Journal of The Korea Society of Computer and Information
Vol. 17, No. 10, October 2012
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci/2012.17.10.043

약속된 제스처를 이용한 객체 인식 및 추적

배 대 희*, 이 준 환*

Object Detection Using Predefined Gesture and Tracking

Daehee Bae*, Joonhwan Yi*

요 약

본 논문에서는 화면상 약속된 동작을 찾고 추적하는 알고리즘을 이용한 사용자 인터페이스를 제안한다. 현재 frame과 복수의 이전 frame간의 차영상을 이용하여 움직임 영역을 검출하고 약속된 제스처를 취하는 영역을 제어 대상으로 인식한다. 이를 통하여 사용자가 장갑을 사용한다던지, 인종, 피부색등에 구애받지 않고 손동작 영역을 검출해 낼 수 있다. 또한 기존 색체 분포 추적 알고리즘을 개량하여 유사한 배경을 가로지르는 경우의 무게중심 위치의 정확성을 높였다. 그 결과 기존 피부색 인식 방법에 비해 약속된 손동작 인식률의 향상이 있었으며 기존 색체 추적 알고리즘에 비교하여 추적 인식률 향상을 확인할 수 있었다.

▸ Keywords : 휴먼 컴퓨터 인터랙션, 움직임 영역 검출, 손동작 추적, 제스처 인식, 약속된 제스처

Abstract

In the this paper, a gesture-based user interface based on object detection using predefined gesture and the tracking of the detected object is proposed. For object detection, moving objects in a frame are computed by comparing multiple previous frames and predefined gesture is used to detect the target object among those moving objects. Any object with the predefined gesture can be used to control. We also propose an object tracking algorithm, namely density based meanshift algorithm, that uses color distribution of the target objects. The proposed object tracking algorithm tracks a target object crossing the background with a similar color more accurately than existing techniques. Experimental results show that the proposed object detection and tracking algorithms achieve higher detection capability with less computational complexity.

• 제1저자 및 교신저자 : 배대희
• 투고일 : 2012. 09. 11, 심사일 : 2012. 09. 19, 게재확정일 : 2012. 09. 25.
* 광운대학교 컴퓨터공학과(Dept. of Computer Engineering, Kwangwoon University)
* 본 논문은 한국연구재단에서 지원하는 2012년도 일반연구자지원사업(No. 2012-0008329)의 연구수행으로 인한 결과물임을 밝힙니다.
* 이 논문은 2012년도 광운대학교 교내 학술 연구비 지원에 의해 연구되었음.

▸ Keywords : HCI, moving object segmentation, hand tracking, gesture recognition, predefined gesture

I. 서 론

손동작 인식을 이용한 사용자 인터페이스는 인터페이스의 단순성과 직관성에 부합되는 방식이다[1]. 이러한 인터페이스의 단순성과 직관성은 현재 스마트 TV나 mobile device와 같은 다양한 분야에서 핵심적인 경쟁력으로 대두되고 있다. 현재 손동작을 이용한 인터페이스는 별도의 입력 장비 및 복잡한 명령체계를 배제 가능한 새로운 방식의 2차원 사용자 인터페이스라 할 수 있다. 손동작 인식 인터페이스는 상용화의 시작단계로써 카메라와 피사체간의 거리를 이용하여 손을 인식한 후 이를 이용하여 2차원 메뉴제어를 하는 방식은[2] 현재 제품으로 출시되어 판매되고 있으며 다양한 손동작 인식 관련 제품이 출시 준비 중이다[3]. 그러나 기존 출시된 제품의 경우 2차원 메뉴 제어를 목적으로 하는 인터페이스의 경우 특수한 고가의 장비를 요구한다.

손동작 인식은 손 영역 인식과 추적으로 나누어 이루어진다. 기존 손 영역 인식하는 방법에는 일반적인 카메라를 이용한 색체기반, 템플릿 기반, 배경 차분화 방법과 적외선 카메라, 3d-depth 카메라, data glove를 사용하는 방법이 있다.

우선 색체를 기반으로 하는 알고리즘[4][5][6][7]은 사전에 정한 피부색과 일치하는 영역을 손으로 인식한다. 이러한 방법은 적은 연산량을 사용하는 장점이 있다. 하지만 색체에 기반을 둔 손동작 인식 방법은 조명 혹은 사용자의 피부색과 같은 환경적인 요소에 많은 영향을 받는다.

두 번째로 템플릿 기반 손동작 인식방법[8][9][10]은 사전 저장된 템플릿의 양에 따라 다양한 손동작을 인식한다. 손 인식률이 템플릿의 양과 질에 좌우되고, 연산량이 큰 단점이 있다.

세 번째로 배경 차분화를[11][12][13] 사용한 손 인식 방법은 피부색의 영향을 받지 않으며 복잡한 배경에서도 움직이는 영역을 찾아낼 수 있다. 하지만 기존 단일 frame과 현재 frame간의 차를 이용하는 경우[11] 손 영역만을 정확히 추출하기 어렵다. 또한 배경을 학습화 하는 경우는[12] 미세하게 움직이는 영역도 움직임 영역으로 인식하여 손 영역만을 검출하기 어렵고 배경의 초기 학습 시간으로 인해 빠른 명령 입력이 어렵다.

적외선 카메라나 3d-depth 카메라, data glove를 사용하는 방법은[14][15][16] 이러한 환경적 영향을 줄이며 계산량을 줄일 수 있는 방법이다. 하지만 이러한 별도의 영상장비를 사용하는 경우 비용이 크게 증가하는 단점이 있다.

본 논문에서는 화면상 일정 속도 이상으로 움직이는 영역들을 추출하여 이들 중 약속된 동작을 취하는 영역만을 관심영역(Region of interest : ROI)로 인식한다. 일정 속도 이상으로 움직이는 영역을 찾기 위해 복수의 이전 frame들과의 차영상을 이용하는 frame voting이라는 움직임 영역 추출 방법을 제안한다. 움직임 영역만을 인식하기 때문에 특징점이나 색상정보를 이용한 계산이 불필요하다. 또한 일정 속도 이상의 움직임만 추출하기 때문에 배경학습이 필요 없고, 하나의 이전 frame과의 차영상만 이용하는 차분화 방법보다 복잡한 배경에서도 움직이는 영역을 정밀하게 추출하는 장점이 있다. 즉, frame voting과 약속된 동작을 동시에 이용하여 별도의 장비 없이 기존 손 영역 추출 기술들보다 연산량은 적고, 인식률을 높일 수 있다. 약속한 동작을 수행하는 물체라면 색상과 모양에 상관없다는 것 또한 눈에 띄는 장점이다.

추출된 ROI에 대하여 추적을 진행한다. 화면상의 객체에 대한 추적 방법에는 색체에 대한 추적방법과 특징점을 이용한 추적방법이 존재한다. 특징점을 기반을 둔 optical flow 방법[17]은 배경과 물체가 매우 유사한 형태 및 색을 지니더라도 강건한 성능을 발휘할 수 있다. 하지만 화면 전체의 각각의 특징점에 대하여 motion vector를 계산하여야 함으로 많은 연산량을 가지고 있다는 단점이 존재한다.

