. <표 3>은 삭제된 손 모습을 정리한 것이다. 사용되지 않는 손 모습이란 어떤 수화 동작의 시작 프레임과 끝 프레임에 나타나는 손 모습과 거의 일치하지 않으면서 우리의 중간 프레임 생성 알고리즘의 실행 결과에 포함되

지도 않는 손 모습을 말한다. 우리의 중간 프레임 생성 알고리즘은 다음 장에서 자세히 기술한다.

〈표 3〉 삭제한 손 모습

손의 위치	삭제한 손 모습
1	R-3, 6, 9, 12, 15, 16, 17, 18 L-1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18
2	L-16, 17, 18
3	L-8, R-18
4	L-18, R-18
5	R-16, 17, 18
6	R-1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 17, 18 L-3, 6, 9, 12, 15, 16, 17, 18
7	R-3, 6, 9, 12, 15, 16, 17, 18
8	L-16, 17, 18
11	R-16, 17, 18
12	R-3, 6, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18 L-3, 6, 9, 12, 15, 16, 17, 18
13	R-3, 6, 9, 12, 15 16, 17, 18 L-3, 6, 9, 10, 11, 12, 13, 15, 16, 17, 18
14	L-16, 17, 18
17	R-16, 17, 18
18	R-3, 6, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18 L-3, 6, 9, 12, 16, 17, 18
19	R-3, 6, 9, 12, 15, 16, 17, 18 L-3, 6, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18
20	L-16, 17, 18
23	R-16, 17, 18
24	R-10, 11, 13, 14 L-3, 6, 9, 12, 15, 16, 17, 18
25	R-9, 12, 13, 14 L-9, 10, 11, 12, 13, 14, 15, 16, 17, 18
26	R-9, 12, 15, 16, 18 L-9, 12, 15, 16, 17, 18
29	R-12, 15, 16, 17 L-12, 15
30	R-3, 6, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18 L-3, 6, 9, 12, 15, 16, 17, 18
31	R-3, 6, 9, 12, 16, 17, 18 L-1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18

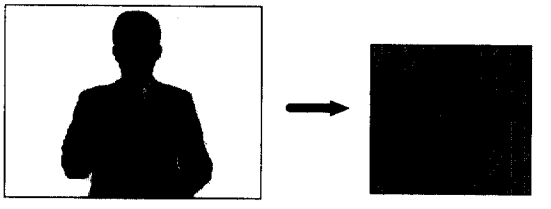
4.3 저장방법

사용되지 않는 것을 제외하여도 왼손, 오른손 각각 1,100개 정도의 손 모습을 데이터베이스에 저장하여야 한다. 손 모습 데이터를 효율적으로 저장하기 위해서 우리는 다음 두 가지를 고려하였다.

첫째, 수화 동작이 손과 팔만을 움직이는 점을 감안해서, 중간 프레임의 생성을 위해서 캡쳐한 수화자의 상반신 영상을 그대로 저장하는 대신에 손과 팔 모습만을 저장하였다. 나중에 손과 팔 모습을 수화자의 상반신 몸체 영상에 합성함으로써 한 프레임을 얻을 수 있다.

둘째, **양쪽 손 모습을 별도로 저장하지 않고 한쪽 손의 모습만을 저장하였다.** 그 이유는 좌우 **대칭을 이용하면** 한쪽 손에서 다른 쪽 손의 모습을 구할 수 있기 때문이다.

(그림 7)은 어떤 수화 프레임으로부터 손과 팔만을 떼어낸 영상을 보여주고, (그림 8)은 합성을 위해서 사용되는 손과 팔이 없는 상반신 영상을 보여준다. 데이터베이스에 저장되는 각 손 모습에 대해서는 손의 위치, 피치, 요, 롤 각도를 파악해서 함께 저장한다.



(그림 7) 떼어낸 손과 팔 영상

(그림 8) 합성용 상반신 영상

5. 중간 프레임 생성 알고리즘

우리의 중간 프레임 생성 알고리즘을 대략적으로 설명하면 다음과 같다. 원천 프레임 S와 목적 프레임 D가 주어지면, 이들 사이에 존재하는 왼손 모습 ID와 오른손 모습 ID의 쌍들을 순서대로 구하고, 각 쌍의 손 모습을 (그림 8)과 같은 수화자의 상반신 영상과 합성함으로써 하나의 중간 프레임을 생성한다.

