DOI: 10.3745/KIPSTB.2011.18B.3.109

적외선 카메라를 이용한 에어 인터페이스 시스템(AIS) 연구

김 효 성[†] · 정 현 기[†] · 김 병 규[††]

요 약

본 논문에서는 기계적인 조작 장치 없이 손동작만으로 컴퓨터를 조작할 수 있는 차세대 인터페이스인 에어 인터페이스를 구현하였다. 에어 인터페이스 시스템 구현을 위해 먼저 적외선의 전반사 원리를 이용하였으며, 이후 획득된 적외선 영상에서 손 영역을 분할한다. 매 프레임에서 분할된 손 영역은 이벤트 처리를 위한 손동작 인식부의 입력으로 사용되고, 최종적으로 개별 제어 이벤트에 맵핑된 손동작 인식을 통하여 일반적인 제어를 수행하게 된다. 본 연구에서는 손영역 검출과 추적, 손동작 인식과정을 위해 구현되어진 영상처리 및 인식 기법들이 소개되며, 개발된 에어 인터페이스 시스템은 길거리 광고, 프레젠테이션, 키오스크 등의 그 활용성이 매우 클 것으로 기대된다.

키워드 : 에어 인터페이스, 영상처리, 손 동작 추적, 패턴인식

A Study on Air Interface System (AIS) Using Infrared Ray (IR) Camera

Hyo-Sung Kim[†] · Hyun-Ki Jung[†] · Byung-Gyu Kim[††]

ABSTRACT

In this paper, we introduce non-touch style interface system technology without any touch style controlling mechanism, which is called as "Air-interface". To develop this system, we used the full reflection principle of infrared (IR) light and then user's hand is separated from the background with the obtained image at every frame. The segmented hand region at every frame is used as input data for an hand-motion recognition module, and the hand-motion recognition module performs a suitable control event that has been mapped into the specified hand-motion through verifying the hand-motion. In this paper, we introduce some developed and suggested methods for image processing and hand-motion recognition. The developed air-touch technology will be very useful for advertizement panel, entertainment presentation system, kiosk system and so many applications.

Keywords : Air Interface, Image Processing, Hand Tracking, Pattern Recognition

1. 서 론

현대사회는 컴퓨터 하드웨어와 네트워크 인프라의 급격한 발전으로 인해 사용자가 컴퓨터를 의식하지 않아도 언제 어디서나 네트워크에 접속하여 업무를 처리할 수 있는 유비쿼터스 환경이 도래하게 되었다. 그러나 이러한 급격한 하드웨어의 발전과는 달리 이와 상호작용하는 인터페이스 장치는 기계적인 장치인 마우스와 키보드에서 크게 벗어나지 못하고 있는 실정이다. 진정한 유비쿼터스 시대로 가기 위해서는 인터페이스 장치 또한 유비쿼터스 환경에 걸맞은 사용자 경험(UX : User eXperience)을 제공할 수 있어야 한다.

이러한 흐름을 반영하면서 현재 널리 상용화된 인터페이스 장치로 터치스크린이 있다. 터치스크린 기술은 햅틱, 멀티 터치 기술 등과 결합되면서 보다 풍부한 사용자 경험을 제공하고 있고, 다양한 구현 기법들의 등장으로 인해 낮은 단가로도 대형 LCD 패널 등에 구현 가능하게 되는 등 폭넓은 범용성을 가지고 기존의 인터페이스 장치들을 대체하고

(그림 1) 사용자 인터페이스 장치와 유비쿼터스의 결합

† 준 회 원 : 선문대학교 컴퓨터공학과 석사과정
†† 정 회 원 : 선문대학교 컴퓨터공학과 조교수
논문접수 : 2010년 9월 17일
수 정 일 : 1차 2010년 11월 29일
심사완료 : 2011년 1월 21일

있다[1]. 이러한 많은 장점을 가지고 있는 터치스크린이지만, 조작 대상을 직접 터치해야 하므로 소형의 휴대기기 조작 등에는 적합하나 대형의 TV나 인포메이션 보드 등에는 조작 인터페이스로써 불편한 것이 사실이다. 또한 원격으로 조작이 불가능하기 때문에 자동차에 달린 내비게이션을 조작하기 위해 팔을 길게 뻗어본 경험은 많은 사람들이 해보았을 것이라 생각한다.

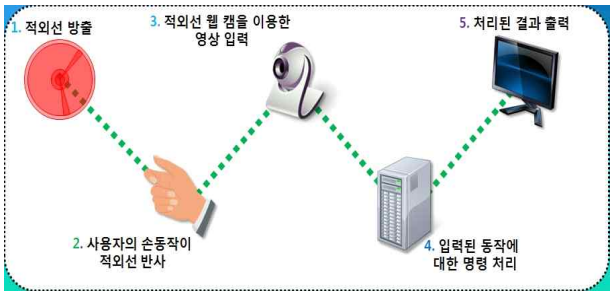
터치를 대신하여 공중에서 손의 움직임만으로 조작하는 에어 인터페이스라면 이러한 터치스크린의 불편한 점들을 해결할 수 있다. 에어 인터페이스를 구현하는 방법에는 기계 착용 방식과 영상처리 방식이 있다. 기계 착용 방식은 장갑 등의 입력 장치를 손에 착용하여 컴퓨터를 조작하는 방식이다. 이미 상용화되어 판매되고 있는 에어 마우스 또한 이러한 방식의 일종이라 할 수 있다. 그러나 가격이 비싸고 조작을 위해 입력장치를 휴대하고 다녀야 한다는 단점이 존재한다.