색체를 이용한 추적 방법에는 continuously adaptive meanshift(CAMshift)[18]와 이를 계량한 알고리즘인 adaptive background CAMshift (ABCshift)[19] 등이 있었다. 이러한 색체 추적 알고리즘은 배경과 물체가 비슷한 색체를 지닐 경우 좋은 성능을 유지하기 힘들다. 하지만 optical flow에 비하여 연산량이 적음으로 실시간 처리에 용이하다는 장점이 있다. 본 논문에서는 기존 색체 추적 알고리즘의 단점을 보완하기 위하여 배경과 물체의 유사성에 적게 영향을 받는 알고리즘인 density based meanshift (DBMshift)알고리즘을 사용한다.

본 논문은 다음과 같이 구성되었다. Ⅱ장은 관련연구를 고찰하며 Ⅲ장은 제안하는 사용자 인터페이스 알고리즘의 전체 개요를 서술한다. Ⅳ장은 frame voting을 서술하고 Ⅴ장에서는 DBMshift를 제안한다. 그리고 Ⅵ장은 실험 결과를 보여주고 마지막 Ⅶ장 결론으로 마무리한다.

II. 관련 연구

본 논문은 약속된 제스처를 검출하기 위해 화면상 배경을 차분화 하여 움직이는 물체를 검출하고 검출된 물체의 색체를 추적하는 방법을 사용한다. 배경 차분화 및 색체 추적 방법의 기존 방법은 다음과 같다.

1. 기존 배경 차분화 방법

배경 차분화 방법에는 frame간의 차를 이용한 방법과 배경 학습화를 이용한 방법이 있다. Frame 간의 차를 이용하는 방법으로 Lipton *et al.*[20]은 현재 frame과 이전 frame의 각 픽셀의 값의 차를 이용하여 해당 차의 값이 일정 임계값 이상인 경우 해당 픽셀을 움직임 영역으로 설정하는 알고리즘을 제안했다. 이 알고리즘은 연산량이 매우 적다는 장점이 있다. 하지만 느리게 움직이는 물체는 움직임 영역(blob)에 홀(hole)이 발생하고 빠르게 움직이는 물체는 잔상이 생긴다는 단점이 있다.

Zivkovic *et al.*[21]은 이전의 frame들에 대하여 혼합 가우시안 모델 (mixture of gaussian model :MoG)을 통해 배경 모델링을 하는 방법을 제안하였다. 해당 알고리즘은 각 픽셀에 대한 이전 frame들의 색상 변화에 대하여 복수개의 커널(kernel)을 지닌 가우시안 분포를 생성하여 배경을 모델링한다. 이 방법은 배경의 변화에 강건하며 비교적 정확한 움직임 영역을 추출할 수 있다는 장점이 있다. 하지만 초기 시스템 작동 시 배경을 모델링하는 동안 움직임 영역을 인식할 수 없으며 오랫동안 물체가 정지한 경우 해당 영역을 배경으로 인식할 수 있다. 또한 미세하게 움직이는 영역도 움직임 영역으로 인식하기 때문에 손동작만을 labeling하기 어렵다는 문제가 있다.

2. 기존 색체 추적 방법

기존 동영상에서 색체 분포 영역을 추적하는 방법에는 meanshift를 응용한 CAMshift, ABCshift algorithm 등이 있다. Meanshift는 한 장의 frame 영역 상에서 찾고자하는 색체 표본이 존재할 확률이 가장 높은 위치에 무게중심점을 수렴시키는 확률 밀도 추정 방법이다.

이러한 meanshift를 환경의 변화가 심한 동영상에 적용하기위해 Bladski *et al.*는 CAMshift[18]를 제안하였다. CAMshift는 meanshift에 색상 학습의 개념을 추가시켰다. 그림 1a는 색체 분포의 탐색을 위한 초기 윈도우 영역 W영역을 설정하는 것을 보여준다. 그림 1.b는 W내부 좌표 (x, y)에 위치하는 화소 $I(x, y)$들의 hue값 h를 표본데이터로 하는 색상 확률 질량함수 $f_{hue}(h)$를 그래프로 나타낸 것이다. 즉 $N_w(h)$을 W내의 화소 중 hue값이 h인 화소의 수라 하고 $N_w(*)$을 W내의 총 화소수라 한다면 $f_{hue}(h)$는 아래 수식과 같다.

$$f_{hue}(h) = \frac{N_w(h)}{N_w(*)} \qquad (1)$$

그림 1.c는 W내 화소 $I(x, y)$의 hue값 h에 해당하는 $f_{hue}(h)$의 값을 대입한 함수 $f_w(x,y)$를 색체 그래프로 나타낸 것이다. 가로와 세로축은 각각 화소의 좌표축인 x와 y축을 의미하고, 각 화소좌표 (x,y)에 있는 화소 $I(x,y)$의 hue값 h에 따른 $f_{hue}(h)$값에 따라 다른 색을 표시하였다. 파란색으로 갈수록 $f_{hue}(h)$이 0에 가깝고, 붉은색으로 갈수록 $f_{hue}(h)$이 1에 가깝다. 함수 $f_w(x,y)$를 이용하여 무게중심점의 좌표 (x_m, y_m)를 계산한다[19]. 무게중심점을 구하는 수식은 아래 수식을 따른다.

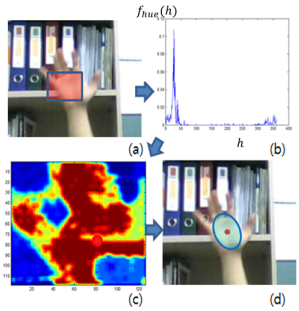


그림 1. CAMshift의 개념도 (a) 손 영역 설정 (b) h의 색상확률 분포 (c) h에 대한 색체 그래프 (d) 갱신된 ROI
Fig. 1. Diagram of CAMshift (a) set of hand region (b) color probability of h (c) color graph of h (d) renewed ROI

$$x_m = \frac{\sum_x \sum_y x f_w(x,y)}{\sum_x \sum_y f_w(x,y)}, y_m = \frac{\sum_x \sum_y y f_w(x,y)}{\sum_x \sum_y f_w(x,y)} \qquad (2)$$

그림 1.d는 그림 1.c의 그래프를 이용하여 생성된 무게중심점의 좌표 (x_m, y_m)로 W의 위치를 변경하며 W의 크기를 새롭게 갱신한 그림을 보여준다. CAMshift의 과정은 아래와 같다.

Step 1. 초기 W영역의 중심점 (x_c, y_c)와 크기 설정

Step 2. $f_{hue}(h)$를 생성

Step 3. $f_w(x,y)$를 생성

Step 4. $f_w(x,y)$이용 무게중심좌표 $(x_m,\ y_m)$ 생성

Step 5. $(x_m,\ y_m)$을 중심점으로 하는 W영역 재설정

Step 6. 새로운 W영역의 $(x_c,\ y_c)$와 $(x_m,\ y_m)$사이의 거리가 0에 수렴할 때 까지 Step 2 ~ 5 반복

Step 7. 새로운 frame으로 갱신

Step 8. Step 2 ~ 7 반복

CAMshift는 매 frame 마다 W의 크기와 위치를 갱신하여 색상의 확률분포함수 $f_{hue}(h)$를 새롭게 생성한다. 따라서 조명의 변화에 대처가 가능하다. 하지만 배경을 고려하지 않고 물체의 색체 변화만을 고려하므로 유사한 색을 갖는 배경에 머물 경우 배경과 물체를 구분하지 못한다는 단점이 있다.

