HMM 인식기 상에서 방향, 속도 및 공간 특징량에 따른 제스처 인식 성능 비교

(A Comparison of Gesture Recognition Performance Based on Feature Spaces of Angle, Velocity and Location in HMM Model)

윤 호 섭 † 양 현 승 ††
(Ho-Sub Yoon) (Hyun-Seung Yang)

요 약 본 논문은 카메라로부터 획득된 영상 시퀀스로부터 얻어진 제스처 궤적 정보를 바탕으로 가장 기본적인 방향, 속도 및 공간 특징을 추출한 후, 각각의 특징 정보들의 인식 결과를 비교하여 어떠한 정보가 가장 유용한지 평가한다. 이를 위해 제스처 궤적 추적을 위해선 컬러 정보 및 모션 정보를 사용하였고, 인식모델로는 시간 데이타 처리에 적합한 HMM을 구성하였다. 실험을 위한 제스처 DB로는 인식하고자 하는 그래픽, 숫자, 알파벳모양의 48개 제스처에 대해 20명으로부터 5개씩 총 4800개의 데이타를 구축하였다.

키워드 : 제스처 인식, HMM, 특징 공간

Abstract The objective of this paper is to evaluate most useful feature vector space using the angle, velocity and location features from gesture trajectory which extracted hand regions from consecutive input images and track them by connecting their positions. For this purpose, the gesture tracking algorithm using color and motion information is developed. The recognition module is a HMM model to adaptive time various data. The proposed algorithm was applied to a database containing 4,800 alphabetical handwriting gestures of 20 persons who was asked to draw his/her handwriting gestures five times for each of the 48 characters.

key words : gesture recognition, HMM, feature space

1. 서 론

컴퓨터 기술의 빠른 발전은 유연한 사용자 접속 요구를 증대시키게 되었다. 사용자 접속기술은 영상과 음성을 처리하는 멀티미디어 기술 및 마우스 등과 같은 사용도구의 발전에 의해 매우 편리하게 되었으나, 궁극적으로는 사람과 사람 사이의 의사소통 수준에 이르러야 한다. 이를 위해 음성과 시각정보에 기반한 사용자 접속 인터페이스에 관한 시스템들이 현재 활발히 개발되어 이미 상용화되고 있으며, 이에 따라 제스처 또한 사용자 접속의 한 파트로서 활발히 연구가 진행되고 있다.

제스처란 사람이 지니고 있는 사고(concept)를 손을 포함한 몸의 움직임으로 나타낸 물리적 표현으로, 사람이 인식하는 제스처는 논리적인 사고가 신체의 움직임이라는 물리적인 운동을 거쳐, 눈에 들어오는 사고의 시각정보로서 정의 된다. 즉, 사람은 대화 중 무의식적 혹은 의식적으로 어떤 동작을 취함으로써 자신의 의사를 상대편에게 보다 잘 전달 되도록 하며, 또한 언어로서 나타나지 않는 자신의 감정까지도 표출하게 된다. 이와 같이 제스처는 사람의 일상생활에서 자신의 의사를 표현하는 데 있어 중요한 보조 수단으로 활용되고 있다.

그러나, 사람이 취하는 제스처는 대부분의 경우 정형화되어 있지 않으며, 어느 정도 정형화 되어있다 하더라도 이의 변화 영역이 매우 넓고 사람마다 혹은 인종마다 다를 수 있다. 또한 동일한 사람이 같은 의미의 제스처를 취하는 경우에도 그 당시의 시공간적(spatio-temporal) 환경에 따라 매우 다르게 해석되기도 한다. 따라서, 제스처 인식 메커니즘을 지능적인 프로그램을 이용해 컴퓨터상에서 구현하기 위해서는 컴퓨터 시각과

† 종신회원 : 한국전자통신연구원 컴소연 공간정보기술센터
yoonhs@etri.re.kr
†† 종신회원 : 한국과학기술원 전자전산학과 교수
hsyang@cs.kaist.ac.kr
논문접수 : 2001년 3월 10일
심사완료 : 2003년 2월 13일

관련된 인공지능 분야는 물론 심리학, 운동역학, 인지과학 등의 여러 분야가 모두 망라되어 연구되어야 할 것이다.

이에 따라 제스처 인식에 관한 연구는 활발한 연구활동에 비해 기술구현의 어려움으로 인하여 매우 제한적인 제약조건에서 추진되고 있다. 즉, 몸 동작 전체가 아닌 주로 손의 움직임을 감지하여 인식하는 연구가 주를 이루고 있으며, 손 제스처를 대상으로 하는 경우에도 제한된 범위의 어휘를 대상으로 하고, 입력정보로 영상 자체 보다 데이타글로브에 의한 전기적인 신호를 인식 대상으로 하는 경우가 많고, 대부분의 경우에 인식 대상이 되는 의미 있는 제스처를 분리하기 위한 스팟팅(spotting)을 고려하지 않는 경우가 많으며, 마지막으로 단순히 손의 움직임을 추적하고 위치를 알아내는 포인팅 수준의 인식을 하는 경우가 많았다. 그러나, 이러한 여러 가지 어려운 장벽을 단시일 내에 극복하기는 매우 어려우며, 당분간은 제한적인 조건하의 제스처 인식이 연구의 주를 이룰 것으로 판단된다.

본 연구에서 다루는 손 제스처 인식의 주요 응용 분야를 분류하면 시각 장애자를 위한 수화인식, VR 시스템에서 가상 공간상의 물체의 취급, 원격지 로봇의 조정, 가전제품의 제어, 컴퓨터에서의 사용자 접속 등의 분야로 대별된다. 최근, VR 시스템에서 데이타 글로브에 의해 가상 공간상의 물체를 제스처에 의해 다루려는 연구가 활발히 진행되어 왔으나, 이 방법에서는 움직임의 정보가 정확하게 실시간으로 입력될 수 있는데 비해 데이타 글로브를 껴야 하고 이에 연결된 전선에 의해 사용자가 부자연스러움을 느끼게 된다. 따라서, 자연스런 사용자 접속도구가 되기 위해서 제스처 인식은 시각정보에 의해 이루어 져야 한다.

시각정보에 의한 제스처 인식에서의 중요하게 다루어져야 할 선행 문제로는 손의 움직임을 어떻게 실시간으로 추적하느냐 이다. 발전된 영상처리 기술에도 불구하고 실시간에 의한 이동물체의 추적은 매우 어려운 문제로 남아있다. 따라서, 손을 쉽게 찾기 위해 구별이 쉬운 장갑을 끼거나, 특별한 색상의 표식을 갖는 장갑을 끼는 경우가 있어왔다. 그러나 본 연구에서는 어떠한 외적인 기구를 사용하지 않고 시각 정보만을 기반 하여 궤적 추적을 수행한다. 또한, 본 연구에서는 제스처 인식을 위한 특징 정보로 단순한 체인 코드 특징에서 탈피하여 카티션(cartesian) 및 극(polar) 좌표상에서의 방향 특징, 위치 특징, 속도 특징 등의 다양한 특징들을 실험하였으며 최적화된 특징 정보에 의해 시스템이 구동 되도록 하였다.

인식 대상인 동적 제스처의 어휘는 그래픽 기본 도형 요소 여섯 가지와 (circle, triangle, rectangle, arc, horizontal-line, vertical-line), 편집 명령 제스처 여섯 가지 (move, copy, undo, swap, remove, close), 숫자 제스처 열 가지 (0, 1, 2, … , 9), 그리고 영문 알파벳 스몰 여섯 가지 (a, b, c, d, …, z)의 총 마흔 여덟 개의 제스처 어휘를 가지고 실험을 하였다. 그림 1에서 인식 대상 제스처의 형태를 보여준다. 각 제스처의 특징들은 시공간 (spatio-temporal) 상에서 mesh 및 k-means 알고리즘에 의해 정량화 되어 심볼 데이타로 표현되도록 하였으며, 정량화 된 특징 정보는 Left-Right HMM(Hidden Markov Models)에서 학습하고 인식하였다.

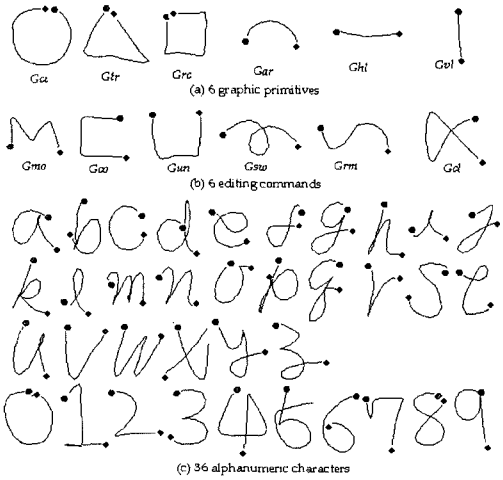


그림 1 인식 대상 제스처의 형태

제스처 연구에 관한 국내의 관련 연구를 살펴보면 국내의 제스처 관련 연구동향으로는 먼저 데이타글러브를 이용한 인식방법으로 ETRI의 김종성 등[1]이 퍼지 이론을 이용한 특징 해석법 및 손가락의 정적인 모양을 분류하기 위한 퍼지 최대-최소 신경망을 사용해 14가지 손 모양을 분류하는 한글 수화용 실시간 인식 시스템을 개발하였다. 경북대학교 김주홍 등[2]은 제스처 인식 및 음성인식을 이용한 윈도우 시스템 제어에 관한 연구를 수행하였는데 호출 명령 5개, 키보드 제어명령 5개의 총 10개의 제스처를 인식하였다. 포항공대 윤명환 등[3]은 3-D 글러브를 이용한 손 동작 분석 시스템을 개발하여 Kinematic Hand Model과 3-D Hand Biomedical Model을 개발하였다.