본 논문에서는 유형의 입력장치 없이 사용자의 손동작만으로 컴퓨터와 상호작용이 가능한 HCI(Human- Computer Interaction) 시스템인 에어 인터페이스 시스템을 구현하였다. 에어 인터페이스 시스템은 사용자 손의 움직임을 추적하여 이를 바탕으로 마우스 포인터를 조작할 수 있고, 사용자의 손동작을 통해 컴퓨터에 명령을 내릴 수 있는 시스템이다.

2. 하드웨어 및 소프트웨어 구성

2.1 하드웨어 구성

에어 인터페이스 시스템 구현을 위해 본 논문에서는 적외선 LED와 적외선 웹캠을 사용하였다. 적외선 LED는 디스플레이 장치 아래에 장착되고, 적외선 웹캠은 디스플레이 장치 위에서 사용자를 바라보는 방향으로 장착된다. 적외선 LED가 적외선을 방출하면 사용자가 취한 손동작이 이를 반사하게 된다. 적외선은 전반사하는 성질이 있으므로 반사된 적외선을 웹캠으로 촬영하면 적외선 LED에 가까운 손 부분은 밝은 영상이 된다. 그러나 손 부분 뒤에 있는 사용자의 몸 부분과 배경은 적외선이 손에 전반사되어 도달하지 못하므로 어두운 영상이 된다. 이러한 적외선의 전반사 원리를 이용하는 것이 본 연구의 핵심이며 전체 하드웨어 구성을 그림으로 나타내면 다음과 같다.



(그림 2) 하드웨어 구성도

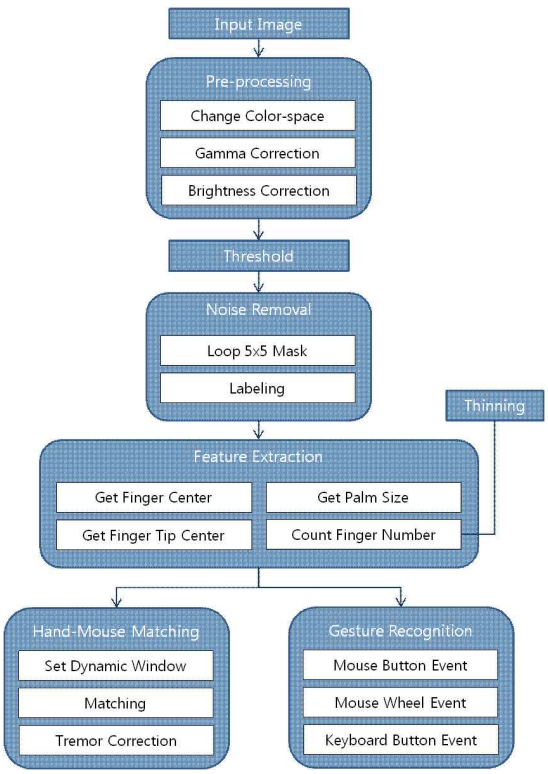
하드웨어 구성에서 핵심이 되는 적외선 웹캠은 일반 웹캠을 개조하여 제작하였다[2]. 웹캠을 분해하여 렌즈 뒷부분을 보면 붉은색 플라스틱으로 된 적외선 차단 필터가 있다(그림 3-a). 이것을 제거하여 적외선이 통과 가능하게 한 뒤(그림 3-b), 제거된 부분에 가시광선 차단 필터를 부착하면 적외선 웹캠이 완성된다(그림 3-c). 가시광선 차단 필터는 일반 사진 필름을 인화하면 가시광선 차단 필터의 역할을 할 수 있으므로 이를 사용하였다.

(그림 3) 적외선 웹캠 개조 과정

2.2 소프트웨어 구성

본 논문에서 구현한 에어 인터페이스 시스템의 소프트웨어 구성은 다음과 같다. 적외선 웹캠으로 촬영한 영상으로부터 노이즈를 제거하고 손 영역을 레이블링하는 과정(3절), 레이블링된 손 영역을 추적하여 마우스 포인터와 매핑하는 과정(4절), 손 영역의 특징을 추출하여 손동작을 인식하는 과정(5절).

전체 소프트웨어 구성과 개발환경을 하단에 (그림 4)와 <표 1>로 나타내었다.



(그림 4) 소프트웨어 구성도

〈표 1〉 개발 환경

OS	Windows 7
Using Tools	Visual Studio .net 2008
	Expression Blend 2.5
Dependency	MFC
	WPF
	OpenCVsharp

3. 손 영역 추출 및 추적

3.1 손 영역 추출을 위한 색 공간 선택

손 영역 추출 및 추적에 있어서 큰 쟁점 중 하나는 손 영역을 추출하는데 필요한 계산량을 최소화시켜 실시간으로 동작 가능하게 하는 것이다. 이를 위해 많은 알고리즘들이 제안되었다[3],[4]. 본 연구에서는 적외선이 손에 반사된 밝기의 정도를 통해 손 영역을 추출하므로, 그 계산량은 실시간으로 추적하기에 충분히 낮은 수준이다. 그러므로 추출 및 추적에 다른 추가적인 고속 알고리즘은 사용하지 않았다.

