컬러 영상의 스킨컬러 정보 값 기반 손 제스처 인식 기술 개발

저자1 신나라○ 저자2 허범근 저자3 홍기천
수원대학교 정보통신공학과
저자1 nara9n@suwon.ac.kr, 저자2 heo0928@naver.com, 저자3 kchong@suwon.ac.kr

Development of Hand Gesture Recognition Technic based on Image's Skin Color Information

저자1 Shin Na Ra○ 저자2 Heo Bum Geun 저자3 Hong Ki Cheon
University of Suwon Telecommunication engineering

요 약

본 논문에서는 인간의 신체 중 가장 편리하고 자주 사용되는 손을 인식하여 기계와 사람의 상호작용이 가능하게 하여 보다 편리한 기계의 사용이 이루어질 수 있도록 손인식을 통한 마우스제어 방법을제시하였다. CCD웹캠으로 받은 영상에서 스킨컬러 값을 기반으로 손영역을 추출하여 손동작을 인식하는 알고리즘을 제안한다. 본 논문의 알고리즘의 장점은 손의 추출과 손가락 개수 파악이 저 사항의 환경에서도 빠르게 영상처리가 가능하다는 것이다. 이러한 장점을 이용하여 실생활에 적용, 기계와 더 편리한 커뮤니케이션의 방안을 제시한다.

1. 서 론

오랫동안 인간의 손을 이용한 여러 가지 신호가 상호간의 정보를 교환하기 위한 수단으로 사용되어왔다. 복잡한 실생활 환경에서 특별한 장비 없이 손동작을 이용해 컴퓨터와 사용자가 상호작용을 할 수 있다면 편리할 것이다. 이에 본 논문에서는 사람과 컴퓨터가 효율적으로 상호작용을 할 수 있는 간단한 방법에 관하여 연구하였다.

사람과 컴퓨터의 상호작용 방법으로는 크게 음성과 비전을 이용하는 방법이 있다. 현재 음성을 이용한 방법은 어느 정도 안정화되고 실생활에 적용 되어가고 있다. 비전을 이용하는 방법은 입력된 영상에서 사람의 몸짓이나 손, 얼굴 등을 인식하여 사람과 컴퓨터가 상호작용을 하는 방법이다. 이중 표현력이 뛰어나고 널리 이용되는 것이 손동작을 사용하는 것이다. 기존의 비전을 이용한 손동작 인식방법으로는 장비를 이용한 방법과 장비 없이 실제 손 영상에 대해 비전을 이용한 방법이 있다. 장비를 이용한 방법은 사용자가 장갑을 착용하고, 장갑에 부착된 센서로부터 입력된 신호를 이용하여 손동작을 인식하는 방법이다. 이 방법은 실생활에서 이용하기 어렵다. 이 문제에 대한 접근 방법으로서, 본 논문에서는, 특별한 장비 없이 웹캠만을 이용하여 손동작을 인식하는 방법을 연구하였다.

손 영역의 위치정보와 손가락의 개수 정보를 사용하여 손의 움직임을 실시간으로 추적할 수 있는 방법을 제안한다. 또한, 본 논문에서 제안된 손동작 인식 및 추적 방법을 응용하여 마우스 없이 손을 이용해 컴퓨터를 제어할 수 있도록 구현하였다.

2. 본 론

2.1 손동작 인식 및 추적

주어진 영상에서 손영역을 검출하는 방법으로는 두 가지 방법이 있다. 영상에 손이 들어오기 전에 배경 이미지를 저장하고 이 배경 이미지와 매 프레임에 들어오는 영상 간의 차를 이용하면 손영역이 검출 되는 방법이 있다. 다른 방법으로는 영상 안에서 RGB영역 기반의 스킨 컬러 값 또는 YCbCr, HSV 영역에서의 스킨컬러 값을 기반으로 해서 영상에서 살색 부분을 찾음으로써 영상에서 손 부분을 검출 하는 방법이 있다. 전자는 영상의 영상처리의 시간이 느려져서 실시간으로 하기엔 적합하지 않아서 후자의 방법으로 손영역을 검출하였다.