R.Stolkin *et al.*은 이러한 단점을 보완하기 위하여 ABCshift[19]를 제안하였다. ABCshift는 CAMshift의 단점을 보완하기 위하여 W영역을 배경 영역 ABR(adjacent background region)와 관심영역 ROI(region of interest)으로 나누었다. 이는 그림 2로 나타낸다. 이를 이용해 Bayesian rule을 기반으로 W내의 임의의 화소 $I(x,\ y)$의 hue값이 h일 때, 이 화소가 ROI에 존재할 확률 $P[ROI|h]$를 구할 수 있다. 식(3)에서 W내부에 $I(x,\ y)$좌표가 h라는 색 값을 가질 확률 $P[h|W]$은 ROI의 색체 확률 $P[h|ROI]$와 ABR의 색체 확률 $P[h|ABR]$의합으로 이루어 졌다는 것을 보여준다.

$$P[h|W]=P[h|ROI]P[ROI]+P[h|ABR]P[ABR] \quad (3)$$

식(3)의 $P[h|ABR]$는 ROI 내부에 $I(x,\ y)$ 좌표가 h라는 색 값을 가질 확률을 나타낸 것으로 아래 식(4)로 구할 수 있다.

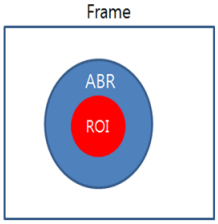


그림 2. ABCshift의 영역 분할
Fig. 2. Region Division of ABCshift

$$P[h|ROI]=\frac{N_{ROI}(h)}{N_{ROI}} \quad (4)$$

위 식(4)에서 N_{ROI}는 ROI 내 화소의 수이다. $N_{ROI}(h)$는 ROI 내 화소 중 h와 같은 hue 값을 지니는 화소의 수이다. 식 (3)에서 $P[ROI]$와 $P[ABR]$는 각각 W 내부에 ROI와 ABR의 픽셀이 몇 개가 존재하는지에 대한 확률이다. 식(5)는 이와 같은 방법으로 인접 ABR 내부에 h값이 분포되는 확률 $P[h|ABR]$을 구하는 것을 보여준다.

$$P(h|ABR)=\frac{N_{ABR}(h)}{N_{ABR}} \quad (5)$$

식(6)은 $P[h|W]$와 $P[h|ROI]$를 알면, 임의의 화소 $I(x,\ y)$의 색체 값 h가 주어졌을 때, $I(x,\ y)$가 ROI내에 있을 확률 $P[ROI|h]$을 구하는 식을 보여준다.

$$P[ROI|h]=\frac{P[h|ROI]P[ROI]}{P[h|W]} \quad (6)$$

하지만 ABCshift의 경우 $f_w(x,y)$의 ABR영역에도 각 $I(x,y)$가 ROI일 확률 $P[ROI|h]$에 확률 값이 분포함으로 부정확한 무게중심을 설정한다는 문제가 발생한다. 이러한 문제는 만일 ROI가 비슷한 색을 지닌 ABR을 가로지르는 경우 ABR로 ROI의 무게중심을 빼앗기는 상황을 발생시킬 수 있다. 아래 그림 3은 구하고자 하는 ROI이 ABR상의 $P[ROI|h]$ 분포로 인하여 잘못된 ROI의 위치를 설정하는 예를 보여준다. 그림 3상에서 손 뒤에 존재하는 책장은 손과 유사한 색이다. 따라서 책장의 영역 즉 ABR에는 손의 영역의 색 분포인 $P[ROI|h]$가 분포함으로 책장 방향으로 ROI의 위치가 이동하게 된다.

III. 사용자 인터페이스 전체 개요

그림 4는 본 논문에서 제안하는 시스템의 flowchart를 보여준다. 처음 영상이 입력된 경우, 아직 손 영역이 추출되지 않았으므로 추적모드가 아닌 손 영역 추출을 수행한다. 본 논문에서는 화면 내에 약속된 제스처를 가진 물체를 추출하기 위하여 화면상에 움직임 영역을 frame voting 이라는 알고리즘을 이용하여 추출해 낸다. 움직임 영역을 구한 후 구해낸 움직임 영역의 집합(blob)에 대하여 모폴로지 연산을 사용하여 noise를 제거한다. 노이즈가 제거된 각각의 움직임 영역에 대해 DBSCAN[22]을 이용하여 별도의 객체로 분할(clustering)하는 과정을 거친다. 이렇게 분할된 각각의 객체의 움직임이 사용자가 정의한 제스처를 취하는지 여부를 판별하기 위해 본 논문에서는 Viterbi 알고리즘으로 각 물체의 움직임 벡터 시퀀스(moving vector sequence)를 고려하여 사용자가 원하는 제스처를 취하는 물체를 판별해 낸다. 해당 과정을 거치면 사용자는 약속된 동작을 취하는 영역을 손 영

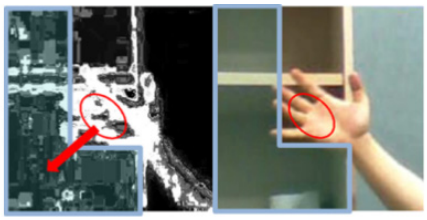


그림 3. ABR 확률분포에 의한 잘못된 ROI 설정
Fig. 3. Wrong ROI location due to the similar color distribution in ABR

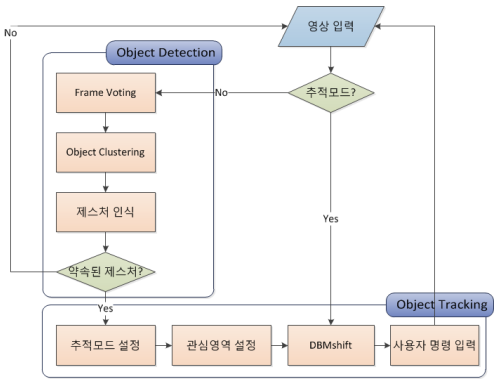


그림 4. 제안하는 사용자 인터페이스 개요
Fig. 4. Overall flow of the proposed user interface

역으로 인식할 수 있게 된다. 손 영역이 인식 된 이후부터는 추적모드를 시행하게 된다. 인식된 손 영역을 ROI으로 설정한 후 이후 frame부터 해당 영역을 추적하기 위하여 우리는 색체 추적 알고리즘인 CAMshift의 변형 알고리즘인 DBMshift를 제안하여 사용하였다. DBMshift를 통해 손 영역의 추적을 시작하게 되면 DBMshift를 통해 얻어진 손 영역의 모양 및 ROI내부의 특징점 등의 정보를 통해 사용자로부터 2차원 메뉴 제어를 수행할 수 있게 된다.