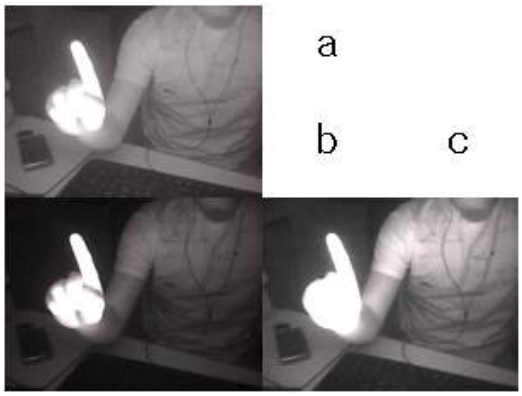
영상을 처리할 때 일반적으로 사용하는 RGB 색상모델은 빨강(Red), 초록(Green), 파랑(Blue) 3색의 조합으로 영상을 표현한다. RGB 색상모델을 통해서는 영상의 밝기정보를 얻어내기 어려우므로 밝기정보를 포함하고 있는 색상모델을 선택할 필요가 있다. YCbCr 색상모델은 밝기(Y) 및 색차(Cb, Cr)로 구성된 색상모델로, 본 연구에서는 RGB 색상모델을 YCbCr 색상모델로 변환한 뒤 0부터 255까지 분포된 Y값 중 그 값이 240 이상인 영역만을 손 영역으로 추출하는 방법을 사용하였다. RGB 색상모델을 YCbCr 색상모델로 변환하는 공식은 다음과 같다.

$$\begin{aligned} Y &= 0.29900R+0.58700G+0.11400B \\ Cb &= -0.16874R-0.33126G+0.50000B \\ Cr &= 0.50000R-0.41869G-0.08131B \end{aligned} \quad \text{수식(1)}$$

3.2 전처리 및 노이즈 제거

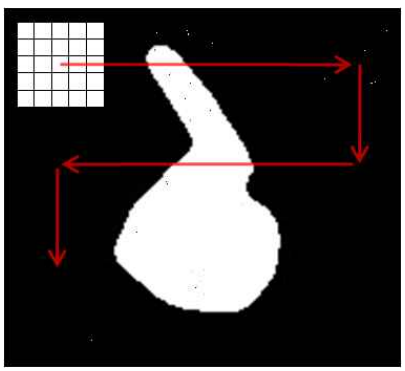
적외선이 반사되는 밝기의 정도를 통해 손 영역을 추출할 때 발생할 수 있는 문제 중 하나는 손의 일부분이 손의 굴곡에 의하여 그늘이 지는 현상이다(그림 5-a). 이러한 경우 완전한 손 영역을 추출하지 못하므로 오작동의 원인이 될 수 있다. 본 연구에서는 획득한 영상을 밝기 값에 따라 이진화하기 전에 전처리를 통해 이러한 현상을 줄이는 과정을 추가했다.

전처리의 첫 번째 단계에서는 입력된 영상(그림 5-a)에 히스토그램 평활화 처리를 하여 명암비를 개선하는 것을 통해 적외선이 반사된 손 영역을 배경보다 선명하게 만든다(그림 5-b). 두 번째 단계에서는 첫 번째 단계를 거친 영상의 히스토그램 값을 일정한 값으로 증가시켜서 영상을 전체적으로 밝게 만든 뒤, 밝기가 200 이상인 인접 픽셀들 중 가장 밝은 값을 취하는 3 x 3 마스크를 씌워서 손 영역 전체를 밝게 만들었다(그림 5-c).



(그림 5) 전처리 과정

전처리 과정이 끝나면 위에서 설명한 바와 같이 Y값 240을 기준으로 손 영역은 흰색으로, 손 영역이 아닌 부분은 검은색으로 이진화한다. (그림 6)은 이진화 과정을 거친 영상에서 손 부분만을 확대한 것이다. 검지 부분을 보면 군데군데 노이즈가 끼어 있는 것을 볼 수가 있다. 본 연구에서는 노이즈를 제거하기 위해 5 × 5 크기의 마스크를 사용하였다. (그림 6)과 같이 5 x 5 마스크로 각 픽셀들을 순환하면서 해당 픽셀 주위의 5 × 5 영역 안에 20 개 이상의 픽셀들이 흰색으로 되어 있다면 해당 픽셀 또한 흰색으로 판정하여 노이즈를 제거하였다[5].



(그림 6) 마스크 기반 노이즈 제거

3.3 레이블링

위 과정을 통해 노이즈가 제거된 손 영역을 추출할 수는 있다. 그러나 위의 방법만을 사용하면 형광등이나 태양빛에도 적외선이 포함되어 있기 때문에, 캠으로 획득한 영상의 배경에 이러한 빛들이 포함될 경우 그 영역 또한 흰색으로 분류가 되는 문제가 발생할 수 있다. 일반적인 조작 환경에서 실제 손이 배경보다 캠에 근접해 있기 때문에, 실제 손 영역이 배경의 형광등이나 태양빛이 반사된 것 보다는 영상에서 더 큰 영역을 차지할 것이다. 이러한 이유로 본 연구에서는 흰색으로 분류된 영역들을 모두 레이블링한 뒤 그 크기를 비교하여 가장 큰 영역만을 남기고 나머지 영역들은 이미지에서 제거하는 알고리즘을 사용하였다. 레이블링이란

인접한 화소들로 구성된 영역을 하나의 객체로 분류하여 번호(Label)를 붙이는 작업이다.