2.1.1 OpenCV의 전체적인 패턴

Visual Studio 6.0과 OpenCV 1.0의 환경에서 프로그램을 제작하였다. OpenCV의 영상처리의 전체적인 패턴은 그림 1에 이해되기 쉽게 표시하였다.

* 본 연구는 경기도의 경기도지역협력연구센터사업의 일환으로 수행하였음[지능형 감시를 위한 이벤트 추출 시스템 개발]
** 수원대학교 정보통신공학과 석사과정
*** 수원대학교 정보통신공학과 교수(교신저자)

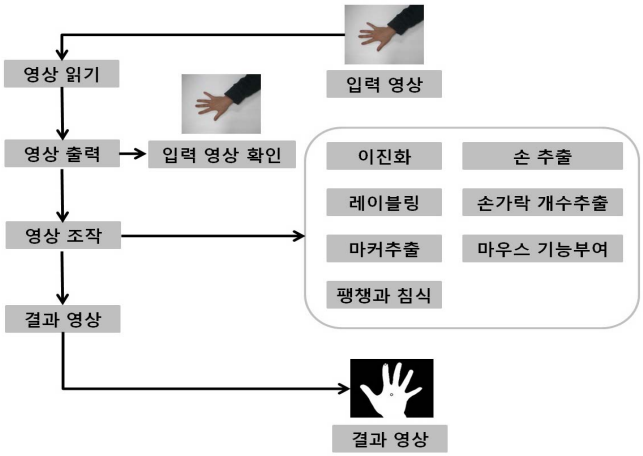


그림 1 OpenCV의 전체적인 패턴

캠을 통해 영상이 입력되면 제작한 프로그램을 통해 영상을 읽고 입력 영상이 제대로 출력 되는지 확인하게 된다. 그 다음 영상을 조작하게 되는 앞으로 설명하게 될 이진화 레이블링등 영상을 프로그램을 통해 조작하게 된다. 조작 결과, 나온 영상을 최종 목표 영상으로 프로그램을 제작하였다.

2.1.2 스킨컬러 값 기반의 손영역 추출

본 논문에서는 칼라 영상에서 대상을 분할하기 위해 일반적으로 칼라 정보를 이용한다. 분할하려는 대상물이 움직이는 대상이므로 연속 프레임간의 자기 차이에 의한 움직임정보와 피부색 정보를 함께 이용하여 대상을 분할하였다. 이때 밝기 변화에 영향을 받지 않고 피부색을 분할하기 위해 정규화 된 rg 색상정보를 이용하였다. RGB칼라 공간에서 밝기 계산식은 <수식1>과 같고, 정규화 된 rg 색상 변환식은 <수식2>와 같다.

$$\text{밝기}(I) = R+G+B \,/\, 3$$

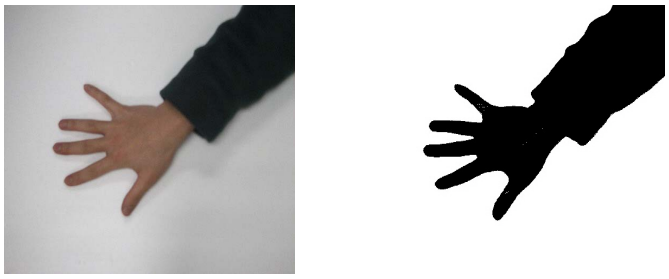
<수식 1> RGB공간에서의 밝기

$$r = R \,/\, R+G+B \quad , \quad g = G \,/\, R+G+B$$

<수식 2> 정규화 된 rg 색상 변환

2.1.3 영상 이진화

입력 받은 영상을 0 과 1로만 나타내어 흑과 백색의 계통만을 이용하여 영상을 변환 시키는 작업을 한다. 이진화의 이유는 손영역 추출을 위한 선 작업인데 외곽선을 추출하기 위한 작업이다. 이진화 작업은 RGB 각각을 이진화 하기 때문에 의도하지 않은 영상을 얻게 될 수도 있다. 그러한 이유로 Gray로 변환 후 이진화 하는 것이 의도한 영상을 얻을 수 있다.