제스처 궤적을 데이타글러브가 아닌 영상처리 기법을 이용해 추출한 방법으로는 숭실대 양선옥 등[4]이 컬러 정보를 이용한 손 영역 추출 방법에 대해 연구하였고, 인하대학교 이종실[5] 등은 수화 동영상 처리를 위해 인접한 프레임간의 차 영상에 대해 엔트로피를 측정하여 배경 영상과 손 영역을 분리한 후, 손 영역 추적을 행하였다. 또한 KAIST의 고병기 등[6]의 손가락 움직임 제스처 인식에서 FMS(finger mouse system)과 FGRS (finger gesture recognition system)에 관해 연구하였다. FMS는 손 모양을 입력받아 그 움직임과 모양을 인식하여 마우스가 갖는 지시 및 선택 기능을 모의하며, FGRS는 FMS를 통해서 입력된 점들의 순열로 구성된 제스처를 인식하여 명령의 입력과 직접 조작을 수행한다. 실험에 사용된 제스처 어휘는 총 14개이다. ETRI의 이찬수 등[7]도 10가지의 손 모양과 7가지 손 운동을 인식하여 총 16개의 아바타 동작 제어 명령을 인식하는 시스템을 구현하였다.

국외 제스처 관련 연구동향으로는 Hienz 등[8]은 독일 수화를 대상으로 팔과 손을 인식하는 연구를 수행하였다. 손 동작의 정보 획득과 분석을 쉽게 하기 위해 특별히 고안된 글로브와 팔꿈치와 어깨 부분에 칼라 마킹을 사용하였다. 수화 범주에서의 모션은 여섯 가지 파라메터 즉, type, orientation, size, manner, repetition, arrangement of the hands로서 특징 지어졌다. 실험대상 제스처는 straight motion, straight motion with direction change of 90 degree, straight motion with 135 degree, straight motion with direction of 45 degree, circle, curve, spiral의 7개를 대상으로 실험하였으며 95% 수준의 인식율을 갖는 것으로 보고되었다.

Carnegie-Mellon 대학의 Yang 등[9]은 HMM을 사용하여 사람의 행위를 학습 및 인식하는 연구를 수행하였다. 제스처로는 마우스를 이용하여 디지타이징된 숫자를 인식하는 실험을 하였으며, 입력정보는 FFT(fast Fourier transform)의 상수의 진폭에 따라 16차원의 벡터로 양자화하고 이를 left-right HMM 상에서 인식하였다. 실험의 결과는 로봇의 원거리 동작(teleoperation)에 활용하였다. Tsukuba Research Center의 Nishimura 등[10]은 8가지의 매우 큰 범위의 제스처 즉, banzai (hurrah!), come-on, bye-bye, no, circle, to-left, to-right, clap을 실험 대상으로 하였으며 15 Hz의 샘플링에서 80% 수준의 인식율을 얻었다. MIT 대학의 Campbell 등[11]은 스테레오 카메라로부터 실시간으로 입력되는 3차원 정보를 활용하여 제스처를 인식하는 연구를 수행하였다. 실험대상 제스처는 18개의 태극권 기본 제스처를 인식하는 데 활용하였다. 인식 엔진은 연속 HMM을 활용하였으며, 극좌표에서의 각 기본요소의 속도에 의해 shift나 rotation에 민감하지 않은 인식 결과를 보였다. 최근 연구로는 일본 규슈 대학의 Tobery 등[12]이 SOM(self-organization map)을 이용하여 6개의 동적 제스처를 정상적인 비디오 입력 하에서 100% 정확성을 가지고 인식하였으며, 미국 MS 및 인텔사의 공동 연구[13]로는 스테레오 카메라를 이용하여 사용자 팔의 3D 좌표를 추출하고 통계적 모멘텀을 이용해 6개의 제스처 set에 대한 3D 포즈를 인식하여 96%의 인식 결과를 얻었다.

위의 연구 현황에서 제스처 인식 전반에 관해 요약해 보면 과거에는 주로 손의 형태 및 모양을 인식(pose recognition)하거나, 손의 궤적을 추적하는 연구가 수행되었으며, 근자에 이르러 본격적으로 동적 제스처 인식을 시도되었는데 대부분 인식 대상 제스처의 수가 매우 제한적이라는 것을 알 수 있다.

본 논문의 구성은 1장 서론에 이어 2장에서는 제스처의 궤적을 얻기 위한 제스처 추적에 대해 살펴보고, 3장에서는 제스처의 인식 모델 구축에 대해 설명한다. 4장에서는 본 논문의 주안점인 궤적의 방향, 속도 및 공간 특징량에 따른 인식율의 실험 결과를 구현하고, 마지막으로 5장에서 실험 결과 분석 및 결론을 맺었다.

2. 제스처 추적

2.1 제스처 추적

본 절에서 설명될 손 영역 추출 및 추적 알고리즘은 그림 2와 같은 구조를 갖는다. 구현된 알고리즘은 크게 컬러 정보 추출, 차 영상 정보 추출, 그리고 두 정보를 이용한 손 영역 추출 모듈로 구성된다. 그리고 연속적으로 입력되는 제스처 영상에 대해 이들 3 가지 모듈을 반복적으로 수행함으로써 손 영역의 추적 기능을 제공한다.

그림 2의 컬러 정보 추출 모듈에서는 피부 색상의 특성을 이용하여 입력된 영상으로부터 피부 색상을 갖는 영역을 분리한다. 조명 상태 및 개인에 따른 피부 색상의 변화를 해결하기 위해 먼저 사용자의 손 영역을 5×5 영역 셀(cell)을 통해 입력받아 I와 Q의 히스토그램 모델 정보로 등록한다. 등록된 모델 정보는 새로 입력된 영상으로부터 히스토그램 매핑 방법을 통해 유사도를 검사하여 피부색 영역을 결정하는데 사용된다.

움직이는 물체를 추출하기 위해 일반적으로 접근하는 또 다른 방법으로는 연속되는 두 영상간의 시간 차를 이용하는 방법이 있다. 즉, 움직이는 물체의 경우 시간

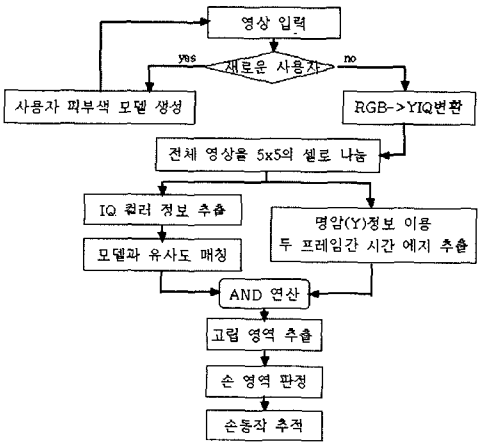


그림 2 손 영역 추적 흐름도

적인 위치 변화가 그 물체를 추출하는데 중요한 정보를 갖고 있으므로 연속되는 두 프레임 간의 차이를 구하면 움직임이 발생된 부분을 쉽게 검출할 수 있다. 이러한 컬러 정보 및 에지 정보 처리 결과는 입력 영상을 1/25로 축소되어 처리되며 향후 축소된 영상에서 처리가 이루어지므로 실시간 처리가 가능하다.

컬러 정보를 이용한 손 영역 추출 방법은 비교적 간단히 손 후보 영역을 추출 할 수 있는 장점이 있으나, 손의 피부색과 유사한 영역(얼굴 및 배경)이 영상 안에 포함될 때 이를 처리하는 복잡한 후 처리가 요구된다. 또한 연속적인 시간 축상의 두 영상간의 차이를 이용하여 손 영역을 추출하는 방법은 손이 이동하는 경우에는 손 영역을 쉽게 찾을 수 있으나, 움직임이 존재하지 않는 경우는 손 영역을 찾을 수 없는 단점이 있다. 이러한 두 방법의 단점은 두 방법을 각자 수행한 후, 간단한 AND 연산을 통해 해결 가능하다. 즉, 손의 피부색 영역과 유사한 영역(얼굴 및 배경)은 움직임이 존재하지 않을 가능성이 높고, 손이 움직이지 않을 경우에도 이전 프레임에서 존재했던 손 영역의 위치와 피부색 영역을 이용하면 손 영역의 추출이 가능하다.

2.2 제스처 시작과 끝 결정(Spotting)

제스처 추적에서 가장 중요한 처리중의 하나는 제스처의 시작과 끝을 결정하는 문제이다. 이 문제는 대개 제스처 인식 시스템의 활용 용도에 따라서 결정된다. 본 연구에서는 사용자가 제스처를 수행할 때 다음과 같은 일반적인 조건이 성립한다고 가정한다.

1. 사용자가 하나의 제스처를 수행하기 위해서는 의도적으로 카메라의 시야 안에서 손을 일정한 시간 동안 움직인다. 이 때 제스처를 끝내기 전에 사용자의 손이 잠시 멈출 수는 있으나 카메라 시야를 벗어나지는 않는다.
2. 두 개의 연이은 제스처 사이에는 약간의 공백 시간이 존재한다. 즉, 사용자가 여러 개의 제스처를 계속해서 수행할 때 한 제스처와 다음 제스처 사이에서 잠시 손을 내려서 쉬는 상태가 존재한다.