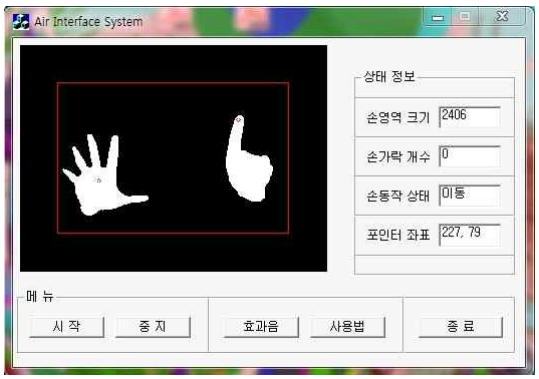
레이블링을 하기 위해서 OpencvSharp의 cvBlob 클래스에 있는 Label 메소드를 사용하였다. 이해를 돕기 위한 샘플 코드는 아래와 같다.

```
CvBlobs blobs = new CvBlobs ();
blobs.Label (imgBinary, imgLabel);
blobs.RenderBlobs (imgLabel, imgSrc, imgDst);
blobs.FilterByArea (minArea, maxArea);
```

CvBlobs 클래스의 객체인 blobs를 생성한 뒤 Label 메소드를 호출하여 이진화된 이미지인 imgBinary로부터 레이블링된 이미지를 생성한다. RenderBlobs 메소드로 레이블링된 이미지를 그려줄 수 있다. FilterByArea 메소드는 레이블링된 영역의 크기를 기준으로 minArea 보다 작거나 maxArea 보다 큰 영역을 제거하는 메소드이다. 본 연구에서는 양손으로 조작하는 듀얼 트래킹 구현까지 고려하고 있기 때문에 minArea는 (레이블링 영역 중 두 번째로 큰 영역의 값 - 1)로 세팅되었다. 이렇게 함으로써 양손을 제외한 나머지 영역들을 제거하는 것이 가능하다. 사용자가 손을 웹캠에 너무 근접해서 조작할 경우, 캠에서 받아들인 이미지로는 손의 움직임이나 제스쳐를 판별하기 어려울 수 있다. 그렇기 때문에 적합한 동작을 보장할 수 있는 최소한의 유효거리를 설정해야 한다. 이를 반영하기 위해서 maxArea는 (이미지 해상도의 넓이 / 4) 로 세팅되었다.

(그림 7)은 에어 인터페이스 시스템을 제어하기 위해서 MFC로 제작된 GUI 프로그램이다. (그림 7-a)의 picture box를 보면 적외선 웹캠으로부터 받은 영상을 전처리한 영상을 볼 수 있다. 영상을 관찰해 보면 양 손 뿐만 아니라 양 팔의 소매 부분, 그리고 창밖의 태양빛 및 안경에 적외선이 반사된 부분이 높은 밝기로 촬영되어진 것을 볼 수 있다. (그림 7-a)의 시작 버튼을 누르면 이진화와 노이즈 제거 및 레이블링이 완료된 영상인 (그림 7-b)가 작동된다. 위에서 설명된 레이블링 알고리즘에 의해 양손을 제외한 나머지 부분들이 모두 제거된 것을 볼 수 있다. 특히 처리하기 어려운 부분 중 하나인 소매부분이 깔끔하게 제거됨을 알 수 있다.

(a)



(b)

(그림 7) 레이블링 결과

(그림 7-b)를 보면 왼손의 중심과 오른손 검지의 중심에 붉은 원이 그려진 것을 볼 수 있다. 그리고 양손 바깥으로는 붉은색 박스가 그려져 있는 것이 보인다. 이는 다음 절에서 논의될 매칭의 중점과 동적 윈도우를 표현한 것이다. 이에 대한 자세한 사항은 다음 절을 참고하기 바란다.

4. 추출된 손 영역과 마우스 포인터의 매칭

4.1 매칭을 위한 중점의 선택

추출된 손 영역의 움직임을 PC 화면의 마우스 포인터(이하 포인터)로 반영하기 위해서는, 먼저 손 영역의 어느 부분을 포인터와 매칭하기 위한 중점으로 할 것인지를 정해야 한다. 검지의 끝과 손바닥의 중심이 그 후보로 선택되었다.

처음에는 (그림 7-b)의 왼손과 같이 손바닥의 중심을 포인터와 매칭하기 위한 중점으로 사용하였는데, 포인터를 이동시키려면 손바닥 전체를 움직여야 하므로 조금만 사용해도 팔이 피로해지는 문제가 있었다. (그림 7-b)의 오른손과 같은 검지의 끝을 매칭의 중점으로 할 경우 사람이 가장 유연하게 다룰 수 있는 손가락인 검지로 포인터를 조작하게 되므로 손바닥 중심에 비해 직관적이면서도 편한 조작이 가능했다.

수학적으로 정확한 손바닥의 중심 및 검지의 끝을 획득하는 것은 상당히 복잡한 알고리즘이 필요하다. 본 연구에서는 실시간 동작이 가능해야 하므로 간단한 방법을 사용하였다. 손바닥으로 분류된(즉, 그림 6에서 흰색으로 분류된) 모든 픽셀들의 x 좌표들의 합을 흰색 픽셀들의 총 수로 나누고 y 좌표에도 마찬가지 방법을 수행하면 손바닥 중심에 근접한 픽셀의 x, y 좌표를 획득할 수 있다[5]. 즉, 흰색 픽셀들의 평균 좌표를 손바닥의 중심으로 사용한다. 검지 끝의 중심은 손 영역의 최상단인 minY와 최하단인 maxY를 구한 뒤, minY로부터 minY+(maxY-minY)/8에 이르는 영역에 손바닥 중심을 구한 것과 마찬가지 방법을 수행하여 획득하였다. 손바닥 중심과 검지 끝의 중심, 그리고 손 영역 픽셀들의 총 수(손의 크기)는 후에 손동작 인식을 위한 특징으로도 사용된다.