S와 D 사이에 존재하는 손 모습 ID들을 구하는 알고리즘인 Find_Inbetweens를 C언어 스타일의 pseudo 코드로 기술하면 다음과 같다. 구해진 손 모습과 상반신을 합성하는 과정은 명확하기 때문에 여기에서는 설명을 생략한다.

```
PROCEDURE Find_Inbetweens(프레임 S, 프레임 D)
BEGIN
 STEP 0.
    S에 나타나는 왼손의 위치 P1, 피치 각도 α1, 요 각도 β1, 롤 각도 γ1을 구하고,
    D에 나타나는 왼손의 위치 P2, 피치 각도 α2, 요 각도 β2, 롤 각도 γ2를 구하고,
    S에 나타나는 오른손의 위치 P3, 피치 각도 α3, 요 각도 β3, 롤 각도 γ3을 구하고,
    D에 나타나는 오른손의 위치 P4, 피치 각도 α4, 요 각도 β4, 롤 각도 γ4를 구한다;
 STEP 1.
    P1P2 직선상의 좌표들을 구하고, P1과 P2를 제외한 모든 좌표들을 배열 CL에 저장한다. 이때, CL에 저장된 좌표의 수를 NL이라 하자;
    P3P4 직선상의 좌표들을 구하고, P3과 P4를 제외한 모든 좌표들을 배열 CR에 넣는다. 이때, CR에 저장된 좌표의 수를 NR이라 하자;
    NαL = Count_Flexion_Angles(α1, α2);
    NαR = Count_Flexion_Angles(α3, α4);
    NβL = Count_Adduction_Angles(β1, β2);
    NβR = Count_Adduction_Angles(β3, β4);
    NγL = Count_Supination_Angles(γ1, γ2);
    NγR = Count_Supination_Angles(γ3, γ4);
 STEP 2.
    NL, NR, NαL, NβL, NγL, NαR, NβR, NγR 중의 maximum을 구하고, 그것을 N이라 함. --- (a)
    if(N이 0임) return;
    if(N ≠ NL)
      if((NL > 0) { --- (b)
        CL의 내용을 배열 C에 복사한다; /* 배열 C는 임시 변수임 */
        for(i = 0; i < N; i++) if(NL > 0) CL[i] = C[⌊i*(NL/(float)N)⌋];
      }
      else  CL을 N개의 P1로 채운다;
    if(N ≠ NR)
      if((NR > 0) { --- (c)
        CR의 내용을 배열 C에 복사한다;
        for(i = 0; i < N; i++) if(NR > 0) CR[i] = C[⌊i*(NR/(float)N)⌋];
      }
      else  CR을 N개의 P3로 채운다;
 STEP 3:
   Find_InbetweenLeftHands(α1, β1, γ1, α2, β2, γ2, CL, N, IL);/*왼손의 손모습 ID를 구함 */
   Find_InbetweenRightHands(α3, β3, γ3, α4, β4, γ4, CR, N, IR);/*오른손의 손모습 ID를 구함*/
END.
```

```
PROCEDURE Find_InbetweenLeftHands(정수 αs, 정수 βs, 정수 γs, 정수 αd, 정수 βd, 정수 γd, 좌표 배열 C, 정수 N, 정수 배열 I)
BEGIN
   for(i = 0; i < N; ++i) {
      α = Flexion_Angle(αs + ⌊(αd - αs) * (i / (float)N-1⌋)); -- (d)
      β = Adduction_Angle(βs + ⌊(βd - βs) * (i / (float)N-1⌋)); -- (e)
      γ = Supination_Angle(γs + ⌊(γd - γs) * (i / (float)N-1⌋)); -- (f)
      위치가 C[i], 피치 각도가 α, 요 각도가 β, 롤 각도가 γ인 손모습의 ID를 c라고 하자;
      I[i] = c;
   }
END.
```

중간 프레임 생성에서 우리는 왼손과 오른손을 별도로 고려한다. 원천 프레임의 왼손 모습과 목적 프레임의 왼손 모습에 대해서 중간 손 모습들을 순서대로 구하고, 오른손에 대해서도 같은 작업을 한다. 각 수화 프레임에 나타나는 손의 위치 및 각도는 미리 전처리를 통해서 구해지고 데이터베이스에 저장되어 있다. Step 0에서는 프레임 S와 D에 나타나는 왼손과 오른손의 위치, 피치 각도, 요 각도, 롤 각도를 구하는데, 이 값들은 데이터베이스로부터 얻어 온다.