위의 가정 1과 2를 이용하면 카메라의 시야(view)에 손이 나타나는 시점을 제스처의 시작으로 간주하고 손이 사라지는 시점을 제스처의 끝으로 간주할 수 있다. 따라서 하나의 제스처는 카메라 시야에 손이 나타나서 사라질 때까지 나타나는 손의 움직임으로 구성된다.

실질적으로 제스처의 시작과 끝의 검출은 손 영역 추출 알고리즘에 의해서 수행된다. 손 영역 추출 알고리즘은 적절한 손 영역을 검출하지 못하는 경우에 손이 존재하지 않는다고 판단한다. 따라서 손이 존재하지 않다가 손 영역이 추출되면 제스처의 시작이 되고, 손 영역이 계속 추출되다가 손이 존재하지 않으면 제스처의 끝이 된다. 한 프레임에서 손이 존재하는가의 판단은 손 영역 추출 알고리즘에 의해 이루어지지만, 손이 나타났는가 사라졌는가를 판단하기 위해선 이러한 손 영역 추출 알고리즘의 판단 결과를 효과적으로 이용하여야 한다. 즉, 손 영역 추출 알고리즘이 완벽할 수는 없기 때문에 이 알고리즘에서 손이 존재하지 않을 때 손 영역을 잘못 추출하거나, 손이 존재할 때 손 영역을 추출하지 못할 가능성을 고려하여야 한다. 이 문제를 해결하는 손쉬운 방법은 손의 존재 여부가 변화할 때, 몇 개의 연속되는 프레임에서 손의 존재 여부를 계속 관찰한 뒤에 손이 나타났는가 또는 사라졌는가를 결정하는 것이다.

손 영역 추출 및 제스처 시작 및 끝을 결정하는 보다 구체적인 내용은 본 논문에서는 자세히 기술하지 않고 이전에 발표한 논문 "시공간적 패적 분석에 의한 제스처 인식"[14]을 참고 문헌으로 제시한다.

3. HMM을 이용한 인식 모델 구축

3.1 Hidden Markov Models

Hidden Markov Model (HMM)[17]은 유한 상태 오토마톤에 확률 개념을 도입한 것이라고 볼 수 있으며, HMM은 상태(state)라 불리는 노드와 이들간의 전이를 나타내는 선분으로 구성된 그래프로 표현될 수 있다. 그래프의 각 노드에는 공간적인 특성을 모델링하는 관측 심볼 확률분포와 초기상태 확률분포가 저장되어 있으며, 각 선분에는 관측열의 시간적인 특성을 모델링하는 상태전이 확률분포가 저장되어 있다.

HMM은 1960년대 후반에 소개된 이래로 최근에 그 사용이 매우 활발하게 되었다. 여기에는 두 가지 이유를 들 수 있는데, 첫째는 이 모델이 수학적인 구조에 기반을 두고 있다는 점과, 둘째는 몇몇 중요한 분야에서 그 실용적인 응용에 성공적이었다는 데 있다. 본 절에서는 HMM에서 다루는 기본적인 개념과 통계적인 모델링 방법을 살펴보고자 한다.

HMM의 이론적인 배경은 1960년대 말과 1970 년대 초에 걸쳐 Baum 등에 의해 발표된 고전적인 연구에 기초하며, 이를 기반으로 카네기 멜론 대학의 Baker 및 IBM의 Jelinek 등에 의해 음성인식 분야에서 응용이 시도되었다. 그러나, 초기엔 기초 이론만이 수학 관련 학자들 사이에서 다루어져 졌고, 이의 응용에 대한 정리가 잘 소개되지 못하여 상당한 시간이 흐른 다음부터 활발한 응용 연구가 시도되게 되었다.

HMM은 시간의 흐름에 따라 연속적으로 나타나는 관측치 인식의 통계적 기법으로 설명할 수 있다. 시간 흐름 상에서의 관측 값들은 이산형이거나 연속형이며 이들은 스칼라나 벡터 값으로 표현될 수 있다. HMM에서 변화하는 통계적 특성 값들을 모델화하기 위해 마코프 체인을 사용하며, 이때의 특징 값들은 각 상태에서의 출력 및 상태 사이의 전이 특징 값으로 나타난다. 따라서 마코프 프로세스는 이중의 통계 프로세스 즉, 1) 상태전이 행렬로서 정의되는 관측이 불가능한 마코프 체인과, 2) 마코프 체인의 각 상태에서의 관측이 가능한 출력 값으로 특징 지워진다. 또한, 출력 값의 특성에 따라 이산형 출력 값으로 나타나는 이산형 HMM과 연속적인 특성 값으로 나타나는 연속형 HMM으로 분류된다. 이러한 이중의 통계적 프로세스는 출력 현상 뿐 아니라, 시간 축 상에서의 거리에 대해 모델링이 가능하다. 위에서 설명된 내용과 같이 각 상태는 두개의 확률집합 즉, 변위 확률집합과 이산형 혹은 분산형 출력 확률집합으로 정의되며 이를 기반으로 아래와 같은 요소로 HMM을 수식화할 수 있다.

1. N: 상태의 수, : $S=\{S_1, S_2, \cdots, S_N\}$: 상태의 집합, q_t : 시간 t 의 상태
2. M: 관측 심볼의 수, $V=\{v_1, v_2, \cdots, v_M\}$: 관측 심볼의 집합
3. $A=\{a_{ij}\}$: 상태 전이 확률 분포, $a_{ij}=P(q_{t+1}=S_j \mid q_t=S_i),\ 1\le i,j\le N$: 상태 i에서 상태 j로 전이할 확률
4. $B=\{b_i(k)\}$: 관측 심볼 확률 분포, $b_j(k)=P(v_k \text{ at } t \mid q_t=S_j),\quad 1\le j\le N, 1\le k\le M$: 상태 j에서 심볼 v_k를 관측할 확률
5. $\pi=\{\pi_i\}$: 초기 상태 확률 분포, $\pi_i=P(q_1=S_i),\ 1\le i\le N$: 초기 상태가 i일 확률

일반적으로 하나의 HMM은 $\lambda=(A,B,\pi)$로 표시된다. 주어진 모델 λ와 관측열 $O=O_1, O_2, \cdots, O_T$에 대해 O의 생성 확률은 아래와 같다.

$$P(O|\lambda)=\sum_{\text{for all } Q}\left[\pi_{q_1}b_{q_1}(O_1)\prod_{t=2}^{T}a_{q_{t-1}q_t}b_{q_t}(O_t)\right]$$

HMM의 응용에는 세 가지의 해결해야 할 문제가 존재하였다. 즉, 평가, 해석, 그리고 학습에 있으며 이들은 각기 Forward-Backward 알고리즘, Viterbi 알고리즘 그리고 Baum-Welch 알고리즘으로 해결되었다.

본 연구에서는 HMM을 이용하여 손 동작을 인식하였다. HMM은 사용자에 의해 생성되는 손 동작 데이타 및 손 영역 추출의 불확실성을 실제 데이타를 이용한 학습에 의해 흡수할 수 있는 모델로, 유한 상태 오토마톤을 비롯한 다른 인식 모델들보다 손 동작 인식에 적합한 것으로 평가되었다. 현재 인식하려는 손 동작은 48가지 부류로 구성된다. 그러므로 각 부류에 대해서 학습 데이타를 생성하고, 이를 이용하여 48개의 모델, $\lambda_1, \lambda_2, \cdots, \lambda_{48}$를 구축한다. 인식은 $P(O|\lambda)$값이 최대인 모델에 해당하는 손 동작으로 관측 열을 분류함으로써 이루어진다.

3.2 실험 환경 구축

동적 손 제스처 인식 시스템은 일반 PC상에서 실 시간으로 구현되었다. 사용된 컴퓨터는 IBM PC Pentium-Ⅲ 450MHz로서 OS로는 Windows98를 이용하였다. 영상 입력을 위한 하드웨어로는 영상 입력 장치로 일반 캠코더나 CCD 카메라를, 영상 캡쳐 보드는 100만원대의 저가형인 Meteor-Ⅱ보드를 사용하였다. 개발 프로그램 언어는 Visual C++ 6.0를 사용하여 구현하였으며, 카메라를 통해 입력된 영상의 해상도는 160×120 pixels 및 24-bit true 컬러이다. 이와 같은 하드웨어 조건 하에서 본 논문에서는 초당 5 프레임 이상의 처리를 수행함으로 실시간으로 제스처 인식이 가능하였다

실험을 위한 제스처 DB로는 인식하고자 하는 그래픽, 숫자, 알파벳모양의 48개의 제스처에 대해 20명으로부터 5개씩 총 4800개의 데이타를 구축하였다. 그림 3은 그래픽 '삼각형', 숫자 '5' , 알파벳 'a'에 대한 각각 100개의 데이타를 보여준다. 획득된 실험 데이타를 이용하여 HMM을 이용한 인식을 수행하기 위해선 먼저 HMM에

서 최적의 상태 수와 적절한 수준의 학습 데이타 양을 정해야 한다. 본 장에서는 먼저 최적의 HMM을 구성하기 위한 최적의 상태 수와 학습 데이타 양 결정방법에 대해 기술한다.