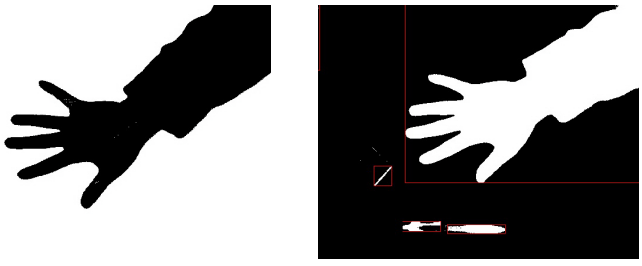


(a) 원영상 (b) 이진화된 영상

그림 2 이진화 된 영상

2.1.4 레이블링

이진화된 영상에서의 컴포넌트를 잘 나타내기 위한 방법이다. 인접한 화소에는 같은 번호(Label)을 붙이고 연결되지 않은 다른 성분에는 다른 번호를 붙이는 작업을 한다. 영상 전체 픽셀을 검사하는데, 해당 픽셀의 이전 방문 여부와 되돌아갈 곳의 위치를 저장하는데 구조체를 선언한다. 또한 레이블링시 레이블이 저장될 이미지와 각 레이블의 정보가 담길 각각의 변수들을 선언한다. 레이블링이 수행된 뒤, 각 레이블의 정보를 저장하고 있는 변수로부터 각 레이블의 정보를 가져와 화면 출력을 위해 만들어둔 이미지에 각각의 영역을 표시한다.



(a) 이진화 영상 (b) 레이블링된 영상

그림 3 레이블링 작업후 영상

작성한 레이블링 알고리즘은 Foreground로 판단하기 때문에, 앞서 이진화된 영상을 반전하게 된다.

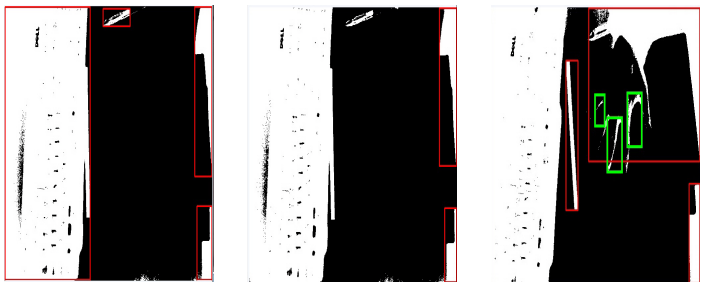
2.1.5 마커 추출과 검증

위 레이블 된 영상에서 잡영 여부를 판단하여 이를 제거하게 된다. 각각의 레이블들을 지정한 넓이와 높이 값과 비교하여, 큰 값들은 저장하게 되고 작은 값들은 버린다. 본 논문에서는 영상의 80% 이상, 30pix 이하의

레이블들은 모두 제거하였다.

앞선 작업을 통해 검출된 마커 후보영역들 중에서 실제 마커 영역이 가지는 특징을 가지는 것을 최종적으로 남기는 검증단계를 구현한다. 레이블의 내무에 홀이 있는지 없는지 파악한 뒤, 그 홀의 위치와 크기를 외부 레이블의 그것과 비교해 보고, 현재 레이블이 마커인지 아닌지 아래와 같이 구분해 낸다. 홀을 찾는다는 것은 반전된 이미지의 개체를 찾는 것 과 같다. 가가 레이블들의 가로 세로 정보를 토대로 각 레이블을 담을 새로운 이미지를 생성한 뒤, 각각의 레이블 이미지를 반전하여 담게 된다.

반전된 이미지를 레이블링 한 뒤, 그것의 크기가 테두리 레이블 크기의 40% 이상, 80% 이하인 것들만 남기고 제거한다. 적절한 수치는 마커의 크기에 따라 테스트를 계속 하면서 바꿔줄 수 있다. 반전된 이미지의 레이블이 테두리 레이블의 가장자리에 있다는 것은, 이 테두리 레이블이 닫힌 형태가 아니라는 것, 즉 홀이 존재하는 것이 아니라는 말이므로 삭제한다. 녹색의 네모가 홀을 검증해낸 것이다. 반전된 이미지의 레이블이 테두리 레이블의 가장자리에 있다는 것은, 이 테두리 레이블이 닫힌 형태가 아니라는 것, 즉 홀이 존재하는 것이 아니라는 말이므로 삭제한다. 녹색의 네모가 홀을 검증해낸 것이다.