Step 1에서는, 왼손을 옮길 때에 그 손이 지나가는 지점들의 좌표들을 구해서 배열 C_L에 넣는다. 오른손이 지나가는 지점들의 좌표들은 배열 C_R에 넣는다. 우리가 고려하는 지점들은 (그림 3)의 셀들로서, 셀 31의 좌표는(0, 0) 즉 원점이고, 오른쪽은 X축 양의 방향이며 위쪽은 Y축 양의 방향이다. 예를 들면, 셀 26의 좌표는 (1, 1)이다. 수화자가 직선 운동으로 손을 옮기는 점을 감안하면, 손이 지나가는 셀들은 Bresenham의 알고리즘[23]으로 구할 수 있다. Count_Flexion_Angles는 두 각도 사이에 존재하는 중간 프레임용 피치 각도들의 개수를 세는 함수이다. 예를 들면 20°와 40°에 대해서는 결과가 1인데, 그 이유는 30°만이 이들 두 각도 사이에 존재하기 때문이다. 비슷하게, Count_Adduction_Angles와 Count_Supination_Angles는 두 각도 사이에 존재하는 중간 프레임용 요 각도들과 롤 각도들의 개수를 세는 함수이다.

Step 2의 (a)에서 구한 N은 앞으로 생성할 중간 프레임 수로 사용된다. 만약, N이 0이면, 원천 프레임과 목적 프레임이 손의 위치와 각도 면에서 일치 또는 거의 일치한다는 점을 의미한다. 만약, 배열 C_L의 좌표 수인 N_L이 N 보다 작으면, 그 배열의 좌표들을 중복시키는 방법으로 좌표 수를 늘려서 N과 동일하게 만든다. 이때, N_L이 0이면 N개의 P_1을 C_L에 채워 넣는다. C_L의 좌표 수를 N으로 확장하는 이유는 Step 3에서 나타나듯이, 왼손에 대해서 N개의 중간 손 모습을 생

성하기 때문이다. 배열 C_R에 대해서도 같은 작업을 수행한다. Step 2의 (b)는 $N_L < N$인 경우, (c)는 $N_R < N$인 경우이다. 참고로, ⌊ ⌋는 floor 함수로서, 예를 들면 ⌊A⌋는 A보다 크지 않은 최대 정수를 나타낸다.

Step 3에서는, 함수 Find_InbetweenLeftHands를 통해서 왼손의 중간 손 모습들을 구하는데, 구해진 왼손 모습들의 ID는 배열 I_L에 저장된다. 함수 Find_Inbetween-RightHands는 오른손의 중간 손 모습을 구하는 것이다. 이 두 함수는 거의 비슷하게 동작하기 때문에, 본 논문에서는 Find_InbetweenLeftHands 만을 기술한다.

중간 프레임에 사용될 왼손 각도인 피치, 요, 롤 각도는 원천 프레임과 목적 프레임에 나타나는 왼손 각도의 중간 값들로서, 이들은 선형적 인터폴레이션(Interpolation)을 통해서 구한다. 함수 Find_InbetweenLeftHands의 (d), (e), (f)는 각각 왼손의 피치, 요, 롤 각도를 구하는 것이다. 배열 N_L에 저장된 값, 즉 어떤 왼손 각도와 위치는 특정한 왼손 모습을 지정하게 되고, 그 왼손 모습의 ID를 배열 I_L에 차례로 저장하게 된다. 함수 Flexion_Angle는 인터폴레이션으로 구한 피치 각도로부터 중간 프레임용 피치 각도들 중에 가까운 값을 구하는 함수이다. 예를 들면, 22도에 대해서는 30도가 구해진다. 함수 Adduction_Angle과 Supination_Angle는 요 각도와 롤 각도에 대해서 각각 동일한 작업을 수행한다.

원천 프레임과 목적프레임에 나타나는 손의 각도와 위치는 수작업을 통해서 파악된다. 손의 위치는 손목의 중점이 어떤 셀 내에 있는 지를 따져 봄으로써 알 수 있다. (그림 3)의 셀 1, 셀 6, 셀 31 및 셀 35로 정의되는 사각형 밖의 지점의 좌표는 셀의 크기를 감안해서 결정하면 된다. 예를 들면, 수화자 입술의 좌측 끝 지점은 셀 4의 바로 위에 위치한 셀 내에 있을 것이므로 그 좌표는 (-1, 3)이 될 것이다. 손의 각도는 우선 손을 $\alpha = 0°$, $\beta = 0°$ 및 $\gamma = 0°$가 되도록 한 후에 원하는 손 모습이 나오도록 피치 운동, 요 운동 및 롤 운동을 적절히 수행하면서 손의 각도를 관측을 통해서 파악한다.