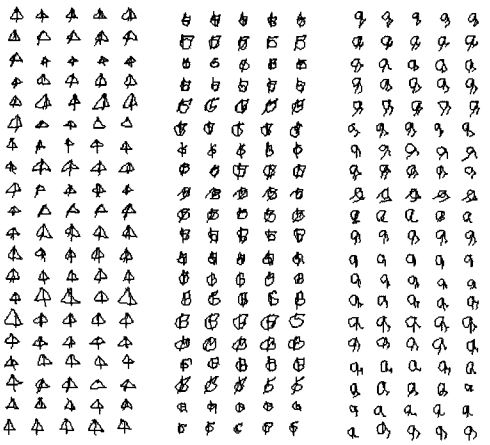


그림 3 '삼각형', '5', 'a'의 제스처 데이타베이스

3.3 HMM 인식기의 환경 설정

3.3.1 최적의HMM 상태 수 결정

HMM을 사용하여 패턴을 인식하고자 할 때, 각 인식하고자 하는 패턴의 상태 수는 인식율에 영향을 미치지만 최적의 상태 수를 결정하는 방법은 대부분 경험에 의존한다. 본 논문에서도 제스처 패턴에 대해 최적의 상태 수를 결정하기 위해 상태 수를 3개에서 12개까지 변경시켜 가면서 공간 특징 정보를 대상으로 인식율의 변동을 살펴보았다. 입력 제스처로는 알파벳 제스처 26개를 샘플로 이용하였으며, 각각 50개의 제스처를 학습에 사용하였고, 학습에 사용하지 않은 나머지 50개 데이타를 사용하여 인식율의 변동을 검사하였다. 그림 4는 상태 수에 변화에 따른 인식율의 변화를 보여준다.

그림 4를 분석해 보면 상태 수가 증가할수록 훈련 데이타와 테스트 데이타의 인식율이 완만하게 증가하다가 10개에 이르러 학습된 데이타의 인식율이 가장 높음을 알 수 있다. 상태 수가 10개를 초과하면 학습된 데이타에서는 인식율이 증가하나 학습되지 않은 데이타에서는 인식율이 감소하였다. 실제 온라인 인식 시스템의 인식율은 학습된 데이타의 인식율보다 학습되지 않은 데이타의 인식율에 의해 좌우되므로, 이러한 결과는 상태 수 10개가 본 인식 시스템에서 가장 적절한 개수임을 보여준다.

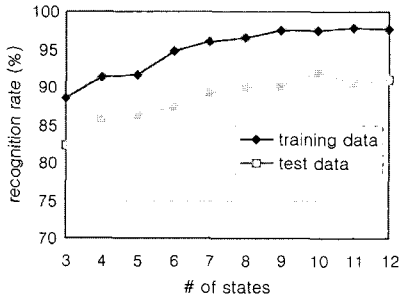


그림 4 HMM상태 수에 따른 인식율의 변화

전체 상태 수가 결정되면 다음 단계는 각각의 제스처 모델마다 HMM의 상태 수를 변경하면서 인식율의 변동을 살펴본다. 이는 제스처의 특성 및 길이에 따라 각 제스처가 갖는 최적의 상태 수가 다르기 때문이다. 보통 전체 제스처에 대한 HMM의 최적 상태 수가 위와 같은 실험에 10개로 결정되었다면 각각의 제스처에 대해 최적의 상태 수를 찾기 위해선 7에서 12사이로 상태 수를 변경하면서 다시 훈련시킨 후, 전체 인식율의 변동을 검사한다. 이 방법을 사용하면 전체 인식율을 약 1~5% 까지 향상시킬 수 있다.

3.3.2 최적의 훈련 데이타 수 결정

일반적으로 신경망과 유전자 이론 등과 같은 학습 알고리즘에서는 인식하고자 하는 패턴에 맞는 훈련 데이타를 획득하는 것과 몇 개의 훈련 데이타를 이용하여 훈련시킬지 결정하는 것이 매우 중요하다. 즉, 훈련 데이타의 양이 모자라면 정확한 인식을 수행하지 못하며, 이와 반대로 훈련 데이타의 양이 지나치게 많으면 훈련 시간이 오래 걸릴 뿐 아니라, 잘못된 패턴을 학습할 확률이 높아진다. HMM을 이용한 제스처 인식기도 적절한 훈련 데이타를 결정하기 위해서 적합한 훈련 데이타 수를 결정해야 한다. 본 논문에서는 각 제스처에 대해 입력받은 100개의 데이타 중 50개는 학습에 사용하고 나머지 50개는 인식 실험에 사용하였다. 따라서 48개 제스처에 대해서는 총 2400개의 데이타가 학습되었고 나머지 2400개를 테스트 데이타로 사용하였다.

4. 방향, 속도 및 공간 정보들간의 인식 성능 비교

4.1 방향, 속도 및 공간 정보를 이용한 특징 추출

4.1.1 방향 정보를 이용한 특징 추출

(1) 체인 코드

시간에 따른 손 중심점의 이동 방향은 궤적의 의미 해석에 중요한 정보를 제공한다. 일반적으로 체인 코드

는 연속된 두 점들간의 방향성을 표현하는데 널리 이용되며 방향을 몇으로 세분하느냐에 따라 4-방향 체인 코드, 8-방향 체인 코드, 12-방향 체인 코드 등으로 확대 가능하다.

4-방향 체인 코드는 두 점간의 각도가 90도 이내로 변하지 않으면 같은 코드를 할당하며, 동일한 방법으로 8-방향 체인 코드는 두 점간의 각도가 45도 이내로 변하지 않으면 같은 코드를 할당하다. 체인 코드를 사용할 때는 응용 분야에 따라 몇 방향 체인 코드를 사용할 것인가를 결정해야 한다. 만일 각도를 크게 결정하면 생성되는 코드는 잡음에는 강하나 다양한 형태의 코드를 생성하지는 못한다. 반대로 각도를 작게 하면 코드는 세분화 되나 잡음에 민감하게 된다. 그림 5는 8방향 체인 코드의 예를 보여준다.

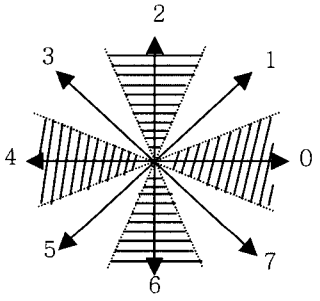


그림 5 8-방향 체인 코드

위 그림에서 점선 안의 부분은 각 코드가 대표하는 각도를 의미한다. 즉, 코드 0은 337.5~22.5도 사이의 방향을 대표하고 코드 2는 67.5~112.5도 사이의 방향을 대표한다. 8-방향 체인코드 생성시 인접한 두 점간의 거리는 x 혹은 y축 방향으로 최소한 1 화소 이상이어야 한다. 만일 두 점간의 거리가 1 화소 밖에 나지 않는다면 코드는 0, 2, 4, 6의 코드만 생성되며 1, 3, 5, 7의 코드는 나타나지 않는다. 또한 인접한 두 점간의 거리가 동일하다면 체인 코드는 정의된 0~7 사이의 코드 값으로 표현 할 수 없다. 일반적으로 이러한 경우는 잘 발생되지 않지만 본 논문에서는 Spotting을 위해 제스처 동작을 멈춰야 하므로 이때 여러 점들이 아주 근접한 위치에서 나타나게 된다. 그러므로 멈춰져 있는 동작 사이에서 발생된 제스처 점들 간의 체인 코드는 연속되는 두 점들간의 거리가 매우 근접하고, 또한 불규칙한 방향 정보를 갖게 된다. 이 경우 두 점간의 거리가 3 화소 이하일 경우에는 의미 없는 방향성의 정보보단 멈춰져 있음을 의미하는 코드 8을 할당한다. 본 논문에서 사용된 8-방향 체인 코드는 멈춰져 있음을 표시하는 코드 8과 움직이고 있는 상태의 8-방향 체인 코드를 합해 총 9개의 코드로 나타난다.

(2) 각도(θ) 코드

체인 코드가 인접한 두 점간의 방향성 코드를 생성하는데 비해 각도 코드는 제스처 중심점으로부터 각 점간의 방향성 코드를 생성한다. 그러므로 먼저 제스처 궤적의 중심점을 구하는 것이 필요하다. 구하고자 하는 중심점을 (C_x, C_y)라 정의하면 중심점은 식 (1)의 Spotting된 궤적 $G_{spotting}$로부터 구할 수 있다.

$$G_{spotting} = \{(x_1, y_1), \cdots, (x_t, y_t), \cdots, (x_N, y_N),\} \; 1 \le t \le n \quad (1)$$

$$(C_x, C_y) = \left(\frac{1}{n}\sum_{t=1}^{n} x_t, \frac{1}{n}\sum_{t=1}^{n} y_t\right) \quad (2)$$

원점이 구해지면 원 점으로부터 각 점간의 각도는 그림 6과 같이 base-line을 기준으로 구할 수 있다. 구해진 각도를 포함하는 집합을 φ라 하면 φ는 다음과 같다.

$$\varphi = \{\theta_1, \theta_2, \cdots, \theta_t, \cdots \theta_n\} \qquad 1 \le t \le n \quad (3)$$

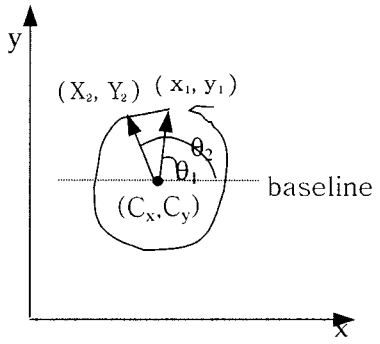


그림 6 각도 코드 생성 예

그림 6과 같이 제스처 궤적으로부터 θ가 생성되면 이를 코드화 하는 방법은 체인 코드와 유사하다. 즉 8-방향 각도 코드일 경우 전체 360도를 8방향으로 45도씩 세분화 하여 코드를 할당하면 된다.