IV. Frame voting을 이용한 움직임 영역 추출

1. 움직임 영역 검출

Frame voting은 n번째 frame 상의 좌표 (x, y)에 위치하는 화소 $I_n(x,y)$와 이전 s개의 frame 상의 화소 $I_k(x,y)$, $n-s \le k < n$, 값을 비교한다. 여기서 $s \ge 3$인 홀수이다. 조명 등 frame간 촬영환경에 따라 같은 물체의 색을 촬영 후 화소 값으로 정량화했을 때 미세한 차이를 보일 수 있다. 이러한 환경변화에 따른 미세한 색의 차이를 무시하고, 실제로 다른 화소 값을 판단하기 위해 임의의 임계값 τ를 설정하여 화소들을 비교한다. 즉, $|I_n(x,y)-I_k(x,y)|>\tau$ 이면 n번째 frame과 k번째 frame에 있는 화소의 색이 다르다 판단하고, 그렇지 않으면 같은 색이라 판단한다. $I_n(x, y)$

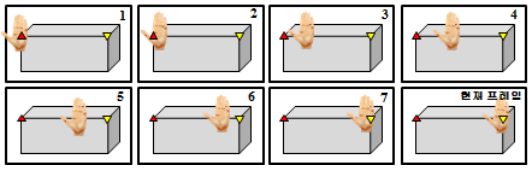


그림 5 Frame voting 예 (frame 수 s = 7)
Fig. 5. Example of frame voting (number of frame s = 7)

와 모든 $I_k(x, y), n-s \le k < n$,를 비교하여 $\lceil s/2 \rceil$ 보다 많거나 같은 수의 frame에서 화소색이 다르면, $I_n(x, y)$는 움직이는 물체의 화소라 판단한다. 여기서 $\lceil a \rceil$는 ceiling(올림)함수로 임의의 정수 b에 대하여 $b-1 \le a < b$인 경우 $\lceil a \rceil = b$ 이다.

그림 5는 frame voting 알고리즘의 예를 보여준다. 위 그림에서 손 모양의 도형은 화면상 회색 사각형 위를 좌측에서 우측으로 이동한다. 위 그림에서는 $s=7$이다. 현재 frame에서 역삼각형이 표시하는 화소 $I_n(x,y)$는 움직이는 손에 위치한다. 그리고 이전 7개의 frame 중 같은 위치의 화소가 다른 색, 즉 손이 아닌 회색 사각형에 속하는 frame의 수는 6개이다. 이 6개의 frame 번호는 {1, 2, 3, 4, 5, 6}이다. Frame voting 알고리즘에 의하면 $\lceil s/2 \rceil = 4$ 보다 6이 크므로 $I_n(x,y)$은 움직이는 물체에 속한다.

현재 frame n이전에 s개의 frame 중 현재 frame과 다른 색의 화소로 인식되는 frame의 수 $V_n(x,y)$를 수식으로 나타내면 아래와 같이 나타낼 수 있다.

$$D(x,y,n,k) = \begin{cases} 1\,(\mathrm{if}\,|I_n(x,y)-I_k(x,y)|>\tau) \\ 0\,(otherwise) \end{cases} \quad (7)$$

$$V_n = \sum_{k=n-s}^{n-1} D(x, y, n, k) \quad (8)$$

$D(x,y,n,k)$는 두 frame n과 k의 화소 $I_n(x, y)$와 $I_k(x,y)$의 색 값이 임계값 τ보다 크면, 즉 다른 색이면 1을 그렇지 않으면 0을 갖는다.

$$M_n(x,y) = \begin{cases} 1\,\mathrm{if}\,(\,V_n(x,y) > \lceil s/2 \rceil\,) \\ 0\,(otherwise) \end{cases} \quad (9)$$

식(9)에서 보이는 바와 같이 $V_n(x,y)$의 값이 $\lceil s/2 \rceil$ 보다 크면 움직이는 물체의 화소로($M_n(x,y)=1$), 아니면 움직이지 않은 화소로($M_n(x,y)=0$) 판별한다.

Frame voting에서 움직임 영역이 온전하게 나오기 위해서는 frame간의 시간 간격과 움직이는 물체의 크기가 고려되어야 한다. 위 그림 6a는 찾고자하는 물체가 온전하게 나오는 예시를 보여준다. 그림 6a에서 길이가 l인 물체가 현재 frame까지 2^*l만큼 움직인다. 그림 6e는 현재 frame에서 화소 $I(x, y)$별 $V_n(x,y)$ 값을 무늬로 나타낸 그림이다. 식 (5)에 따라 현재 frame과 이전 frame들 가운데 $\lceil s/2 \rceil$ 이상의 frame이 다르면 그 영역은 움직임영역으로 인식한다. 따라서 그림 6c의 회색 원 전체가 움직임 영역으로 추출된다. 반면 그림 6b의 경우 크기 l인 물체가 l만큼 이동하게 되는데

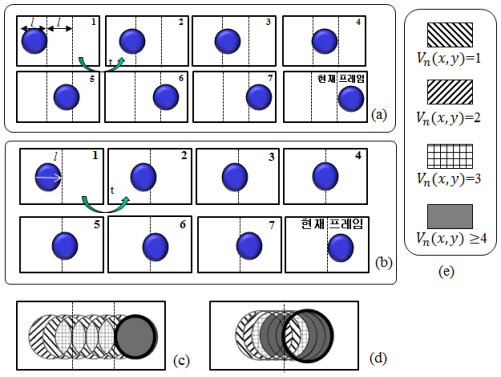


그림 6. Frame voting의 움직임 영역 추출 정확도 (a) 정확도가 높은 예 (b) hole 혹은 잔상이 발생하는 예 (c) 그림 6a의 현재 frame에서 $V_n(x,y)$ 분포 (d) 그림 6a의 현재 frame에서 $V_n(x,y)$ 분포 (e) $V_n(x,y)$에 대한 그림 범례
Fig. 6. Accuracy of moving object detection by frame voting: (a) example with accurate detection (b) example with holes and afterimages (c) $V_n(x,y)$ of the current frame in Fig 6a (d) $V_n(x,y)$ of the current frame in Fig 6b (e) Figure legend of $V_n(x,y)$

이때의 일련의 frame동안 $V_n(x, y)$의 분포는 그림 6d와 같다. 그림 6d에서 굵은 색 테두리의 원은 현재 frame에서의 물체 위치이다. 여기서 현재 frame의 내부에는 $V_n(x, y) < 4$인 영역, 외부에는 $V_n(x, y) \geq 4$인 영역이 존재한다. 따라서 결과 움직임 영역은 현재 frame에서 물체가 위치한 영역 외부에 잔상 및 물체 내부에 hole이 발생되어지게 된다.