4.2 중점과 포인터의 매칭

검지 끝의 움직임을 포인터로 반영하기 위한 방법에는 여러 가지가 있다. 처음으로 생각해 볼 수 있는 것은 마우스를 통해 PC를 조작하는 것과 마찬가지로 검지의 움직임의 가속도 정도를 포인터에 반영하는 방법이다. 그러나 이 방식은 실제 조작하는 손의 위치와 화면상의 포인터 위치가 일치하지 않아 조작에 혼동을 줄 수 있다는 문제가 있다. 이러한 이유로 본 연구에서는 웹캠으로 받아들인 영상의 손의 위치를 그대로 화면에 매핑하는 방법을 사용하였다.

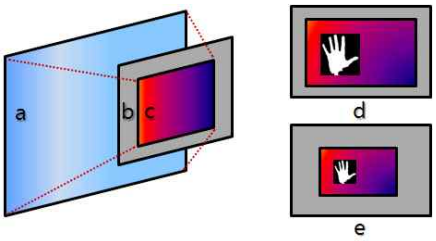
웹캠 영상의 손의 위치를 화면에 매핑할 때 문제가 될 수 있는 것은 조작하는 사용자와 웹캠의 거리에 따라서 조작감이 달라진다는 것이다. 본 연구에서는 모니터와 책상 사이의 거리인 0.5m를 적정조작거리로 잡았는데, 사용자가 적정거리 보다 가까이에서 조작할 경우 웹캠에 잡히는 손 영역의 크기는 커질 것이며 손을 조금만 움직여도 화면상의 포인터는 많은 거리를 이동하게 될 것이다. 이와 반대로 적정거리보다 멀리서 조작할 경우, 같은 거리를 이동하기 위해서 손을 더 많이 움직여야 하며 이는 사용자에게 피로를 유발한다. 사용자가 조작하는 거리가 달라지더라도 안정적인 조작감을 느낄 수 있도록 본 연구에서는 동적 윈도우 구조를 설계하였다.

동적 윈도우 구조에서는 웹캠에 잡힌 사용자의 손의 크기로 사용자와의 거리를 식별한다. 손의 크기는 앞서 손으로 검출되어 흰색으로 나타나는 픽셀들의 총 수로 정의된다. 우리는 10명의 남녀로 구성된 대상에 대한 실험을 통해 320 * 240의 해상도, 그리고 0.5m의 조작거리에서 손의 크기가 2300 ~ 2700 사이의 평균치를 갖는다는 것을 알 수 있었다. 손의 크기가 미리 정의된 평균치보다 클 경우, 사용자가 적정조작거리보다 가까운 거리에서 조작을 하고 있다는 것을 의미하므로 손의 크기를 가중치로 적용하여 동적윈도우의 크기를 확대하였다(그림 8-d). 반대로 영상 내의 손의 크기가 작다면, 사용자가 원거리에서 조작을 하고 있는 것을 의미하므로 동적 윈도우의 크기를 축소시켰다(그림 8-e). 동적 윈도우의 초기 Width와 Height는 웹캠 해상도의 2/3로 세팅되며, 수식(2)에 의해 얻어지는 가중치 W를 곱하여 최종 Width와 Height가 결정된다.

$$W = 1 + \left(\frac{HandSize - 2500}{2500} \times C\right) \quad - \text{수식(2)}$$

C는 보정상수로 거리에 따른 동적 윈도우의 변화 정도를 조절하는 역할을 한다. C의 값은 실험에 참가한 사용자들의 의견을 바탕으로 1/3로 설정하였다.

(그림 8)은 본 연구에서 설계한 동적 윈도우 구조를 도시한 것으로써 실제적으로 PC 화면(그림 8-a)과 매핑되는 것은 웹캠에서 촬영한 영상(그림 8-b)이 아닌 그 안에 논리적으로 존재하는 동적 윈도우(그림 8-c)가 된다. 손 영역의 크기와 동적윈도우의 크기는 비례하기 때문에 거리에 따른 조작감의 차이를 동적윈도우가 보상해주는 구조를 갖는다.



(그림 8) 동적 윈도우 구조

손동작을 인식하기 위해서는 기본적으로 웹캠에서 받아들인 영상 안에 인식하고자 하는 손 영역 전체가 담겨있어야 하는데, 예를 들어 포인터가 화면 가상에 위치했을 때 이를 조작하는 손의 일부가 영상 밖을 벗어나 손동작이 인식이 안 되는 상황이 발생해서는 곤란하다. 이러한 이유로 동적 윈도우(그림 8-c)의 Width와 Height의 최대 크기는 위의 상황에서도 충분히 안정적으로 동작할 수 있도록 웹캠 해상도의 5/6로 설정된다.

4.3 보간법과 손 떨림 보정

본 연구에서 적외선 웹캠으로 받아들이는 영상의 크기는 실시간 동작을 위해 320 * 240의 작은 해상도를 갖는다. 현재 일반적으로 사용되는 PC 모니터의 해상도는 1920 * 1080으로 매우 높기 때문에 웹캠에서 촬영된 검지 의 중심좌표를 PC 모니터에 1:1로 매핑할 경우 해상도의 차이로 인해 움직임의 정확도가 상당히 손실되며 포인터의 움직임이 툭툭 끊기는 느낌을 주게 된다. 구체적인 예로, 해상도가 6배 차이가 날 경우 웹캠 영상의 검지 중심좌표는 정수 값을 가지므로 이 중심좌표가 1 포인트 이동할 때에 모니터상의 포인터는 6 포인트를 이동하게 된다. 이는 (그림 9-a)에서 볼 수 있듯이 6 포인트 보다 작은 세밀한 움직임은 불가능하게 됨을 의미한다.