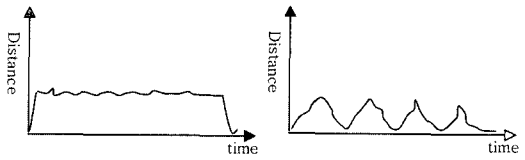
체인 코드와 각도 코드는 방향성을 대표한다는 점에서 동일한 결과를 갖지만 각도 코드일 경우 체인 코드에 비해 멈춰져 있는 상태를 표시할 필요가 없는 장점이 있다. 즉, 체인 코드는 이미 기술한 바와 같이 두 점간에 위치 변화가 없을 때 코드 0를 할당해서 이를 표시하여야 하지만 θ 코드는 이러한 작업을 요구하지 않는다. 다만 이 경우 생성된 코드 구성을 보고 멈춰진 곳인가를 판별하기 어려운 단점도 있다.

4.1.2 속도 정보를 이용한 특징 추출

제스처 인식을 위해 방향 정보만을 이용하는 것보다는 각 제스처를 생성할 때 구할 수 있는 속도 정보를 함께 이용하는 것이 더 좋은 성능을 기대할 수 있다. 이러한 가정은 원과 같은 제스처를 생성할 때는 전체 제스처가 동일한 속도 분포를 갖고, 복잡한 제스처를 생성할 시는 다양한 속도 변화를 갖는다는 사실에 바탕을 둔 것이다. 이러한 제스처의 속도 정보는 제스처 궤적을 이루는 각각의 점들간의 위치 차이 값으로 얻어진다. 본 논문에서는 거리 정보로 두 점간의 속도 및 각 속도를 구하여 사용해 보았다. 먼저 두 점간의 속도를 구하는 공식은 식 (1)의 궤적 $G_{spotting}$로부터 다음과 같이 구할 수 있다.

$$Vd_i = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2} \quad (4)$$

위에서 얻어진 속도 정보를 원과 사각형 도형에 대해 그래프로 그려보면 이상적으로는 그림 7과 같은 형태를 띄게 된다.



(a) 원 제스처의 속도 정보 (b) 네모 제스처의 속도 정보

그림 7 제스처의 속도 정보

그림 7에서 원 제스처와 사각형 제스처간에 속도의 변화는 이상적으로는 매우 잘 구분될 수 있을 만큼의 다른 패턴으로 나타나므로 이 정보를 이용하면 제스처 인식에 도움을 줄 수 있다.

4.1.3 공간 정보를 이용한 특징 추출

방향 및 속도 정보 이외에 제스처 궤적에서 얻을 수 있는 중요한 특징으로 각 점들간의 공간상의 위치 정보를 들 수 있다. 즉, 각 제스처는 2차원 공간상에서 각각 다른 위치들의 정보 집합으로 구성된다. 이러한 공간상의 위치 정보는 제스처를 구별 할 수 있는 중요한 특징으로써, Cartesian(x-y)좌표 공간에서 이를 사용하기 위해선 먼저 MBR(minimum bounding rectangle)을 구한 후 이를 특징 공간으로 분할하는 작업이 필요하다. 제스처 궤적으로부터 MBR을 구하는 방법은 제스처 궤적을 이루는 각 점을 주사(scan) 하면서 가장 큰 x, y 좌표 위치와 가장 작은 x, y 좌표 위치를 찾으면 된다. MBR이 결정되면 MBR안에 위치한 궤적들의 위치 정보를 특징 공간으로 분할해 코드화하게 된다. 공간 정보를 이용하는 또 하나의 방법으로는 x, y 좌표를 극 좌표 공간으로 변환한 후 코드화 하는 방법이 있다. 이 방법은 극 좌표 공간으로 좌표계를 변환할 때 최대 r을 구한 후, 이의 상대적인 값으로 ρ을 설정하므로 자동적으로 제스처의 크기 정규화를 수행할 수 있다는 장점이 있다. Cartesian 좌표계를 가진 식 (1)의 궤적으로부터 극좌표 공간으로 변환된 제스처 궤적을 F라 하면 F는 다음의 단계를 거쳐 생성된다.

$$(C_x, C_y) = \left(\frac{1}{n}\sum_{i=1}^{n} X_i, \frac{1}{n}\sum_{i=1}^{n} Y_i\right) \quad (5)$$

$$\theta_t = \tan^{-1}\frac{Y_t - C_x}{X_t - C_y}, \quad r_t = \sqrt{(X_t - C_x)^2 + (Y_t - C_y)^2} \quad (6)$$

$$r_{max} = \max(\sum_{i=1}^{n} r_i) \quad (7)$$

$$\varphi_t = \frac{\theta_t}{2\pi}, \quad \rho_t = \frac{r_t}{r_{max}} \quad (8)$$

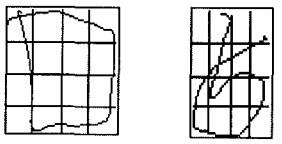
$$F = \{(\rho_1, \varphi_1), (\rho_2, \varphi_2), \cdots, (\rho_t, \varphi_t) \cdots, (\rho_n, \varphi_n)\} \quad (9)$$

4.2 코드 생성(Parameter Estimation)

좌표 공간이 정해지면 이를 분할해 코드화하는 알고리즘이 필요하다. 코드화 알고리즘으로는 일반적으로 많이 사용되는 메쉬(mesh) 및 클러스터링 알고리즘을 이용하였다.

4.2.1 메쉬를 이용한 코드화

메쉬를 이용한 코드화는 좌표 공간을 임의의 개수로 수평, 수직으로 분할한 후, 각각의 사각형 공간을 하나의 코드로 치환하는 것을 말한다. 사각형 및 숫자 5 제스처가 x, y 좌표계에서 수평 방향으로 네 번, 수직 방향으로 네 번 분할된 메쉬 공간상에서 코드화하는 예를 그림 8에서 볼 수 있다.



0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

a) 사각형과 '5' 제스처의 mesh 분할 (b) 코드 지도(map)

G(rectangle) : 0 0 1 2 3 3 7 11 15 15 14 13 12 8 4 0 0

G(five) : 1 1 5 9 9 6 7 11 15 14 13 12 11 8 5 6 3 3

(c) 추출된 특징코드

그림 8 x-y 좌표계에서 메쉬 특징 예

일반적으로 메쉬 공간을 수평, 수직으로 나누어 분할할 때는 정규화 문제점이 발생한다. 즉, MBR의 수직 길이가 여섯 칸이고 나누고자 하는 칸이 네 칸이라면 2

개의 나누어지지 않는 칸이 발생한다. 이 경우 나눗셈의 나머지인 이 두 칸을 보간(interpolation) 하는 알고리즘이 요구된다. 즉, 인식하고자 하는 제스처 궤적의 MBR 크기는 제스처 입력 시마다 가변적으로 변화하기 때문에 얼마로 분할하는 가에 따른 크기 보간법이 필요하다. 이러한 크기 보간 알고리즘은 여러 방법이 있는데 본 시스템에서는 선형 보간법[15]을 사용하였다.

4.2.2 클러스터링(clustering)을 이용한 코드화

메쉬를 이용한 좌표 공간의 코드화는 크기 보간 문제 이외에도 격자를 어떻게 나누어야 최적의 결과를 얻을 수 있는지 판별하는 방법이 없다. 신경망을 이용한 패턴 인식 연구에서 메쉬를 특징 추출에 사용했던 많은 연구들은 실험적 경험 치로서 수평, 수직의 격자를 나누어 왔을 뿐, 나누어진 격자의 크기가 적합한지에 관해선 언급되어 지지 않는다. 이러한 단점은 클러스터링을 이용하면 해결할 수 있다. 클러스터링 알고리즘은 통계적 특성을 이용해 격자를 분할하는 방법이다. 즉, 사용자가 나누고자 하는 코드의 개수를 결정하고 기존에 발생된 코드들의 위치 정보를 입력하면 통계적 방법으로 학습해서 모든 코드가 균일한 분포로 나타날 수 있도록 영역을 분할하는 방법이다. 이러한 클러스터링 알고리즘으로는 대표적으로 k-means[16] 알고리즘과 LBG[10] 알고리즘을 들 수 있다. 본 논문에는 좌표 공간에 대하여 k-means 알고리즘을 사용하여 클러스터링을 적용하였다. k-means 알고리즘을 이용한 클러스터링의 단점은 최적의 클러스터 개수를 알 수 없다는 것이다. 그러므로 본 논문에서는 클러스터링 개수를 4, 8, 16, 24, 48, 64로 나누어 실험한 후, 그 결과를 이용해 가장 높은 인식율을 갖는 클러스터의 개수를 최적의 클러스터 개수로 선택하였다. 그림 9에서 클러스터의 개수 변화에 따른 다이아그램을 Voronoi 다이아그램을 이용해 보여준다. Voronoi 다이아그램은 각 군집들의 중심점을 이용해서 전체 영역이 어떤 군집들의 중심점에 포함되는지를 보여주는 알고리즘이다.

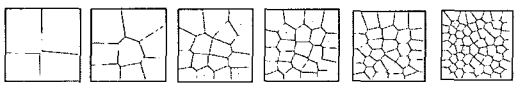


그림 9 클러스터링 개수에 따른 Voronoi 다이아그램

4.3 특징 정보에 따른 인식율의 변동

제스처의 궤적 정보로부터 얻을 수 있는 정보는 여러 가지 운동 역학적 요소들이며 이중 가장 기본적인 특징 정보로 방향, 속도 ,및 위치 정보를 들 수 있다. 거의 모든 제스처 인식 시스템들은 이러한 정보를 기반으로 인식을 수행하는데, 이 특징 정보들 중 어떠한 정보가 가장 유용한지, 또한 방향 정보를 어떻게 나누어 사용할지, 또한 각각의 정보를 어떻게 결합해야 인식기의 성능을 높일 수 있는지에 대해서는 아직까지 연구되어진 바 없다. 본 논문에서는 앞에서 얻어진 상태 수 10개, 학습 및 훈련 데이타가 2400개로 구성된 HMM 제스처 인식기의 구성을 바탕으로 각각의 특징들이 인식에 어떠한 영향을 미치는지 분석하고자 한다.