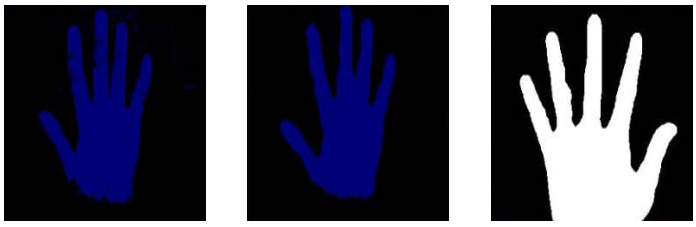


(a)잡영 제거 전 (b)잡영 제거 후 (c)마커 검증 영상

그림 4 잡영 제거후 마커 검증 과정

2.1.6 팽창과 침식 연산

위 과정을 거치고 난 뒤 3차원 영상 이미지에서 파란색으로 구분된 영역을 1차원 흰색 영상으로 변환하고 팽창과 침식연산을 통해 영상을 보정한다. 또한 손의 원활화하기 위해서 OpenCV 라이브러리의 Edge Detection 함수를 통해 소벨 에지 처리를 한다.



(a) 노이즈 제거 전 (b) 노이즈 제거 후 (c) 결과 영상

그림 5 침식과 팽창연산 과정

위의 과정(팽창 침식 침식 침식 팽창)을 통해서 손영역 부분의 홀과 노이즈를 없애게 된다.

2.1.7 손 중심과 손가락 추출

처리된 영상을 가지고 손의 중심 좌표를 찾는다. 손의 중심을 찾는 방법은 화면상의 흰색 부분의 x좌표의 값과 y좌쵸의 값을 모두 더한 뒤 흰색의 카운트로 나누는 방법으로 간단하게 중심을 찾을 수 있다. 손의 중심값을 가지고 있는 함수가 리턴하는 값으로 이 후 마우스 처리 기능을 구현하면 마우스 포인터의 값이 많이 떨리게 된다. 이를 보정해 주어야 하는데, 보정은 평균값을 이용한 보정을 사용한다. 보정 과정을 거치면 좌표의 값이 안정화가 되어 나중에 기능 구현을 했을 때 더 정확하게 처리 할 수 있다.

이젠 마우스의 포인터가 따라다녀 할 손가락을 찾아야 한다. 본 논문에서는 검지로 정하였고 방법은 손의 중심을 잡은 상태에서 손 중심으로부터 일정한 각도 ceta 값 240 ~ 275 으로 정해주고, 손 중심으로부터 일정한 거리까지를 살펴보면서 가장 끝 부분의 좌표값을 리턴해 주게 된다.

검지의 손끝을 찾은 후 손의 중심좌표를 평균값을 이용하여 포인트 보정을 해준 것처럼 검지의 손끝도 마찬가지로 평균값을 이용하여 포인트 보정을 처리한다. 검지 손끝은 마우스의 이동기능을 담당하는 부분이기에 이 평균값 보정이 꼭 필요한 부분이라고 할 수 있다.

2.1.8 손 끝에 따른 마우스 포인터 이동

해상도와 모니터에 따라 영상에 나오는 손끝의 좌표값이 다르기 때문에 좀더 섬세한 마우스 포인터의 움직임을 위해 x , y좌표값에 따른 상대적 이동값을 계산하였다. 영상을 통해 입력받은 x , y 좌표를 X 이고 상대적 이동값의 좌표는 offX이다. 입력영상 640 X 460 해상도에 최적화를 위해 accRate의 값은 0.7 linRate의 값은 3을 주었다.

offX = (x * x * accRate + x) * linRate
offY = (y * y * accRate + y) * linRate

<수식 3> x , y값에 상대적 이동값 계산

2.2 손가락 개수와 마우스 기능부여

2.2.1 손가락 개수 추출

손 중심과 검지 끝을 찾았으므로 어제 손가락의 개수를 추출하는 작업을 한다. 손가락의 개수 추출 방법은 중심으로 일정거리부터 임의로 정한 거리만큼 영상에서의 손 부분들을 검출하고 임의의 영상 포인터에 저장하게 된다. 그리고 손 중심으로부터 엄지 손가락의 각도상에 손의 색이 존재하는지 여부를 검사하고 변수에 기록하게 된다. 이때 손이 존재한다면 곧 엄지손가락이 펴져 있다는 것이다. 후에 이변수를 검사해봄으로써 Function의 기능을 수행하게 된다.