따라서 그림 6을 보면 $\lceil s/2 \rceil$ frame 동안 물체는 현재 크기인 l의 위치에서 완벽히 벗어나야 온전한 움직임 영역을 검출할 수 있다는 것을 알 수 있다. 즉 위 그림을 통해 물체는 물체가 특정 방향으로 움직이는 속도를 $\vec{v}$라 하고 frame 사이의 간격이 되는 시간을 t라 할 때 움직이는 물체의 속도 $\vec{v}$는 아래의 수식을 만족하여야 온전히 움직이는 물체를 검출할 수 있다.

$$\vec{v} \geq \frac{l}{\lceil s/2 \rceil t} \qquad (10)$$

제안하는 사용자 인터페이스는 약속한 제스처를 이용하는데, 약속한 제스처를 정의할 때 식 (10)의 조건을 고려하여 움직이는 물체가 잘 검출될 수 있도록 설계하는 것이 바람직하다.

예를 들어, 거실과 같은 일상 가정환경에서 TV를 제어하는 것을 가정하자. 우리는 카메라로 촬영되는 이미지의 넓이 대비 1/8의 크기 이하를 손의 너비 l로 설정하였다. 즉 frame의 너비가 320픽셀이라면 l은 40픽셀 이하일 때, 잔상이나 홀이 생기지 않게 된다. 물론, 일부 홀이나 잔상이 생기더라도 약속된 동작을 이용하여 손 영역을 찾는 수는 있지만, 가급적 식 (6)에서 제시된 조건을 만족시키는 것이 좋다. 만일 카메라와 사용자간의 거리가 매우 가깝다면 화면상의 손 영역인 l이 커짐으로 요구되어지는 $\vec{v}$가 커진다. 식 (6)에서

그림 7 급격한 움직임에서의 결과 검출 (a) temporal subtraction (b) frame voting
Fig. 7. Detection of a quickly moving object (a) temporal subtraction (b) frame voting

비교되는 frame의 수 s 는 본 논문의 실험에서 7을 사용하였고 t의 경우 본 논문의 실험상으로는 1/30초(30fps)를 사용하였다. 사용자가 의도적으로 약속된 특정 제스처를 취할 때 (↳, ⇌, O 같은 도형) 손의 움직이는 속도는 비의도적으로 손을 움직일 때보다 빠르다. 이를 고려하였을 때 약속된 제스처를 취하는 손의 속도 $\vec{v}$는 $\lceil s/2 \rceil * t$초(본 논문의 실험에서는 $4/30 \fallingdotseq 0.13$초)안에 손이 손의 크기인 l만큼 이동할 수 있어야 한다. 만약 요구되는 속도 $\vec{v}$가 너무 크면, 비교하는 frame의 간격을 넓혀 t를 크게 만들 수 있다. 이와 같이 l과 t는 카메라와 사용자 사이의 거리 및 비교 frame간의 시간차로 조정이 가능하다. 약속된 동작의 속도는 주변 움직이는 물체와 제어대상 물체- 예를 들어 손 -를 구별할 수 있도록 결정하고, l과 t는 frame voting으로 제어대상 물체를 정확하게 추출할 수 있도록 계산하여 적용한다.

그림 7a는 이전 frame과 현재 frame의 차를 이용하여 구해낸 움직임 영역이며 그림 7b는 frame voting을 통해 구해

진 움직임 영역이다. 두 그림을 비교하면 그림 7a의 경우 물체가 두 개로 나오는 현상이 발생하며 영역의 집합에 구멍(hole)이 발생하며 손 영역이 두 개로 나오는 것을 볼 수 있다. 하지만 그림 7b의 경우 물체가 단일 영상으로 나오며 구멍이 이전보다 작아지는 것을 확인할 수 있다. s값이 커질수록 홀의 크기는 작아진다.

2. 모폴로지 연산 및 군집화

이렇게 구해진 움직임 영역에서 우리는 각 이미지는 카메라 렌즈 및 여타 연산 과정 가운데 zero-mean Gaussian noise 가 포함된다고 가정하였다. 따라서 구해진 움직임 영역에 노이즈가 존재함으로 각 픽셀을 중심으로 일정 영역의 마스크 영역의 평균적인 픽셀 값을 계산하여 모폴로지 연산을 수행하였다.

화면상에는 다수의 움직임 영역이 존재할 수 있음으로 각각의 움직임 영역을 군집화 하는 작업이 필요하게 된다. 본 논문에서는 이러한 군집화 작업에서 DBSCAN[22] 알고리즘을 사용하였다. DBSCAN 알고리즘의 경우 노이즈 제거와 군집화를 동시해 진행해 주지만 모든 픽셀에 대하여 재귀적 연산을 수행하기 때문에 연산량이 매우 크다. 따라서 본 논문에서는 연산량을 줄이기 위하여 각 픽셀을 5x5 픽셀을 간격으로 block화하여 계산하였다.

3. 약속된 제스처 인식

현재 frame의 움직임 영역을 구한 후 해당 움직임 영역이 사용자가 원하는 제스처를 취하는지 여부를 분석해야 한다. 본 논문은 패턴 매치 알고리즘에서 매우 많이 사용되어지는 알고리즘인 비터비(viterbi) 알고리즘[23] 을 사용하여 움직임 시퀀스를 모델링 하였다. 비터비 알고리즘은 과거의 상태만을 사용하여 현재의 상태를 추측하는 대표적인 방법 중 하나이다. 본 논문에서는 n번째 frame을 F_n, n-1번째 frame을 F_{n-1}로 나타낸다. DBSCAN을 통해 한 덩어리로 인식된 물체 Obj가 두 frame F_n과 F_{n-1}에서 Obj_n과 Obj_{n-1}로 표현된다 하자. Obj_n가 충분히 느리게 움직이고, 겹쳐진 물체가 없다면, Obj_n과 Obj_{n-1}의 중심점 c_n와 c_{n-1}은 가장 가까운 거리에 위치한다. 여기서 각 Obj의 중심점은 DBSCAN을 통해 생성된 단일 색을 지닌 blob에 대한 무게중심점을 의미한다. 이러한 일련의 무게중심점을 이용하여 현재 frame의 물체의 중심점 c_n에서부터 일련의 s개의 frame의 물체들의 무게중심점 c_{n-s}까지 최소 거리를 지니는 물체에 대한 시퀀스를 모델링하고 이 모델링된 시퀀스와 현재 사전에 약속된 제스처와 비교하는 작업을 수행하게 된다. 본 논문에서는 손을 좌, 우로 적어도 2초안에 5번을 반복적으로 움직이는 것을 약속된 동작으로 가정하였다. 해당 과정을 통하여 화면 내에 움직임 영역 중 약속된 제스처를 취하는 움직임 영역을 손 영역으로 추적하게 된다.