4.3.1 방향 정보를 이용한 인식

방향 정보를 이용한 실험으로는 앞서 기술한 체인 코드와 기준선으로부터의 각도 정보(θ)에 대하여 인식 결과를 분석한다. 먼저 체인 코드의 실험 결과를 분석하기 위해 하나의 정지코드를 가진 체인 코드를 이용해서 4 방향부터 6, 8, 10, 12 방향까지 5 단계의 체인 코드를 생성한다. 즉 체인 코드로 만들 때 90°부터 60°, 45°, 36°, 30° 로 각 코드가 대표하는 각도를 줄이면서 어떠한 코드가 가장 인식에 적합한지를 찾아본다. 그림 10은 체인 코드의 방향성 세분화에 따른 인식율의 변화를 보여준다.

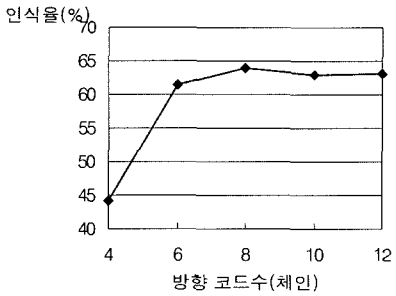


그림 10 체인 코드의 방향성 세분화에 따른 인식율의 변화

그림 10을 분석해 보면 8방향 체인 코드를 사용했을 때 가장 높은 63.8%의 인식율을 얻을 수 있음을 알 수 있다. 즉, 인식하고자 하는 48개의 제스처 궤적은 체인 코드로 인식하고자 할 때 45° 로 각을 나누어 8개의 체인 코드로 정량화 하였을 때 인식율이 가장 높게 나타났다.

앞서 기술한 바와 같이 방향 정보를 사용하는 또 다른 코드로는 각도 코드를 들 수 있다. 각도 코드가 인식에 얼마나 유용한지 평가하기 위해서 체인 코드의 평가와 유사하게 코드가 대표하는 각도를 4에서 6, 8, 10, 12 까지 증가 시키면서 인식율의 변동을 조사하였다. 그림 11에서 그 결과를 볼 수 있다.

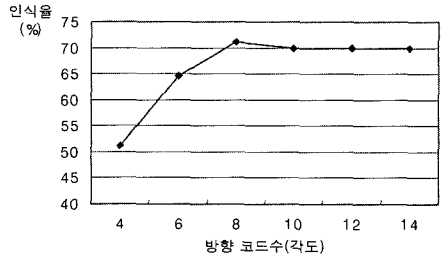


그림 11 θ 코드의 방향성 세분화에 따른 인식율의 변화

그림 11에서 각도 코드의 최적의 인식율은 8로 나누었을 때의 71.01%이다. 같은 방향 코드인 체인 코드에 비해 약간 높은 인식율을 얻었음을 알 수 있다. 또한 체인 코드나 각도 코드 모두 8로 방향을 나누었을 때 가장 좋은 결과를 가짐은 방향성을 세분화해서 코드를 생성할 때 가장 적합한 코드는 8방향성임을 의미한다.

4.3.2 속도 정보를 이용한 인식

속도 정보를 이용한 인식율의 분석은 연속되는 두 점간의 거리를 이용하는 방법과 중심점으로부터의 각 속도를 고려한 방법을 들 수 있다. 먼저 두 점간의 거리를 이용한 특징 추출은 하나의 제스처 궤적에서 연속되는 두 점간의 거리를 구한 후, 이중 가장 큰 값을 1로 정의하고 나머지 값들은 이 값의 상대적인 거리 값으로 치환한다. 이러한 처리를 거치면 속도 정보는 최대 1에서 움직임이 없는 최소 0 사이의 값을 갖게 된다. 이 값을 사용하여 HMM을 이용한 인식을 수행하기 위해서는 최종적으로 간격을 정의해서 코드화해야 한다. 본 논문에서는 간격을 4에서 9까지 나누어 인식율을 실험하였다.

그림 12에서 속도 정보를 이용해 인식율을 검사한 결과 최고 13.83%의 인식율을 얻을 수 있었다. 이런 낮은 인식율은 속도 정보가 다른 정보에 비해 유용하지 못함을 의미한다. 이 같은 결과는 시스템의 OS(operating system)에 의해서도 영향 받는데, 즉, 사용된 Windows98이 시분할 처리로 멀티 테스킹(multi-tasking)을 지원하는 OS이므로 개발된 제스처 인식 시스템이 수행되는 도중에 디스크를 체크 하는 등의 다른 일을 수행하는 경우가 종종 발생된다. 이때 제스처 인식 시스템은 상대적으로 늦게 동작하게 되므로 속도를 이용한 제스처 인식 시 동일한 처리 속도를 보장하지 못한다. 이와 같이 속도가 인식율에 영향을 미치는 정도가 미미함에 따라 각 속도에 따른 인식율의 변화는 실험결과에서 제외하였다.

4.3.3 공간 정보를 이용한 인식

인식하고자 하는 48개의 제스처 궤적은 Cartesian

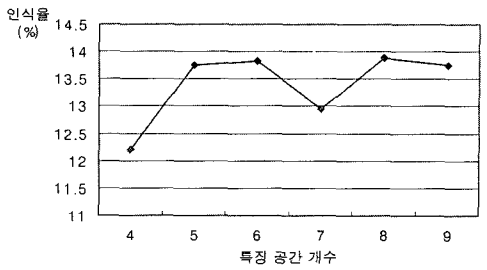


그림 12 속도 정보의 특징 공간 개수 증가에 따른 인식율의 변화

(x-y) 좌표계 혹은 극(ρ-φ) 좌표계 모두에서 독특한 형태를 갖는다. 이러한 각각의 형태에 따라 공간적 좌표를 이용해 특징을 생성하면 좋은 인식 결과를 기대할 수 있다. 본 절에서는 Cartesian 좌표계 및 극 좌표계를 갖는 공간적 좌표계를 메쉬와 클러스터링 알고리즘을 적용해 인식율을 평가해 본다.

(1) Cartesian (x-y) 좌표계

메쉬 알고리즘을 이용한 특징 추출은 앞서 기술한 바와 같이 먼저 하나의 제스처 궤적에 대해 MBR을 구한 후 임의의 크기로 영역을 분할한 다음, 분할된 영역에서 특징을 추출하는 방법이다. 본 논문에서는 최적의 분할 영역 크기를 얻기 위해 4×4부터 4×5, 5×4, 5×5, 5×6, 6×5, 6×6, 6×7로 분할되는 영역을 점차적으로 증가시킨 후, 각각의 영역에서 얻어진 특징 벡터를 이용해 인식율의 변동을 살펴보았다. 이때 4×4분할이란 각각의 제스처 궤적을 포함하는 MBR을 수직으로 네 번, 수평으로 네 번으로 나누어 16개의 직사각형 영역을 대표하는 코드를 부여함을 의미한다. 분할된 코드가 정해지면 각 제스처 점들이 어떠한 코드를 통과했는지 순차적으로 추출하여 특징 벡터를 생성한다. 그림 13은 메쉬 코드의 영역 분할에 따른 인식율의 변화를 보여준다.

그림 13을 분석해 보면 Cartesian 좌표계에서 메쉬 코드를 이용해 특징을 추출한 후, 얻은 인식율은 6×5에서 81.42%의 최적의 인식율을 얻었음을 알 수 있다.

이때 정사각형의 분할이 아니라 수직 길이를 더 세분화했을 때 인식율이 높게 나오는 경향이 있는데 이는 전체 입력 제스처 데이타의 형태가 수평 길이 보다 수직 길이가 상대적으로 큼을 의미한다.

Cartesian 좌표계에서 영역을 분할하는 또 다른 방법으로는 클러스터링 방법을 이용할 수 있다. 메쉬가 전체 영역을 임의로 분할해서 코드화하는 반면 클러스터링을 이용한 방법은 확률을 이용해 영역을 분할하는 것이 다르다.

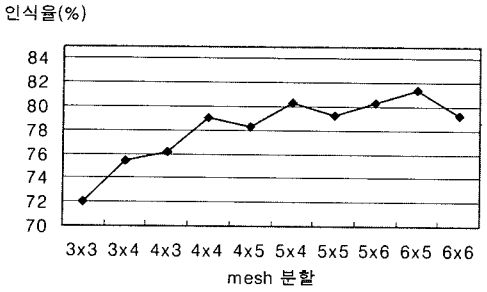


그림 13 Cartesian 좌표계에서 메쉬 코드의 영역 분할에 따른 인식율의 변화

클러스터링 방법을 이용해 공간 특징을 추출하기 위해선 먼저 두 가지 처리를 수행해야 한다. 첫번째 처리로는 제스처 공간의 확률 분포를 측정하기 위한 입력으로 학습 데이타를 생성해야 한다. 즉, 전체 48개의 제스처로부터 5개씩 샘플링 하여 총 240개의 제스처의 좌표 데이타를 클러스터링 학습 데이타로 생성한다. 두 번째 처리로는 Cartesian 좌표계의 정규화가 필요하다. 이를 위해 이미 구해진 MBR의 좌표 코드의 점들을 0에서 1 사이의 값을 갖는 좌표 코드의 점으로 변환한다. 이 같은 좌표 변환은 각 제스처의 크기 정규화를 수행한 것과 같은 역할을 한다. 클러스터링 알고리즘이 수행되면 사용자에 의해 정해진 개수의 클러스터 중심점이 얻어진다. 인식하고자 하는 제스처 궤적의 각 점들은 순차적으로 가장 작은 거리를 갖는 중심점에 대응되어 특징 벡터로 변환된다. 클러스터링 알고리즘을 사용해 얻어진 특징 벡터는 마지막으로 HMM을 통해 인식 작업을 수행하게 된다. 클러스터링 알고리즘을 사용해 인식율을 평가할 때 고려해야 할 중요한 점은 특성 공간을 몇 개로 분할해야 하는가에 있다. 입력 데이타의 패턴에 따라 자동으로 분할 공간의 개수를 찾을 수 있으면 가장 좋으나 이는 인식 결과에 의해 결정되어야 한다. 이를 위해 클러스터 중심점의 개수를 8씩 증가시키면서 인식율이 어떻게 변화하는지 살펴보았다. 그림 14는 군집화 개수를 8부터 16, 24등으로 8씩 순차적으로 증가하면서 얻은 인식율의 변동을 보여준다.