6. 실 험

6.1 구현 및 분석

4장에서 제안한 중간 프레임 생성 알고리즘은 실험용 수화교육시스템의 일부로서 구현되었다. 참고로, C 언어를 사용해서 Pentium-233 MHz급 PC에서 구현되었고, 화면의 수화 동영상은 VFW(Video For Window)를 이용해서 재현한다.

각 수화 단어의 동영상은 실제 수화자로부터 해상도 320 × 240, 15 fps로 캡쳐하여 얻었다. 많은 수화 단어들은 손의 위치를 의미 전달에 사용하지 않는다. 이런 경우에 우리는 수화자의 손을 본인이 편한 곳에 두도록 하였는데, 그 곳이 오른손의 경우에는 (그림 3)의 셀 14, 왼손의 경우에는 셀 17이다. 만약 두 손이 가까이 모여서 동작하는 단어의 경우에는 오른손은 셀 15, 왼손은 셀 16에 두기도 했다. 한 손만 사용하는 단어의 경우에는 사용하지 않은 손의 끝이 배꼽 근처에 두도록, 즉 오른손은 셀 32, 왼손은 셀 35에 두도록 했다.

중간 프레임 생성에 사용되는 손 모습의 영상은 해상도 200 × 180, 256 칼라로 캡쳐해서 PCX 포맷으로 데이터베이스에 저장하였다. 총 1,296개의 손 모습 중에 삭제될 것을 제외한 총 데이터의 양은 약 6.9M 바이트이다. PCX 포맷을 사용한 이유는 방법이 간단하고 복원의 속도가 빠르기 때문이다.

주어진 원천 프레임과 목적 프레임으로부터, 4장의 알고리즘을 통해서 중간 프레임 생성용 왼손 모습 ID와 오른손 모습 ID의 쌍들을 구하고, 구해진 손 모습의 영상과 상반신 영상을 합성해서 중간 프레임들을 완성하고, 이를 AVI 포맷으로 변경한 후에, 화면에 재현하는 프로그램을 작성하였다. 그 프로그램을 10가지 경우에 대해서 실험하였는데, <표 4>는 사용된 수화 단어들과 생성된 중간 프레임의 개수를 정리한 것이다. 실험에 사용된 단어의 수화 동작은[12]에 나타난다. 빠른 중간 프레임 생성을 위해서 데이터베이스에 저장되어 있는 손 모습 영상들은 미리 메인 메모리에 읽어 놓는다.

〈표 4〉 10가지 경우에 대한 실험

원천 프레임	목적 프레임	중간 프레임 수
"나"의 끝 프레임	"공부"의 시작 프레임	5
"목"의 끝 프레임	"아프다"의 시작 프레임	4
"물"의 끝 프레임	"흐르다"의 시작 프레임	3
"똑바로"의 끝 프레임	"가다"의 시작 프레임	4
"이름"의 끝 프레임	"부르다"의 시작 프레임	3
"책"의 끝 프레임	"읽다"의 시작 프레임	4
"언덕"의 끝 프레임	"올라가다"의 시작 프레임	4
"나"의 끝 프레임	"부자(富者)"의 시작 프레임	5
"손님"의 끝 프레임	"오다"의 시작 프레임	3
"어려운"의 끝 프레임	"문제"의 시작 프레임	4

실험을 통해서 발견한 바에 의하면, 4장의 알고리즘을 수행하는데 걸리는 시간은 무시해도 좋을 정도이다. 한 쌍의 왼손 모습과 오른손 모습을 상반신 영상과 합성해서 중간 프레임을 생성하고 이들을 AVI 포맷으로 변경하는데 걸리는 시간은 평균 400us 정도였다. 두 수화 단어의 동영상을 차례로 화면에 재현하면서 그 사이에 생성된 중간 프레임들을 재현해 본 결과, 손의 움직임이 상당히 원만함을 확인할 수 있었다. 앞에 예제들에서 나오는 바와 같이, 우리가 사용한 수화 프레임들은 성인 남자의 수화동작을 캡쳐한 것이다. 수화동작은 성별이나 나이에 관계없이 동일하므로, 본 논문에서 제안한 방법은 수화프레임의 주인공인 수화자가 표준수화에 따라서 손동작을 행한다면, 주인공이 누구라도 적용 가능하다.