V. Density Based Meanshift를 통한 손 영역 추적 및 인터페이스 설계

본 논문에서는 W 내부의 ROI와 ROI의 영역 외의 ABR 간에 우도비(likelihood ratio)를 구하여 ABR상 존재하는 ROI의 유사확률 분포를 줄일 수 있는 density based meanshift (DBMshift) 알고리즘을 제안한다. DBMshift는 ABR상에 ROI 유사 확률을 분포시키지 않음으로써, II장에서 언급한 ABCshift의 문제, 즉 ABR상에 ROI 유사확률이 분포해서 인식률이 떨어지는 문제를 해결 가능하다. 우선 W영역을 ABCshift와 동일하게 ROI와 ABR 두 클래스로 분할한다. 역시 기존 ABCshift와 동일하게 bayesian rule을 통하여 W 영역의 픽셀 $I(x, y)$에 대해 ROI일 확률과 ABR일 확률을 구하게 된다. 그러나 이를 이용해 W영역의 각 픽셀에 대하여 ROI에 속할 경우 1, ABR에 속할 경우 0으로 대입하여 W영역의 ABR상에 ROI 유사 확률을 분포시키지 않도록 한다. 이렇게 함으로써 ROI와 유사한 색을 지닌 영역을 지날 때 ABR 영역 상으로 ROI의 무게중심을 빼앗길 확률을 줄일 수 있게 된다.

이를 구하는 방식은 아래와 같다. 우선 ROI 내부 혹은 ABR 내부에 $I(x, y)$ 좌표가 h라는 색 값을 가질 확률을 앞서 식(4)와 식(5)를 통해 $P(h|ROI)$와 $P(h|ABR)$로 구한바 있다.

여기서 $I(x, y)$의 값이 h로 주어졌을 때 그 값이 ROI인지 ABR인지의 여부를 알 수 있는 식은 아래와 같이 구할 수 있다.

$$P(ROI|h) \quad (11)$$

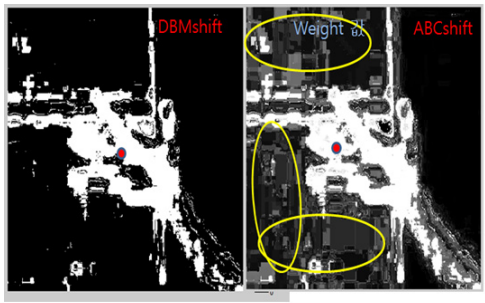
$$P(ABR|h) \quad (12)$$

위 식 9, 10은 Bayesian 정리를 통해 다음과 같이 성립시킬 수 있다.

$$P(ROI|h) = \frac{P(h|ROI)}{P(h|W)} P(ROI) \quad (13)$$

$$P(ABR|h) = \frac{P(h|ABR)}{P(h|W)} P(ABR) \quad (14)$$

위 두 결과는 픽셀 $I(x, y)$가 특정 색체 h일 때 ROI일 확률 혹은 ABR일 확률을 알 수 있다. 현재 W는 ABR과 ROI



(a) (b)

그림 8. ABR상의 ROI 유사 확률 분포가 무게 중심점에 미치는 영향: (a) DBMshift (b) ABCshift
Fig. 8. Effects on centroid when the color distribution of ABR is similar to that of ROI : (a) DBMshift and (b) ABCshift

라는 두 가지 영역으로 나뉘어져 있기 때문에 각 h에 대하여 어떤 영역에 속하는지를 likelihood ratio를 사용하여 계산할 수 있다.

$$R = \frac{P(ROI|h)}{P(ABR|h)} = \frac{P(ROI)P(h|ROI)}{P(ABR)P(h|ABR)} \quad (15)$$

여기서 $P(ROI)$, $P(ABR)$은 각각 전체 W의 면적상에서 ROI와 ABR이 어느 정도 차지하는지에 대한 확률임으로 아래와 같이 정의할 수 있다.

$$\frac{P(ABR)}{P(ROI)} = \tau \quad (16)$$

$$L(c) = \frac{P(h|ROI)}{P(h|ABR)} \quad (17)$$

따라서 아래 식 18과 같이 ABR과 ROI 두 영역중 하나를 선택할 수 있게 된다.

$$f_w(x,y) = \begin{cases} 1 \ (ROI) & \text{if}(L(c) \geq \tau) \\ 0 \ (ABR) & otherwize \end{cases} \quad (18)$$

DBMshift를 이용하여 W 내부 모든 픽셀에 대하여 ROI일 확률 혹은 ABR일 확률로 나누어 이진 영상을 만든다면 그림 8a와 같은 결과를 얻을 수 있다. 그림상의 붉은 점은 물체의 무게중심을 나타낸다. 그림 8b는 ABCshift를 통해 생성된 영상으로 ABR에 손 영역과 비슷한 색이 분포하는 경우 해당 영역에 ROI일 확률 값이 분포하게 되며 무게중심 역시 ABR 영역 방향으로 이동되어 있는 것을 확인할 수 있다.

손 영역이 검출되었을 때 2차원 메뉴제어를 위해서는 click과 같은 다양한 명령을 인식할 필요가 있다. 이러한 방법은 이미 몇몇 논문에서 발표한바 있다[24]. 이를 통해 우리는 손 영역의 손가락 개수를 인식하여 2차원 메뉴 제어 인터페이스 어플리케이션을 만들 수 있었다.

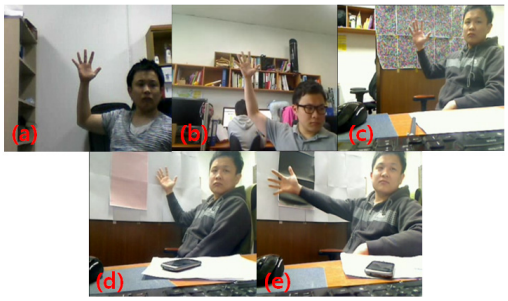


그림 9. 실험에 사용된 다양한 배경
Fig. 9. Various backgrounds used in the experiment

VI. 실 험

1. 실험 환경

제안된 약속된 손동작 인식 알고리즘의 실험하기 위하여 다양한 배경에 대한 손동작 데이터베이스를 구축하였다. 각 실험에는 손 영역 검출 평가의 경우 3가지 배경에 각 20개의 동영상을 사용하여 검증하였으며 손 영역 추적 평가에는 4가지 배경에 대하여 각 20개의 동영상을 사용하여 검증하였다. 각 영상의 해상도는 320*240이며 30fps의 재생속도를 가진다. 실험에 사용된 PC환경은 AMD Phenom II X3 710 processor 2.60Ghz의 CPU와 DDR3 4GB의 RAM, Window 7의 운영체제를 사용하였다. 또한 알고리즘의 구현은 C++와 OpenCV library를 사용하였다.

2. 손 영역 검출 평가

본 논문은 세 가지 배경을 가정하여 약속된 동작을 통한 손 영역 검출 평가를 시행한다. 각 배경은 흰색 배경(그림 9a), 피부색과 유사한 영역이 존재하는 일상 환경의 배경(그림 9b), 임의로 생성된 픽셀을 출력하여 생성된 배경(그림 9c)으로 실험을 진행하였다. 약속된 동작의 검출은 화면상에 존재하는 이동 검출 영역 중 30frame 이내에 좌, 우로 5번 이상 움직이는 영역을 손 영역으로 검출하도록 하였다.

손 영역을 검출할 때 사용되는 방법은 본 논문에서 제안하는 frame voting 방법과 화면상에 피부색을 추출하는 방법[4][5]와 이전 frame과 현재 frame 간의 차영상을 이용하는 방법[11], 배경에 대한 모델링 방법[12]을 사용하였다. 각 알고리즘을 이용하여 손동작 인식률과 초당 처리 가능한

frame 수를 측정 비교하였다.