그림 14를 분석해 보면 중심점의 개수를 증가시킬 때 인식율도 점차적으로 증가하다가 32에서 최고의 인식율을 얻고 그 이상 중심점의 개수가 증가하면 인식율이 조금씩 감소함을 알 수 있다. 이와 같은 결과를 일반 메쉬 코드의 결과와 비교해 보면 거의 유사한 결과를 가짐을 알 수 있다. 이 같은 분석은 다양한 모양의 제스처 궤적이 전체 영역에서 편중되지 않고 비교적 균등하게 나타나 메쉬로 영역을 분할한 결과와 클러스터링으로 영역을 분할한 결과가 비슷하게 나타났음을 알 수 있다.

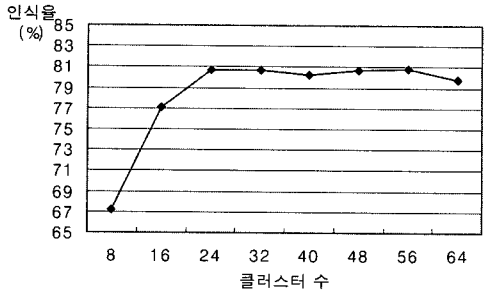
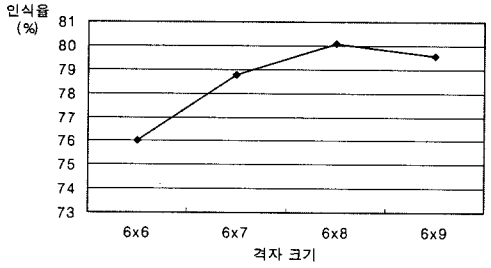


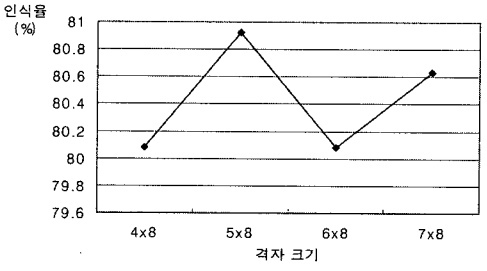
그림 14 Cartesian 좌표 공간의 클러스터 개수에 따른 인식율의 변동

(2) 극 좌표계

공간 정보를 이용해 특징을 추출할 때 극 좌표계를 이용하는 것이 Cartesian 좌표계를 이용하는 것에 비해 제스처 크기 정규화가 쉽게 수행되므로 장점이 있다. 본 절에서는 극 좌표계에서 Cartesian 좌표계에서 수행한 실험과 동일하게 메쉬 분할 및 클러스터링 분할 알고리



(a) ρ 격자 크기 증가에 따른 인식율 변동



(b) φ 격자 크기 증가에 따른 인식율 변동

그림 15 극 좌표계에서 메쉬 코드의 영역 분할에 따른 인식율의 변화

즘을 적용해 인식율을 조사하고 평가해 본다. 그림 15는 극 좌표계에서 메쉬 코드의 영역 분할에 따른 인식율의 변화를 보여준다.

극 좌표계에서 공간 분할은 ρ공간과 φ공간의 중요성에 따라 공간을 나누는 것이 더 효과적이다. 즉, Cartesian 공간과 같이 두 축이 동일한 중요성을 갖는 경우는 x나 y좌표계에 편중되어 영역을 분할 할 수 없으나 ρ축과 φ축을 갖는 극 좌표계에서는 축의 중요성에 따라 공간을 나누는 것이 더 효과적이다.

이를 위해 본 논문에서는 먼저 그림 15의 (a)와 같이 먼저 θ를 고정시키고 r을 변화시키면서 가장 인식율이 높은 ρ를 찾은 후, 그림 (b)에서 γ를 고정시키면서 가장 인식율이 높은 θ를 찾아 5×8 영역에서 최대 인식율 80.92%를 얻었다. 이와 같은 인식율은 앞서 기술한 그림 14의 Cartesian 좌표계에서 메쉬 코드의 영역 분할에 따른 인식율의 변화와 비교 가능하다. 즉, Cartesian 좌표계에서는 5×6의 30개 영역 분할에서 81.42%의 최고 인식율을 얻은 데 비해 극 좌표계에서는 40개 영역 분할에서 최고 인식율 80.92%를 얻었으므로 Cartesian 좌표계 보다 좀더 세밀한 영역 분할이 필요하며 인식율은 상대적으로 약간 낮음을 알 수 있다.

극 좌표계에서 클러스터링을 이용한 인식율 실험은 Cartesian 좌표계에서 수행한 처리와 동일하게 처리된다. 즉, 최적의 클러스터링 개수를 찾기 위해 8부터 8씩 클러스터 개수를 점차적으로 증가 시키면서 최대 64개까지 인식율의 변동을 검사한다. 그림 16은 극 좌표계에서 클러스터 개수에 따른 인식율의 변화를 보여준다. 그림 15와 16을 비교해 보면 메쉬로 영역을 나눈 후 얻은 인식율과 클러스터링으로 영역을 나눈 후 인식율이 매우 유사함을 알 수 있다. 즉, 메쉬를 사용할 경우 5×8의 40개 영역 분할에서 80.92%의 최대 인식율을 얻었고 클러스터링의 경우 48개에서 최대 82.29%의 인식율을 얻었다. 이와 같이 메쉬로 영역을 분할한 경우와 클러스터링을 사용해 분할한 경우는 유사한 개수에서 최대 인식율을 얻었으며 통계적 방법을 이용하지 않은 메쉬 분할보다는 클러스터링을 이용한 방법이 약 1.3%정도의 인식율 향상을 보임을 알 수 있다.

4.3.4 각도, 위치 및 공간 정보를 결합한 인식

이제까지 각도, 위치 및 특징 정보들은 각각 독립적 혹은 둘씩 결합되어 인식되었다. 카티션 좌표계의 x-y 정보는 위치 및 각도 정보가 묵시적으로 결합된 특징을 갖고 있고, 이에 반해 폴라 좌표계의 $\rho-\varphi$ 정보는 위치 및 각도 정보가 명시적으로 결합된 것으로 볼 수 있다. 그러므로 이 3가지 특징 정보를 모두 결합하기 위해선

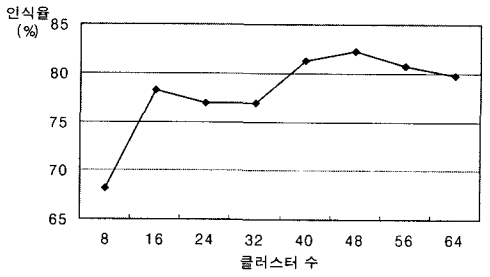
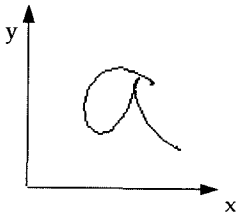


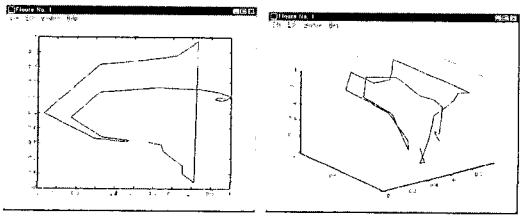
그림 16 극 좌표계에서 클러스터 개수에 따른 인식율의 변화

x-y-v 나 $\rho-\varphi-\upsilon$ 의 3차원 좌표계를 생성하면 된다. 예를 들면 그림 17과 같이 카티션 좌표계의 알파벳 제스처 'a' 는 폴라 좌표계의 $\rho-\varphi$ 및 $\rho-\varphi-\upsilon$의 3차원 좌표계로 생성할 수 있음을 알 수 있다.

3가지 특징이 결합된 좌표계가 생성되면 이를 동일한 크기의 mesh로 분할하거나 k-means알고리즘을 사용해 파라메터 공간으로 변환하고, HMM 인식기를 구현하여 인식율을 검사하면 결합된 특징에 의한 인식율을 얻게 된다. 이와 같이 제스처 궤적에서 생성된 기본정보를 모두 결합하여 인식기에 학습시킴으로써 제스처 궤적의 각도, 위치 및 속도 변화에 따른 효율적인 인식기를 생성할 수 있다.