우리의 방법은 기존의 방법보다 수행속도 면에서 장점이 있다. 프레임의 해상도가 M x N이고, 한쪽 손과 팔만이 나오는 영상의 해상도가 m x n이라 하자. 한 개의 중간프레임을 생성하는 시간을 서로 비교하여 보면, 우리의 방법은 O(r + mn) 시간이 소요되는 반면에, 디졸빙을 이용하는 방법[2]은 O(MN), 모핑을 이용하는 방법[13, 14]은 O(MN) 시간이 소요된다. 여기서, r는 손의 위치와 각도를 계산하는데 필요한 시간으로서 mn보다는 훨씬 적기 때문에 O(r + mn)는 O(mn)로 단순화될 수 있고 mn은 MN보다 적다.

6.2 예 제

"나는 공부한다"라는 문장의 수화 애니메이션은 저장된 "나"의 수화 동화상과 "공부"의 수화 동화상을 차례로 화면에 보여주면 된다. 물론 이들 동영상 사이에는 본 논문에서 제안한 방법을 이용해서 구한 "나"의 마지막 프레임과 "공부"의 시작 프레임 사이의 중간 프레임들도 보여주어야 한다. (그림 8)은 중간 프레임 생성의 예로서, 원천 프레임이 "나"의 마지막 프레임이고 목적 프레임이 "공부"의 시작 프레임이다. (그림 8)에서는 오른손의 이동을 위해서 5개의 중간 프레임이 필요하고, 오른손이 옮겨지는 동안에 왼손의 이동 및 양손의 각도가 변화하고 있다.

7. 결 론

본 논문에서는 두 개의 수화 프레임 사이의 중간 프레임들을 생성하는 방법을 제안했다. 그 방법은 수화 단어들의 동영상을 연속적으로 재현함으로써 임의의 수화 문장에 대한 수화 동작을 보여주는 기능을 제공하는 초급 수화교육시스템을 위해서 고안되었다. 대부분의 경우에, 한 단어의 마지막 프레임과 다음 단어의 시작 프레임에 나타나는 수화자의 손 위치와 각도는 차이가 나기 때문에, 우리의 방법은 손 모습의 변화를 애니메이션하는 중간 프레임들을 생성한다.

(그림 8) 중간 프레임 생성 예

우리의 방법에는, 수화는 손만을 움직이며 팔은 손을 자연스럽게 따라간다는 점을 감안해서, 중간 프레임의 생성에 필요한 여러 위치와 각도를 갖는 손 모습을 미리 파악하고 이들을 실제 수화 동작으로부터 캡

쳐해서 데이터베이스에 저장한다. 주어진 두 개의 수화 프레임에 대해서, 그들 사이에 존재해야 하는 위치와 각도를 갖는 손 모습들을 순서대로 결정함으로써 중간 프레임들을 생성한다.

4장에 나타나듯이 우리의 방법은 계산상 단순하고, 중간 프레임 생성에 사용되는 손 모습 영상의 전체 용량은 약 7 M 바이트 정도이다. 하지만, 실험 결과에 의하면, 짧은 시간 내에 15 fps 속도로 재현할 수 있는 중간 프레임들을 생성하였고, 실제 재현 시에 부드러운 애니메이션 효과를 보여주었다.

두 영상사이의 중간 프레임을 생성하는 기존 기법은 상당한 계산량이 요구되거나 의도하지 않는 영상이 생성되는 단점이 있다. 특히, 많이 사용되는 디졸빙이나 모핑을 이용해서 생성된 수화 중간 프레임들은 손의 자연스러운 움직임을 표현하지 못하고, 두 프레임의 손이 동시에 보이는 점 등으로 인해서 애니메이션의 실제감이 상당히 떨어진다는 단점이 있다.

앞으로 수화자의 입 모양 변화를 나타내는 애니메이션을 추가해서 보다 실감나는 수화 애니메이션을 구현하고자 한다.

참 고 문 헌

[1] 지니어스네트(주), 내부자료, 1997.

[2] 한종호, "수화교육시스템을 위한 동영상 처리에 관한 연구", 한국외국어대학교 교육대학원 석사학위 논문, 1996.

[3] Lee, Y., at. al, "Realistic Modeling for Facial Animation," Proceedings of SIGGRAPH'95, ACM Computer Graphics, pp.55-62, 1995.

[4] Pighin, F., et. al., "Modeling Realistic Facial Expression from Photographs," Proceedings of SIGNGRAPH'98, ACM Computer Graphics, 1998.