아래의 분리된 영역을 컨투어링 해봄으로써 화면상에 조각난 흰색부분이 나타나게 되고 이 조각난 흰색 부분의 개수를 검사함으로서 손가락의 개수를 파악하게 된다.

(a)손의 중심과 손끝 찾기 (b)손가락 개수 파악

그림 6 손가락의 개수를 파악과정

그림 5 (a)에 보이듯이 검지에 있는 손끝 좌표의 평균값을 구해, 실제 마우스를 움직이는 것처럼 자연스러운 이동이 가능해진다. centerx는 손의 중심 x좌표이고 fingerx는 현재 손의 좌표, beforefingerx는 이전영상의 손의 x좌표를 나타낸다. 손 끝 평균값을 계산하지 않고 실험하였을 때 손 끝을 따라가는 마우스 포인터가 많이 흔들리게 된다. 정확한 포인터의 제어가 힘들게 되어 아이콘의 클릭을 하기가 힘들었다. 이 평균값을 통해 실제 마우스의 움직임을 손으로 구현할 수 있게 하였다.

평균값 = fingerx*centerx+beforefingerx*(1-centerx)
평균값 = fingery*centery+beforefingery*(1-centery)

<수식 4> 손 끝 평균값 계산

2.2.2 각도에 따른 손가락 인식

같은 손가락의 개수가 인식 되더라도 검지가 먼저 인식되는지 엄지가 먼저 인식이 되는지에 따라 기능이 달라기지 때문에 손 인식 영상에서 주어진 각도 내에서 손끝을 찾는 연산을 거쳐야한다. 아래의 수식 3을 통해 각도에 따른 다른 기능이 실행되게 된다. dist는 손중심 좌표서부터의 거리이고 radian = 240 * 0.0174329의 값이다. centerx 는 손중심의 x 좌표이다.

x = dist*cos(radian) + centerx
y = dist*sin(radian) + centery

<수식 5> 각도에 따른 손가락 인식

이 수식을 통해 나온 x , y 값을 통해 중심으로 부터의 각도를 구하여 검지손가락이 인식되어 마우스포인터를 움직일지 엄지를 인식하여 Funtion(부가기능)이 수행될지 결정하게 된다.

2.2.3 개수에 따른 마우스 기능 부여

컨투어링과정을 거치면서 손가락의 개수 및 손가락의 존재 여부에 따라 기능을 부여하도록 한다.
손 제스처를 인식 후 기능의 실행은 어느 각도의 손가락이 먼저 인식되는지에 따라 기능이 나누어지게 된다.

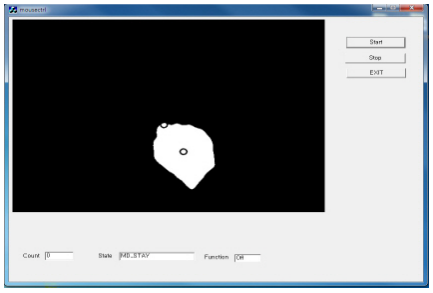
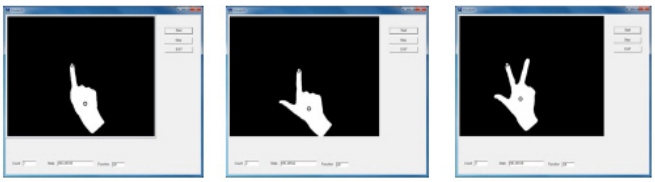


그림 7 대기 상태

기본적으로 검지만 펴져있게 된다면 마우스의 포인터 이동의 기능을 부여한다. 검지와 엄지를 펴게 된 다면 저장된 제스처와 손가락의 정보를 통해 마우스의 드레그 기능을 부여한다. 다시 엄지 손가락을 접게 되면 좌클릭 기능을 부여한다. 중지까지 펴게 되면 제스처와 손가락 셋의 정보를 통해 마우스의 우클릭 기능을 부여한다. 그림 8은 <수식 5>를 거쳐 나온 인식의 각도가 검지일 때 부여 하게 되는 기능들을 나타냈다.