(a) 알파벳 'a' 제스처



(b) $\rho-\varphi$ (c) $\rho-\varphi-\upsilon$

그림 17 x-y 좌표계의 알파벳 제스처 궤적 'a' 로부터 극 좌표계의 $\rho-\varphi$ 공간과 $\rho-\varphi-\upsilon$ 공간으로의 변환

5. 실험 결과 분석 및 결론

5.1 특징 공간에 따른 인식 결과

본 연구에서는 제스처 인식에 유용한 특징을 평가하기 위해 제스처 궤적에서 얻을 수 있는 가장 기본적인 특징을 방향, 속도 및 공간 정보를 나누어 인식율을 평가하여 보았다.

표 1은 본 연구에서 수행한 각각의 특징들을 이용한 인식 결과를 분리 및 결합하여 비교 평가한 결과이다. 표의 인식 결과에서 제스처 인식에 가장 유용한 특징은 세가지 특징이 결합한 *x-y-v(clustering)* 정보로서 나타났으며 훈련된 데이타에 대해서는 98.79%, 훈련되지 않은 데이타에 대해선 92.17%의 인식율을 얻었다. 인식율이 가장 낮은 특징은 속도 정보만을 특징으로 사용한 것으로서 최저 13.83% 의 인식율을 얻었다. 이에 따라, 속도 정보는 특징으로 가치는 약간 있으나 단독으로 사용하기 보다는 다른 특징과 연계해서 보조로 사용되는 것이 효율적으로 판단된다. 반면에 방향 정보와 공간 정보를 만을 이용한 인식은 단독으로 사용하더라도 일정 수준 이상의 인식율을 얻을 수 있으나 이 세가지 특징 정보가 결합된 결과보다는 못함을 알 수 있다.

5.2 결 론

동적 손 제스처 인식 시스템은 사용자 접속 기술 뿐만 아니라 시각 장애자를 위한 수화인식, VR 시스템에서 가상 공간상의 물체의 취급, 원격지 로봇의 조정, 가전제품의 제어, 컴퓨터에서의 사용자 접속 등과 같은 다양한 응용 분야를 갖는다.

본 연구에서는 카메라로부터 획득된 영상 시퀀스로부터 손 제스처 궤적 위치를 실시간으로 추출하고, 제스처 궤적으로부터 특징 정보 추출 시 단순한 체인 코드 특징 추출에서 탈피하여 카티션 및 극 좌표상에서의 방향 특징, 위치 특징, 속도 특징 등의 다양한 특징들을 실험하여 각각의 특징들이 인식에 어떠한 영향을 미치는지 평가하였고, 또한 이 세 특징 정보를 어떻게 결합하였을 때 가장 인식이 잘 되는지를 제시하였다.

이를 위한 기반 환경으로 mesh나 k-means 알고리즘을 사용하여 최적화된 특징 파라메터를 갖도록 코드화하였으며, 연속적인 시간 데이타에 적합한 통계적인 확률로 결정되는 HMM모델을 인식기로 사용하여 90% 이상의 비교적 높은 인식율을 얻었다.

향 후, 제스처 인식 시스템의 성능을 향상시키기 위해 고려할 사항을 정리하면 다음과 같다. 첫째, 제스처를 학습시킬 때 제스처 형태를 검증하여 학습에 적합한 제스처를 선정해서 학습시킨다. 현재 본 연구에서는 순차적으로 50개의 제스처를 추출하여 학습에 사용하므로 얼마나 좋은 데이타가 학습에 참여했는지 알 수 없다. 둘째, HMM자체는 학습 시 인식 대상인 다른 제스처의 형태를 고려하지 않고 학습함으로써 분별력이 떨어지는 단점이 있다. 이를 해결하기 위해 신경망이나 퍼지 시스템과 연계하거나, 다른 모델에 대한 특성을 살펴보면서 학습할 수 있는 알고리즘을 사용할 수 있다면 인식율이 좋아질 것으로 판단된다. 셋째, 서로 비슷한 형태를 갖는 제스처가 존재하여 인식율이 낮아진다면 제스처의 형태를 잘 구별될 수 있도록 변경한다. 이는 각 제스처 형태를 정의할 때 최대한 원래의 형태를 유지시키면서 one stroke 제스처로 변경하되, 다른 제스처와 구별되도록 정의하면 된다. 넷째, 인식하고자 하는 제스처를 전처리에서 완벽하게 처리하여 인식기로 넘겨줄 수 있도록 영상 처리 부분을 개선한다면 좀더 향상된 인식율을 기대할 수 있다.

표 1 특징 공간에 따른 인식율

Feature class	Feature space	# of feature code	Recognition results (%)		
			Training data	Testing data	Overall
Separated	Chain((1)	8	79.8	63.8	71.8
	Angle((2)	8	80.32	70.01	75.1
	Velocity(v)	4	25.7	13.83	19.7
	x-y(mesh)	30	88.84	81.42	85.13
	x-v(clustering)	32	89.73	82.35	86.04
	ρ-φ(mesh)	40	90.53	80.92	85.7
	ρ-φ(clustering)	48	93.2	82.29	87.7
Combined	*x-v-υ(clustering)*	48	98.79	92.17	95.48
	ρ-φ-υ(clustering)	48	99.08	89.63	94.36

참 고 문 헌

[1] 김종성, 이찬수, 장원, 변증남, "한글 수화용 동적 손 제스처의 실시간 인식 시스템의 구현에 관한 연구," 대한전자공학회 논문지 제34권, C편, 제2호, pp. 61-70. 1997.

[2] 김주홍, 진성일, 이남호, 이용범, "제스처 및 음성 인식을 이용한 윈도우 시스템 제어에 관한 연구," 대한전자공학회 추계종합학술대회 논문집, 제21권 2호, pp. 1289-1292, 1998.

[3] 윤명환, 권오채, 한수미, 박재희, 이경태, "3-D Glove를 이용한 손 동작의 분석 시스템 개발," 대한인간공학회 추계학술대회논문집, pp. 393-397, 1997.

[4] 양선옥, 고일주, 최형일, "제스처 기반 인터페이스를 위한 손 영역 획득 시스템, 한국퍼지 및 지능시스템 논문지, Vol. 8. No. 8. pp. 43-52, 1998.

[5] 이종실, 한영환, 민홍기, 홍승용, "엔트로피 분석을 이용한 수화 영상에서의 손의 추적," 대한전자공학회 하계추계종합학술대회 논문집, Vol.21, No.1, pp. 753-756, 1998.

[6] 고 병기, 양 현승, "새로운 인간 컴퓨터 인터페이스로서의 손가락 움직임 제스처 인식 시스템," 정보과학회 논문집, 제22권, B편, 제11호, pp. 1544-1552, 1995.

[7] 이찬수, 김상원, 박찬종, "손 제스처를 이용한 아바타 동작 제어", 한국감성공학회, 98추계학술대회발표논문집, pp. 12-129, 1998.

[8] H. Hienz, K. Grobel, and G. Offner, "Real-time hand-arm motion analysis using a single video camera," *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, Killington, Vermont, pp. 323-327, October 1996.

[9] J. Yang, Y. Xu, and C. S. Chen, "Human action learning via hidden Markov model", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 27, No. 1, pp. 34-44, January 1997.

[10] T. Nishimura and R. Oka, *"Spotting recognition of human gestures from time-varying images", Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, Killington, Vermont, pp. 318-322, October 1996.

[11] L. W. Campbell, D. A. Becker, A. Azarbayejani, A. F. Bobick, and A. Pentland, "Invariant features for 3-D gesture recognition", *Proceedings of the Second International Conference on Automatic Face and Gesture Recognition*, Killington, Vermont, pp. 157-162, October 1996.

[12] T.E. Tobely, Y. Yoshiki, R. Tsuda, N. Tsuruda and M. Amamiy, "Dynamic hand gesture recognition based on randomized self-organizing map algorithm", *Proceedings of 11th Algorithmic Learning Theory 2000*, pp. 252-263, 2000.

[13] R. Grzeszcuk, G. Bradski, MH. Chu and J-Y. Bouguet, "Stereo based gesture recognition invariant to 3D pose and lighting," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Vol.1, pp. 826-833, 2000.

[14] 민병우, 윤호섭, 소정, "시공간상의 궤적 분석에 의한 제스처 인식", 한국정보과학회논문지(B), 제26권, 제1호, pp. 157-166, 1999.

[15] 최형일, *컴퓨터 비전 입문*, 홍릉과학출판사, 1991.

[16] M.R. Anderberg, *Cluster Analysis for Applications*, Academic Press, 1973.

[17] J. Yang, Y. Xu, and C. S. Chen, "Human action learning via hidden Markov model," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 27, No. 1, pp. 34-44, January 1997.

윤 호 섭

1989년 숭실대학교 전자계산학과 졸업(학사). 1991년 숭실대학교 대학원 전자계산학과 졸업(석사). 2003년 KAIST 전산학과 졸업(박사). 1991년~1998 KIST 시스템공학연구소 영상처리연구부 선임연구원. 1998년~현재 ETRI 컴퓨터소프트웨어 연구소 공간정보기술센터 선임연구원. 관심분야는 영상정보처리, 패턴인식, 인공지능시스템

양 현 승

1976년 서울대학교 전자공학과 졸업(학사). 1983년 Purdue University 전자공학과 졸업(석사). 1986년 Purdue University 전자공학과 졸업(박사). 1981~1982 : Pennsylvania State University Electro-optics Lab. 연구원. 1983~1986 : Purdue University Dept. of EE. Robot Vision Lab. 연구원. 1986~1988 : University of Iowa Dept. of EE. 조교수. 1988~현재 : KAIST 교수. 관심분야는 VR/MR, 휴먼 로보틱스, 지능형 에이전트, 뇌과학/휴먼 비전, 정보 보호