이는 획득된 검지 중심이 정수 단위로 이동하기 때문에 나타나는 현상이다. 웹캠과 모니터의 해상도 차이를 보간하는 방법으로 이러한 현상을 해결할 수 있으나 계산량이 증가하는 문제가 있다. 검지의 위치를 특징지을 수 있는 값 중에서 실수단위로 이동하는 값이 있다면 보간법을 사용하지 않고도 움직임의 세밀성을 향상시킬 수 있을 것이다. 검지의 중심을 구하는 과정에서 계산하는 평균값은 제산 과정을 거치므로 실수 값을 가지게 된다. 우리는 이 값을 이용하여 중점과 포인터를 1:1로 매핑함으로써 움직임의 세밀성을 향상시킬 수 있었다. 모니터상의 좌표 Xs, Ys는 웹캠상의의 좌표 Xwc, Ywc를 아래 수식에 의해 계산함으로써 얻어진다.

$$x_s = \left(x_{wc} - \frac{W_{wc} - W_{dw}}{2}\right) \times \frac{W_s}{W_{dw}} \quad - \text{수식(3)}$$

$$y_s = \left(y_{wc} - \frac{H_{wc} - H_{dw}}{2}\right) \times \frac{H_s}{H_{dw}}$$

여기서 W는 가로 해상도, H는 세로 해상도를 의미하며 wc는 웹캠, dw는 다이나믹 윈도우, 그리고 s는 시스템(모니터)를 의미한다.

(그림 9)는 그 결과를 비교하기 위해서 에어 인터페이스 시스템의 이니셜인 AIS를 그림판을 통해 그린 동일 궤적의 결과이다. (그림 9-a)는 검지 중심을 매핑한 것이고 그림 9-b는 실수 값을 매핑한 결과이며 (그림 9-c)는 실제 마우스로 그린 것이다. (그림 9-a)에서는 해상도 차이로 인해 (그림 9-c)에 비해 계단현상(Aliasing)이 두드러지게 나타나는 것을 볼 수 있다. 그러나 (그림 9-b)에서는 눈으로 보기에도 (그림 9-c)에 근접한 궤적을 보여주고 있으며 계단현상도 현저하게 줄어든 것을 확인할 수 있다.

a b c

(그림 9) 매핑 기법 개선에 따른 세밀성 비교

위 과정까지 마친 매칭 기법은 미세한 손 떨림에도 포인터가 반응하기 때문에 추가적인 떨림 보정 기법이 필요했다. 떨림 보정 기법을 잘 못 설계할 경우, 사용자의 세밀한 조작마저 떨림으로 간주하여 보정하게 될 수 있으므로 이러한 세밀한 조작은 살리면서 손 떨림은 보정할 수 있게 설계해야 한다. 우리는 이전 다섯 프레임의 x좌표와 y좌표의 offset 값을 기록한 뒤 그 합이 x좌표와 y좌표 모두 7 이하일 경우에 손 떨림으로 판별하여 포인터를 이동시키지 않는 방식으로 손 떨림 보정을 구현하였다.

5. 손동작 인식

5.1 손동작 인식을 위한 손의 특징 추출

손동작 인식을 하기 위해서는 우선 손동작을 특징지을 수 있는 손의 특징들을 추출해야 한다[6]. 본 연구에서는 손가락의 개수와 위치, 손 중심의 이동속도 등을 바탕으로 손동작을 인식하였다. 손가락의 개수와 위치를 효율적으로 판별하기 위해서는 세선화라는 과정을 거쳐야 한다. 세선화란 두께를 가진 영상 객체의 표면을 벗겨내어 그 형태의 특성을 가진 뼈대만을 남기는 작업이다. 세선화된 결과는 원래 영상 객체의 연결성을 유지한 상태로 중심에서 두께 1의 폭을 가지게 된다.

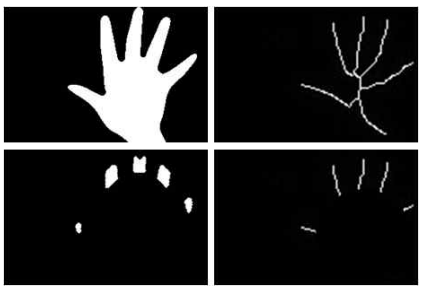
본 연구에서는 널리 알려진 병렬적 처리 기법인 Zhang Suen 세선화 알고리즘을 사용하여 세선화를 수행하였다. Zhang Suen 알고리즘은 좌하 측 방향을 제거하는 함수와 우상 측 방향을 제거하는 함수를 영상 객체의 두께가 1이 될 때까지 반복적으로 수행하여 세선화를 하는 알고리즘으로 흑과 백으로 이진화된 영상에서만 사용가능하다. 해당 알고리즘에 대한 더 자세한 사항은 참고문헌[7]을 참조하기 바란다.

세선화 후 (손 영역의 크기 / 3.14) 에 제곱근을 취한 값을 반지름으로 하여 손 영역을 손바닥의 중심으로부터 원형으로 제거한다. 그러면 손 영역에서 손바닥 부분은 제거되고 손가락 부분만 남게 되는데, 이 남은 영역을 레이블링하여 그 수를 카운트하면 손가락의 개수를 구할 수 있다. 각 손가락의 위치는 손의 중심으로부터 삼각함수를 통해 표현될 수 있다[8]. 예를 들어 엄지의 좌표 x, y는 다음과 같은 식으로 표현될 수 있다.

$$x = r^* \cos(\theta) + Cx \qquad - \text{ 수식(4)}$$
$$y = r^* \sin(\theta) + Cy$$

Cx와 Cy는 손의 중심을 의미하며 r은 엄지까지의 거리(반지름)이며 Θ는 손의 중심을 기준으로 엄지가 위치한 각도이다. 우리는 Θ의 범위를 통해 엄지의 유무를 판별하였다. 엄지의 유무는 클릭 이벤트를 발생시키기 위한 중요한 특징으로 쓰인다.