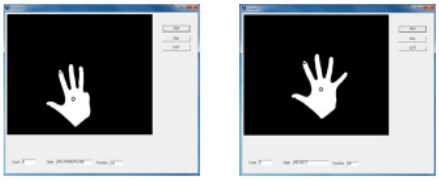


그림 8 각도 계산을 통한 제스처 인식 동작

기능이 없는 마우스도 있지만 일부 마우스에 있는 백스페이스기능과 포워드스페이스의 기능을 부여한다. 각각 손동작이 인식하게 되면 동작한다.

<수식 5>의 각도 계산으로 인해 엄지 손가락이 인식되면 같은 제스처이지만 다른 기능들을 수행할 수 있다.

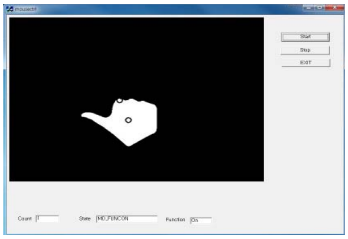


그림 9 추가 기능 손동작

정확한 정보 전달을 위해 손가락의 개수를 사용하였기 때문에 각도에 따른 제스처를 분리하였고 분리에 따라 같은 제스처 이지만 다른 기능이 동작하게 된다.

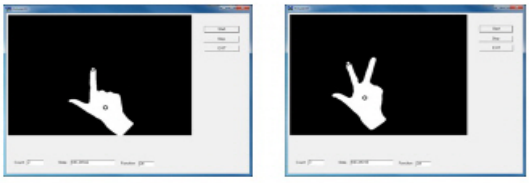


그림 10 추가 기능 손동작에 따른 제스처 인식 동작

2.3 실험 및 토의

V-Gear CCD PC캠을 영상 입력 장치로 사용하였다. ATI RADEON X600 Series를 이용해서 1280*960의 영상을 획득하였다. 마우스 제어를 위한 응용프로그램은 Intel(R) Pentium(R) D CPU 3.40GHz 3.39GHz 상에서 Visual C++6.0으로 구현하였으며, 추적과정에서의 속도는 1초당 7~8프레임정도 처리가 가능하였다.

피부색을 이용하여 손을 분할하기 때문에 사용자의 손과 피부색 배경이 겹쳐 있는 경우 손영역만을 분할해낼 수 없었다.

아래의 실험은 가장 적합한 환경에서의 실험이고, 복잡한 환경과 손색과의 비슷한 환경에서도 실험을 하여 정확도를 조사해보았다. 원 영상에서 YCbCr간으로 변환하여 손의 색을 설정하여 손의 색에 대한 인식이기 때문에 배경에 대해 굉장히 민감하다. 흰 배경에서의 손 인식은 완벽하게 인식한다. 배경에 손색과 비슷한 물건이 없다면 마우스제어는 완벽하게 이루어 졌다. 손색과 비슷한 물건을 놓고 실험 시 손가락 계수 파악에 문제가 있었고, 마우스 포인터의 움직임에도 마음대로 제어할 수 없는 불편함이 있었다.

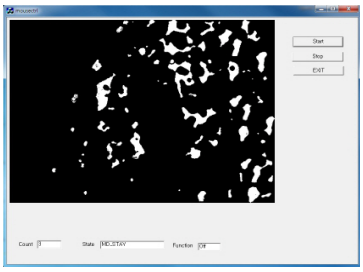


그림 15 손색과 비슷한 환경에서의 실험

검은배경에 대한 마우스제어의 정확도는 흰색배경보다는 정확도가 현저히 떨어지게 되었다.