[5] Badler, N., Manoochehri, K., "Articulated Figure Positioning by Multiple Constraints," IEEE CG&A, pp.28-38, 1987.

[6] Calvert, T. W., Chapman, J., Patla, A., "Aspects of the Kinematic Simulation of Human Movement," IEEE CG&A, pp.41-49, 1982.

[7] Koga, Y., Kondo, K., Kuffner, J., Latombe, J-C., "Planning Motion with Intentions," Proceedings of SIGGRAPH'94, ACM Computer Graphics, pp.395-408, 1994.

[8] Unurna, M., Anjyo, K., Takeuchi, R., "Fourier Principles for Emotion-based Human Figure Animation," Proceedings of SIGGRAPH'94 Conference, ACM Computer Graphics, pp.91-96, 1994.

[9] Wilhelms, J., "Using Dynamic Analysis for Realistic Animation of Articulated Bodies," IEEE CG, pp.12-27, 1987.

[10] Chang, Y.-K., Rockwood, A. P., "A Generalized de Casteljau Approach to 3D Free-form Deformation," Proceedings of SIGGRAPH'94, ACM Computer Graphics, pp.257-260.

[11] Wilhelms, J., "Anatomically Based Modeling," Proceedings of SIGNGRAPH'97, ACM Computer Graphics, 1987.

[12] 김승국, 표준수화, 오성출판사, 1994.

[13] Nieweglowski, J., Moisala, T., Haavisto, P., "Motion Compensated Video Sequence Interplotion Using Digital Image Warping," IEEE Int'l Conference on ASSP, 5, pp.205-208, 1994.

[14] Witkin, A., Popovic Z., "Motion Warping," Proceedings of SIGGRAPH'95, ACM Computer Graphics, pp.105-108, 1995.

[15] Hodgins, J.K., Pollard, N.S., "Physically Realistic Morphing," Proceedings of SIGGRAPH'97, ACM Computer Graphics, 1997.

[16] Belien, S., Leenders, R., "Animation Techniques," http://www.sara.nl/Rik/REPORT.update/2section 2-1-5.html, 1996.

[17] Jinhui, Y., "Inbetweening for Computer Animation Using Polar Coordinate Linear Interpolation," CS Report Series, CSC 90/R23, Univ. of Glasgow, 1990.

[18] Nielson, G. M, "Smooth Interpolation of Orientations," Models and Techniques in Computer Graphics, pp.75-93, Springer-Verlag, 1993.

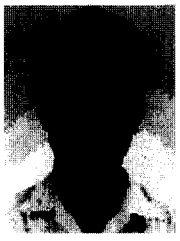
[19] Owen, M., Wilis, P., "Modeling and Interpolating Cartoon Characters," Proceedings of Computer Animation'94, pp.148-155, 1994.

[20] Turbao, S., Rocca, F., "Motion Compensated Image Interpolation," Time-Varying Image Processing and Moving Object Recognition, Elsevier Science Publishers, pp.31-46, 1987.

[21] Hogarth, B., Drawing Dynamic Hands, Watson-Guptill Publications, 1988.

[22] 석동일, 한국수화의 언어학적 분석, p.62, 대구대학교 박사학위 논문, 1990.

[23] Foley, J. D., et. al, Introduction to Computer Graphics, Addison-Wesley, 1993.



오 정 근

e-mail : ojk@chongro-is.ed.seoul.kr

1985년 충남대학교 공업교육대학 기술교육과 졸업(학사)

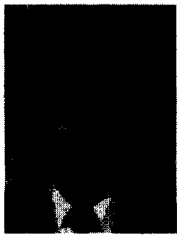
1997년 한국외국어대학교 교육대학원 전자계산교육과 졸업(석사)

1987년~1999년 다수 중등학교의 교사

1999년~현재 종로산업학교 정보응용과 교사

1995년~현재 "PC 단숨에 OK"외 2편 저술

관심분야 : 컴퓨터그래픽스, 인터넷



김 상 철

e-mail : kimsa@maincc.hufs.ac.kr

1983년 서울대학교 컴퓨터공학과 (공학사)

1994년 미시간주립대학교 컴퓨터 공학과(공학박사)

1983년~1994년 한국전자통신연구원, 선임연구원

1994년~현재 한국외국어대학교 컴퓨터공학과 부교수

관심분야 : 멀티미디어통신, 컴퓨터하드웨어, 컴퓨터그래픽스