(그림 10)은 세선화 후 손가락 추출까지 완료한 결과이다. (그림 10)의 좌상단에 있는 그림은 오리지널 이미지이며, 우상단의 그림은 세선화한 이미지, 좌하단의 그림은 세선화없이 손가락 추출을 한 이미지, 우하단의 그림은 세선화 후 손가락 추출을 한 이미지이다. 세선화없이 손가락을 추출한 경우, 각각의 손가락의 두께가 두꺼워 서로 겹쳐질 위험이 있다. 그러나 세선화 후 손가락을 추출한 이미지에서는 그에 비하여 안정적으로 손가락들을 분리하고 있는 것을 볼 수 있다.



(그림 10) 세선화 및 손가락 추출

5.2 추출된 특징을 바탕으로 제스처 구현

5.1절 및 그 이전 절들에서 추출된 손의 특징들을 사용하여 이를 바탕으로 제스처를 구현하였다. <표 3>에는 본 연구에서 구현된 제스처들의 목록 및 구현 방법, 인식률 등이 소개되어 있다. 이전 절들에서 추출된 손의 특징 변수들은 <표 2>에서 다음과 같이 정의된다.

〈표 2〉 손의 특징 변수 정의

변수명	설명
HandSize	손 영역의 크기
FingerCount	손가락의 개수
Center	손의 중심
CenterBefore	바로 이전 프레임의 손의 중심
IndexCenter	검지 중심
Θ	손의 중심을 기준으로 한 손가락의 위치

개발된 시스템은 웹캠으로 부터 초당 30프레임의 이미지 정보가 들어온다. 각각의 프레임에 대해서 콜백 함수를 수행하여 손동작 처리를 반복하는 방법으로 손동작 인식이 구현되었다. <표 3>의 세 번째 열에 그 구체적인 인식과정이 나타나 있다.

인식률은 10명의 실험참가자들을 대상으로 각 제스처를 10회씩 도전하여 성공한 횟수를 평균하여 구하였다. 표에서 제시된 인식률은 실험참가자들의 조작 숙련도에 의해 변할 수 있으므로 객관적인 평가 척도로는 부족한 것이 사실이다. 그러나 구현된 제스처들 간의 조작 난이도를 파악하기에는 좋은 자료가 될 것으로 생각된다.

자주 사용되는 기능인 마우스 이동과 클릭은 진입 조건을 낮게 하여 인식률을 높였다. 드래그 기능은 드래깅 동작 중에 진입 조건을 지속적으로 유지해야 하는 관계로 클릭보다는 인식률이 낮았다. 키보드 방향키 기능과 마우스 휠 기능은 상대적으로 자주 사용되지 않는 동작으로서, 의도치 않은 발생을 줄이기 위해 진입 조건을 어렵게 하였다. 이렇게 함으로써 마우스 이동 중에 방향키 이벤트가 발생하는 등의 오작동을 줄일 수 있었다.

〈표 3〉 제스처 구현 목록 및 인식 과정

제스처	작동 방법	인식 과정	인식률
	<마우스 포인터 이동> 검지를 편 상태에서 움직이면 검지 끝을 따라 포인터가 이동	진입 조건 : FingerCount == 1 처리 내용 : IndexCenter를 4절에 제시된 매핑기법을 통해 매핑 탈출 조건 : FingerCount != 1	100 %
	<마우스 포인터 드래그> 엄지를 펴면 드래그 상태로 전환 다시 오므리면 드래그 상태 종료	진입 조건 : FingerCount == 2 && check(θ >= 130 && θ < 170) 처리 내용 : MOUSE_LEFTDOWN 탈출 조건 : FingerCount != 2	85 %
	<마우스 클릭> 엄지를 폈다가 다시 오므리면 마우스 클릭 이벤트 발생	진입 조건 : FingerCount == 1 && MOUSE_LEFTDOWN 처리 내용 : MOUSE_LEFTUP	93 %
	<마우스 휠 업 (확대)> 손바닥을 펴면 마우스 휠 업 이벤트 발생	진입 조건 : FingerCount == 5 처리 내용 : MOUSE_WHEELUP HandSize에 비례하여 스크롤 속도 증가	82 %
	<마우스 휠 다운 (축소)> 손바닥을 오므리면 마우스 휠 다운 이벤트 발생	진입 조건 : FingerCount == 1 && check(θ >= 30 && θ < 150) 처리 내용 : MOUSE_WHEELDOWN HandSize에 비례하여 스크롤 속도 감소	78 %
	<키보드 위쪽 방향키 (↑)> 손바닥을 위로 빠르게 휘두르면 키보드 위쪽 방향키 이벤트 발생	진입 조건 : abs(Center-CenterBefore) > 8 && def(Center.Y) > def(Center.X) && Center.Y-CenterBefore.Y < 0 처리 내용 : KEYBOARD_UP	72 %
	<키보드 아래쪽 방향키 (↓)> 손바닥을 아래로 빠르게 휘두르면 키보드 아래쪽 방향키 이벤트 발생	진입 조건 : abs(Center-CenterBefore) > 8 && def(Center.Y) > def(Center.X) && Center.Y-CenterBefore.Y > 0 처리 내용 : KEYBOARD_DOWN	67 %
	<키보드 왼쪽 방향키 (←)> 손바닥을 왼쪽으로 빠르게 휘두르면 키보드 왼쪽 방향키 이벤트 발생	진입 조건 : abs(Center-CenterBefore) > 8 && def(Center.Y) < def(Center.X) && Center.X-CenterBefore.X < 0 처리 내용 : KEYBOARD_LEFT	78 %
	<키보드 오른쪽 방향키 (→)> 손바닥을 오른쪽으로 빠르게 휘두르면 키보드 오른쪽 방향키 이벤트 발생	진입 조건 : abs(Center-CenterBefore) > 8 && def(Center.Y) < def(Center.X) && Center.X-CenterBefore.X > 0 처리 내용 : KEYBOARD_RIGHT	74 %