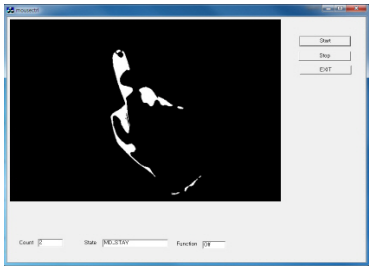


그림 16 검은배경에서의 정확도 실험

복잡한 환경에서의 마우스 제어는 많음 노이즈 때문에 포인터의 이동이 일정하지 않았다. 손가락 계수 파악에도 정확도가 낮아지게 되었다. 이진화의 영상의 임계값을 조절해보았지만, YCbCr 공간으로 변환 후 손의 인식이기 때문에 민감도가 높아 복잡한 환경에 적응 할 수 없었다.

그림 17 복잡한 환경에서의 정확도 실험

손동작을 인식하기 위해서 손가락 개수를 이용하였는데, 이 방법은 팔 전체가 피부색에 의해 분할 된다면 손바닥 중심을 찾아내지 못하는 단점이 있다.

3. 결론

본 논문은 별도의 장비 없이 손동작을 이용하여 사람과 컴퓨터가 효율적으로 상호작용을 할 수 있는 방법을 제시하였고, 마우스 포인터를 제어할 수 있는 시스템을 실제로 구현함으로써 이 방법의 타당성을 증명하였다. 사람의 가장 섬세하고 움직이기 편한 손으로 마우스를 포인터를 제어하여 보다 편리한 마우스제어 크게는 컴퓨터를 제어할 수 있게 하였다. 더 나아가 TV의 제어, 에어컨등 일반 가정에서 사용하는 가전제품등을 손동작만으로 제어할 수 있는 편리성을 제공할 수 있다고 본다. 하지만 향후, 같은 피부색의 손과 배경을 분리할 방법과 팔 전체가 분할 되었을 경우 손목을 분리할 방법을 개발하고 속도의 개선이 해결과제이다. 본 논문에서 제시한 방법은 앞으로 마우스 제어 뿐만 아니라 더 다양한 손동작 인식을 통해 수화등 인간과의 의사소통에도 응용이 가능할 것으로 본다.

참고문헌

[1] J. Lee, T. L, Knuii, "Model-based analysis of hand posture," IEEE Computer Graphics Application, pp.77-86, 1995.

[2] J. Yang, W. Lu, A. Waibel, "Skin-Color Modeling and Adaptation," CMU-CS-97-146, 1997.

[3] Lee Gil-Man, Dae-Sung Moon, Sung-Ok Kim, Min-Hwan Kim, "Hand Gesture Recognition and Tracking under Natural Environment.

[4] Dr. A.W Fitzgibbon , Hand Gesture Recognition Using Computer Vision pp.24-31

[5] S. E. Yang, J. H. Do, H. Y. Jang, K. H. Park, B. J. Nam, Object Position through Hand Pointing Command.

[6] H. J. Kim Program For Hand's Recognition , 2002.

허범근
Heo, Bum Geun

2010년 ~ 현재
수원대학교 일반대학원
정보통신공학과 석사과정
2010년 수원대학교 정보통신공학과(공학사)

관심분야 : 영상 처리, 트렉킹
E-mail : heo0928@suwon.ac.kr

신나라
Shin, Na Ra

2010년 ~ 현재
수원대학교 일반대학원
정보통신공학과 석사과정
2010년 수원대학교 정보통신공학과(공학사)

관심분야 : 영상 처리, 패턴인식
E-mail : nara9n@suwon.ac.kr

홍기천
Hong, Kicheon

1998년 ~ 현재
수원대학교 정보통신공학과 교수
1994년 스티븐스 공과대학 전자공학과(EE) 공학박사
1988년 스티븐스 공과대학 전자공학과(EE) 공학석사
1985년 성균관대학교 전자공학과 (공학사)
1994년 ~ 1998년
삼성 반도체 기흥연구소 및 산호세 연구소 책임연구원
1992년 ~ 1993년
벨통신연구소(Bellcore) 연구원
1988년 ~ 1992년
ATI(Advanced Telecommunications Institute) 연구 조교

관심분야 : MPEG기반 시스템, 임베디드 실시간 시스템, 영상 처리 및 압축, 멀티미디어 프로세서 개발, 패턴인식, 컴퓨터인터페이스
E-mail : kchong@suwon.ac.kr