그림 10은 각 알고리즘의 손동작 인식률을 보인다. 피부색을 통한 손 영역 검출방법은 배경 상에 피부색과 유사한 영역이 존재할 경우(그림 9b) 성능이 급격히 떨어지는 것을 확인하였다. 이러한 문제를 해결하기 위해 피부색 영역 추출 임계

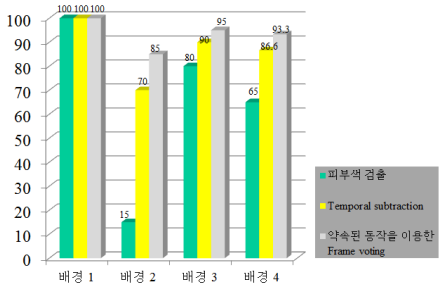


그림 10. 손 영역 인식률 비교
Fig. 10. Comparison of hand detection rates for various algorithms

값의 범위를 좁힐 경우 다양한 피부색에 대응하지 못하게 되는 문제가 발생하였다. 이전의 frame들에 대하여 혼합 가우시안 모델 (mixture of gaussian model :MoG)을 이용한 배경 학습을 단독으로 사용할 경우 움직임 영역으로 검출되어지는 영역이 너무 커 성능을 측정 불가능하였다. Frame voting의 경우 피부색과 유사한 색을 지니는 배경(그림 9b) 상에서도 85%의 인식률을 보이며 임의로 생성된 픽셀로 이루어진 배경(그림 9c) 상에서도 95%의 성능을 보임으로 90%의 성능을 보인 temporal subtraction나 80%의 성능을 보인 피부색 인식의 방법보다 더 높은 성능을 지니는 것을 그림 10을 통해 알 수 있다. 본 논문의 실험에서는 화면상 단일 손동작을 찾아내도록 설계하였다. 앞에서 설명한 DBSCAN과 같은 군집화 알고리즘을 통해 화면상 복수의 움직이는 물체에 대한 개별적인 움직임 시퀀스를 찾을 수 있었다. 이때 화면상 복수의 약속된 제스처를 취하는 움직임 영역이 존재할 수 있는데 이 경우 본 논문에서는 복수의 움직임 중 약속된 제스처를 취하는 물체가 하나만 존재할 때 까지 목표물로 인식하지 않는다. 이를 사전에 방지하기 위하여 약속된 제스처를 사용자가 의도적으로 해야만 하는 동작 혹은 일정 속도 이상의 제스처로 정하는 것이 중요하다.

그림 11은 각 알고리즘의 초당 처리 가능한 frame수를 보인다. MoG의 동작속도는 13.04fps로 가장 떨어지는 것을 확인하였다. 또한 피부색 검출과 MoG는 부동소수점을 사용함으로 실행환경이 FPU를 사용하지 않는 환경이라면 두 경우는 실행 속도가 더욱 떨어질 것으로 예상된다. 수행 속도의 경우 피부색 검출 알고리즘에서 21.53fps의 성능이 측정되었

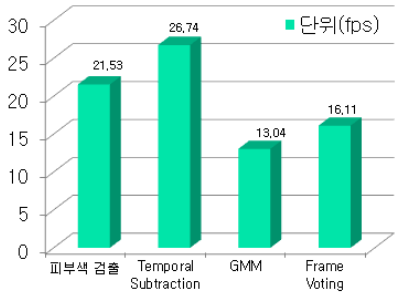


그림 11 손 영역 검출 속도 비교 (초당 처리가능 frame수가 클수록 수행속도 빠름)
Fig. 11. Comparison of computational complexity Higher frame rate is better

으며 frame voting 알고리즘은 16.11fps를 나타내었다. Frame voting의 경우 현재 frame과 이전 여러 장의 frame을 비교해야 함으로 temporal subtraction 및 피부색 검출 방법보다 수행시간이 오래 걸리는 것을 확인할 수 있었다.

3. 손 영역 추적 평가

손 영역 추적 평가에서는 네 가지 환경에 대하여 성능을 평가하였다. 살색 책장이 존재하는 일상 환경(그림 9b)과 임의로 생성된 픽셀을 출력하여 생성된 배경(그림 9c), 흰색 바탕에 피부색 영역을 가로지르는 배경(그림 9d), 흰색 배경에 검은색 영역을 가로지르는 배경(그림 9e)에서 검증하였다. 비교되는 방법은 ABCshift와 ABCshift의 결과를 일정 threshold를 적용하여 1 혹은 0으로 정규화 시킨 방법, DBMshift의 결과를 비교하였다. 실험방법은 배경상의 장애가 되는 구간을 손이 통과하였을 때 추적이 유지되어지는가에 대한 분석을 시행하였다.

일상 배경의 경우 살색 영역(책장)을 지나갈 때 모든 알고리즘의 성능이 감소하였다. 하지만 DBMshift의 경우 감소하는 폭이 80%로 적은 것을 확인하였다. 또한 기존 알고리즘의 경우 흰색 배경에서 검은색 영역을 통과하는 경우 성능이 떨어지는 것으로 확인되었다. 이 는 검은색 배경에서 흰색 배경으로 손이 이동할 때 손의 색과 흰색 배경의 색을 동일한 색으로 인식하게 된다. 그로인해 배경 방향으로 무게중심이 이동되어져 기존 알고리즘의 인식률이 급격히 떨어지는 것을 확인할 수 있었다. 하지만 DBMshift의 경우 흰색 배경 상에 ROI유사확률이 분포하지 않음으로 90%의 높은 인식률을 확인할 수 있었다. 그림 12를 통하여 DBMshift의 경우 모든 배경 상에서 평균적인 인식률은 92.5%로 기존 알고리즘에 비해 향상된 성능을 확인할 수 있다.

Ⅶ. 결 론

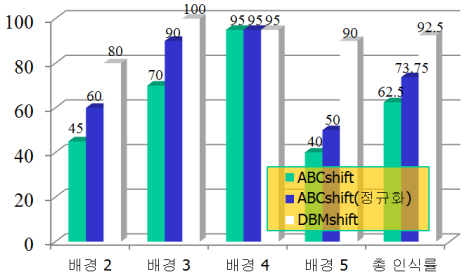


그림 12. 추적 알고리즘 별 인식률 비교
Fig. 12. Comparison of tracking accuracy for various object tracking algorithms

본 논문은 화면상에 약속된 동작을 취하는 물체를 제어 대상으로 가정하여 ROI으로 설정하였다. 이를 통해 배경 및 피부색에 영향을 비교적 적게 받고 원하는 손 영역만을 찾을 수 있다는 장점을 가진다. 약속된 손동작 인식에 적합한 움직임 영역 인식 기술로 frame voting은 색체를 기반 기술에 비해 배경의 영향을 적게 받으며 이전 단일 frame과의 차를 이용하는 기술의 잔상 문제를 효과적으로 줄인다. 이를 통하여 약속된 손동작 인식에 있어 기존 손동작 인식방법들 보다 높은 인식률을 확인할 수 있었다. 손 영역 추적의 정확도를 높이기 위하여 배경 상 ROI 유사 확률 분포를 제거한 DBMshift를 제안하였다. 이를 통해 손 영역이 피부색과 유사한 영역을 지나칠 경우에도 배경에 ROI을 빼앗길 확률을 줄일 수 있었으며 복잡한 배경 상에서 ABCshift보다 높은 인식률을 확인할 수 있었다. 차후 연구 내용으로는 검출된 ROI영역 내부의 손 모양을 인식을 통해 별도의 명령을 입력할 수 있도록 만드는 것이다.