6. 결 론

본 연구에서는 적외선의 전반사 원리를 이용하여 손동작만으로 컴퓨터를 제어 가능한 에어 인터페이스 시스템을 개발하였다. 개발된 시스템은 부가적인 장비를 착용하지 않고 손만으로 원거리에서 조작이 가능하다는 것이 장점이다. 이러한 장점을 바탕으로 길거리 광고나 인포메이션 보드, 게임 및 프레젠테이션 그리고 자동차 내비게이션 등에 응용될 수 있을 것으로 보인다. 개발된 시스템은 0.3m ~ 1.0m 내의 거리에서 초당 30프레임의 처리속도로 지연시간 없이 안정적으로 동작하였으며, 이는 실시간으로 동작하기에 충분한 처리속도이다. (그림 11)은 에어 인터페이스 시스템의 최종 결과로 키보드 방향키 제스처를 통해 미디어 플레이어를 제어하고 있는 모습이다. 디스플레이 위에는 적외선 웹캠이 장착되어 있고 밑에는 적외선 LED가 설치되어 있는 것을 볼 수 있다.

(그림 11) AIS 최종 결과 - 미디어 플레이어 조작

개발된 시스템의 사용거리를 늘리기 위해서는 고속화 기법을 통해 입력 해상도의 사이즈를 늘리거나 고출력의 적외선 LED 소자를 사용하는 등의 방법이 필요할 것이다. 본 연구에서 구현된 제스처들은 웹캠을 통해 입력된 2차원 정보를 바탕으로 구현되었다. 이로 인해 사용자의 손이 똑바로 정면을 향하지 않을 경우에는 인식이 잘 안되는 상황이 발생했다. 이를 해결하기 위해서는 Z-Cam 등의 3차원 카메라를 통해 얻을 수 있는 3차원 정보를 바탕으로 제스처를 구현하는 추가적인 연구가 필요하다.

참 고 문 헌

[1] Sang Hyuck Bae, “Integrating Multi-Touch Function with a Large-Sized LCD", SID 08 DIGEST, 2008.
[2] http://t9t9.com/361
[3] Asanterabi Malima, “A Fast Algorithm for Vision- based Hand Gesture Recognition for Robot Control”, Signal Processing and Communications Applications, pp.1-4, 2006.
[4] Michael Donoser and Horst Bischof, “Real Time Appearance Based Hand Tracking", IEEE International Conference on Pattern Recognition, pp.1-4, 2008.
[5] “손동작 인식을 이용한 Finger pad, Finger pad Using Hand Gesture Recognition" 한양대학교 학사학위 논문
[6] Heung-Il Suk, “Real-Time Hand Pose Tracking and Finger Action Recognition Based on 3D Hand Modeling” 정보과학회 논문지 : 소프트웨어 및 응용 제 35 권 제 12 호
[7] Y. Y. Zhang and P. S. P. Wang, “A Modified Parallel Thinning Algorithm”, IEEE International Conference on Pattern Recognition, pp.1023-1025, 1988.
[8] Taejin Ha, Woontack Woo “Video see-through HMD 기반 증강현실을 위한 손 인터페이스” GIST U-VR Lab.

김 효 성
e-mail : st04nx@sunmoon.ac.kr
2010년 선문대학교 컴퓨터공학과(학사)
2010년~현 재 선문대학교 컴퓨터공학과 석사과정
관심분야 : 영상 및 비디오 신호처리, 차세대 비디오 압축/처리 및 HCI 기술

정 현 기
e-mail : hyunki99@sunmoon.ac.kr
2010년 선문대학교 컴퓨터공학과(학사)
2010년~현 재 선문대학교 컴퓨터공학과 석사과정
관심분야 : H.264/AVC 및 스케일러블 비디오 압축, 비디오 신호처리, 차세대 비디오 압축/처리 기술

김 병 규
e-mail : bg.kim@ieee.org
1995년 국립 부산대학교 전기공학과(학사)
1998년 한국과학기술원(KAIST) 전자전산학과(석사)
2004년 한국과학기술원(KAIST) 전자전산학과(박사)
2004년~2008년 한국전자통신연구원(ETRI) 선임연구원
2009년~현 재 선문대학교 컴퓨터공학과 조교수, IEEE/ACM/IEICE 정회원, 한국정보처리학회, 한국방송공학회, 한국멀티미디어학회 정회원
관심분야 : 영상 및 비디오 신호처리, 비디오 압축 이론 및 고속화 기법, 패턴 인식 및 지능적 영상 시스템, 무선 멀티미디어 센서 (WMSN) 기술 등