감사의 글

본 연구 수행에 큰 도움을 주신 고려대학교 오성준 교수님과 김경원 학생에게 감사의 뜻을 전합니다.

참고문헌

[1] W.O.Galitz "The Essential Guide to User Interface Design," John Wiley & Sons Inc, pp. 127-129, 2007.
[2] Microsoft Xbox Kinect, http://www.xbox.com/ko-KR/Kinect?xr=shellnav
[3] LEAP leafmotion, http://www.leapmotion.com/
[4] J.H.Yun, and C.H.Lee, "Design of Computer Vision Interface by Recognizing Hand Motion," Journal of The Institute of Electronics Engineers of Korea, Vol. 47, No. 3, pp. 256-265, Apr 2010.
[5] A.Cheddad, J.Condell, K.Curran and M.Kevitt, "A Skin Tone Detection Algorithm for an Adaptive Approach to Steganography," in Proc. of Signal Processing, Vol. 89, No. 12, pp. 2465-2478, Dec 2009.
[6] E. Stergiopoulou and N. Papamarkos, "Hand Gesture Recognition Using a Neural Network Shape Fitting Technique," in Proc Engineering Applications of Artificial Intelligence, Vol. 22, No. 8 pp. 1141-1158, Dec 2009.
[7] S.H.Kim, Y.H.Woo and K.E.Lee, "Implementation of Mouse Function Using Web Camera and Hand," Journal of the Korea society of computer and information, Vol. 15, No. 74, pp. 33-38, Apl 2010.
[8] Z.Pan, Y.Li, M.Zhang, C,Sun, K.Gou, X.Tang and S.Z.Zhou, "A Real-Time Multi-cue Hand Tracking Algorithm based on Computer Vision," in Proc 2010 IEEE Virtual Reality Conference, pp. 219-222, Mar 2010.
[9] Y.Fang, J.Cheng, J.Wang, K.Wang, J.Liu and H.Lu, "Hand Posture Recognition with Co-Traning," in Proc 19th International Conference on Pattern Recognition, pp. 1-4, Dec 2008.
[10] S.H.Lee, H.S.Han and Y.J.Han, "The Estimation of Hand Pose Based on Mean-Shift Tracking Using the Fusion of Color and Depth Information for Marker-less Augmented Reality," Journal of the Korea society of computer and information, Vol. 17, No. 7, pp. 155-166, Jul 2012.
[11] J.J.Young, K.H.Jang, J.H.Lee and J.S.Moon, "A Robust Method for the Recognition of Dynamic Hand Gestures based on DSTW," Journal of The Institute of Electronics Engineers of Korea, Vol. 47, No. 1, pp. 92-103, Jan 2010.
[12] J.Guo, Y.Liu, C.Chang and H.Nguyen,

"Improved Hand Tracking System," IEEE Trans. on Circuits and System for Video Technology, Vol. 22, No. 5, pp. 693-701, May 2012.

〔13〕 S.M.M.Roomi, R.J.Priya and H.Jayalakshmi, "Hand Gesture Recognition For Human-Computer Interaction". Journal of Computer Science, Vol. 6, No. 9, pp. 994-999, Jun 2010.

〔14〕 Chan-Su Lee and Shin-Won Park, "Tracking Hand Rotation and Grasping from an IR Camera Using Cylindrical Manifold Embedding," in Proc of 2010 ICPR, pp.2612-2615, Aug 2010.

〔15〕 Van den Bergh,M and Van Gool, L, "Combining RGB and ToF Cameras for Real-Time 3D Hand Gesture Interaction," in Proc. of IEEE Workshop on Application of Computer Vision(WACV), pp. 66-72, Jan 2011.

〔16〕 J.H.Kim, N.D.Thang and T.S.Kim, "3-D Hand Motion Tracking and Gesture Recognition Using a Data Glove," in Proc. of IEEE Symposium on Industrial Electronic, pp. 1013-1018, Jul 2009.

〔17〕 J.L. Barron, D.J. Fleet and S.S. Beauchemin, "Performance of Optical Flow Techniques," Proc of Internatioal Journal of Computer Vision, pp. 236-242, Jun 1994.

〔18〕 G.R.Bradski, "Computer Vision Face Tracking for Use in a Perceptual User Interface," in Proc Interface, Vol.2, No.2, pp.12-21, Feb 1998.

〔19〕 R.Stolkin, I.Florescu, M.Baron, C.Harriar and B.Kocherov, "Efficient Visual Servoing with the ABCshift Tracking Algorithm," in Proc IEEE ICRA 2008, pp. 19-23, May 2008.

〔20〕 A.J.Lipton, H.Fujiyoshi and R.S.Patil, "Moving Target Classification and Tracking from Real Time Video," in Proc IEEE WACV'98, pp. 8-14, Oct 1998

〔21〕 Z.Zivkovic, "Improved Adaptive Gaussian Mixture Model for Background Subtraction," in Proc ICPR'2004, pp. 28-31, Aug 2004

〔22〕 M. Ester, H.P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," Proc. of 2nd International Conference on Knowledge Discovery and Data Mining. FLEXChip Signal Processor, pp. 226-231, Jun 1996.

〔23〕 M. Elmezain, A. Al-Hamadi, and B. Michaelis, "Real-time Capable System for Hand Gesture Recognition Using Hidden Markov Models in Stereo Color Image Sequences," The Journal of conferences in Central Europe on Computer Graphics, Vol. 16, No. 1, pp. 65-72, Jan 2008.

〔24〕 S.Y.Park, E.J.Lee, "Hand Gesture Recognition Algorithm Robust to Complex Image," Journal of Korea Multimedia Society, Vol. 13, No. 7, pp. 1000-1015, Jul 2010.

저 자 소 개

배 대 희

2011 : 광운대학교 컴퓨터공학과 공학사

2011년 ~ 현재 : 광운대학교 컴퓨터공학과 석사과정

관심분야 : Computer Vision, Human Computer Interaction, SoC 설계

Email : placebo1014@kw.ac.kr

이 준 환

1991 : 연세대학교 전자공학과 학사 졸업

1998 : Univ. of Michigan, EECS 석사 졸업

2002 : Univ. of Michigan, EECS 박사 졸업

1991 ~ 1995년 : 삼성전자 시스템LSI 연구원

2003 ~ 2008년 : 삼성전자 통신연구소 수석연구원

2008 ~ 현재 : 광운대학교 컴퓨터공학과 부교수

관심분야 : SoC 구조설계, 저전력 설계, Computer Vision, 반도체설계

Email : joonhwan.yi@kw.ac.kr