논문 2008-45SP-6-16

배경영상 기반 다자유도 포인팅 디바이스의 설계

(Design of a Background Image Based Multi-Degree-of-Freedom Pointing Device)

장 석 윤*, 고 재 원**

(Sukyoon Jang and Jaewon Kho)

요 약

대화형 멀티미디어 기기들이 급속하게 보급되면서 리모컨이나 컴퓨터 마우스와 같은 기존의 사용자 인터페이스 방식으로는 제약이 많아 사용이 불편한 경우가 많은데, 이러한 문제를 해결하기 위한 방안으로 비전기반의 포인팅 디바이스를 소개한다. 포인팅 디바이스에 카메라를 장착하여 카메라에 입력되는 배경 동영상을 분석하면 카메라의 움직임을 추정할 수 있는데 이를 근거로 커서를 움직이는 방법을 이용한다. 동작이 실시간으로 이루어 질 수 있도록 288×208 화소의 저해상도의 카메라를 이용하고 여기에 입력된 스크린의 모서리 영상을 찾아 국부적으로 옵티컬 플로우(Optical flow) 방법으로 트래킹하고 스크린의 크기를 계산하면 스크린과 디바이스간 거리와 디바이스의 움직임 경로를 알아낼 수 있다. 이러한 방법으로 만들어진 포인팅 디바이스는 기존의 비전기반 다자유도 입력장치들에 비하여 어두운 환경에서 유리하고 구현이 간단하며 기존의 마우스와 같이 한 손에 쥐고 사용방법을 학습하지 않고도 직관적으로 사용가능한 장점을 갖는다.

Abstract

As interactive multimedia have come into wide use, user interfaces such as remote controllers or classical computer mice have several limitations that cause inconvenience. We propose a vision-based pointing device to resolve this problem. We analyzed the moving image from the camera which is embedded in the pointing device and estimate the movement of the device. The pose of the cursor can be determined from this result. To process in the real time, we used the low resolution of 288×208 pixel camera and corner points of the screen were tracked using local optical flow method. The distance from the screen and the device was calculated from the size of screen in the image. The proposed device has simple configurations, low cost, easy use, and intuitive handheld operation like traditional mice. Moreover it shows reliable performance even in the dark condition.

Keywords : Optical flow, Pointing device, Background image, Tracking.

Ⅰ. 서 론

표준 컴퓨터 마우스는 여러 가지 제약 때문에 발전 속도가 눈부신 대화형 멀티미디어 환경에서는 인터페이스 작업 시 불편함을 야기할 때가 많다. 일반 마우스는 2차원 평면 위에서만 동작되므로 게임이나 프레젠테이션 작업에 매우 불리하고 멀티미디어를 조작하기 위한 리모컨을 대체하기도 어렵다. 이러한 이유로 공중공간에서 사용이 가능한 새로운 원거리 포인팅 디바이스(remote pointing device)들이 계속해서 개발되고 있다. 자이로 마우스[1], 조이스틱 기반의 마우스, 트랙볼 마우스, 틸트 감지도구[2], 큐빅 마우스[3], 소나펜[4], SOAP[5] 등을 예로 들 수 있다. 이러한 포인팅 디바이스를 이용하면 스크린 상의 마우스 포인터를 원거리의 공중 공간에서도 제어할 수 있다.

자이로마우스는 최근에 상용화된 기술이지만[6~7], 사용된 자이로센서가 느린 움직임이나, 각도의 변화가 없는 평행한 움직임은 감지할 수 없기 때문에, CAD나 그래픽 디자인과 같은 정교한 움직임을 요하는 작업에는

* 학생회원, 연세대학교 생체공학협동과정
(Graduate Program in Biomedical Engineering, Yonsei University)
** 정회원, 유한대학 컴퓨터제어과
(Detp. of Computer Control, Yuhan College)
접수일자: 2008년3월3일, 수정완료일:2008년10월23일

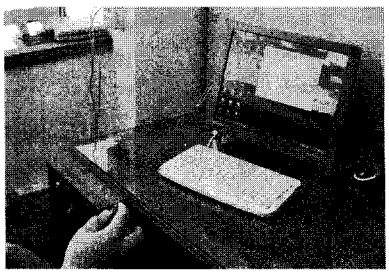


그림 1. 제안된 포인팅 디바이스를 이용한 인터페이스
Fig. 1. User interface using a suggested pointing device.

적용하기 어려운 단점이 있다. 참고문헌 [8]은 자이로 마우스의 단점들을 잘 보여준다. 조이스틱과 트랙볼 마우스는 널리 사용이 되는 편이지만 정밀한 작업을 하기에는 불편하다[9]. 최근에 개발되었거나 개발 중인 다른 여러 포인팅 디바이스들도 움직임에 있어 제한된 자유도[2], 한정된 응용[3], 높은 에러율[4], 느린속도[5] 등 중대한 문제점들을 보이고 있다. 비전기반의 포인팅 디바이스와 관련된 사항들은 Ⅱ장에서 별도로 다루기로 한다. 본 논문에서는 기존의 연구들[1~22]의 단점을 보완하여 좀 더 실용성 있는 포인팅 디바이스를 개발하기 위해 무선 카메라를 이용하여 모니터 스크린의 코너점을 검출하고 트래킹한 후 커서를 제어하는 방법(그림 1)을 이용하였다. 본 논문에서 사용된 기법들은 상용화 가능성을 고려하여 저가의 부품들을 이용하면서도 실시간으로 정확한 포인팅 동작이 가능하도록 고안하였다.

Ⅱ. 관련 연구

비전기반의 지시장치는 구동영상의 이용방식에 따라 전경영상(Foreground image) 기반방식[11~17]과 배경영상(Background image) 기반방식[18~22]으로 구분할 수 있다. 전자는 주로 고정된 카메라 앞에서 사람의 움직임이나 특정 대상물의 움직임 등을 추적하고 그 결과를 이용하여 마우스 포인터를 이동시키는 반면에 후자는 기존의 광마우스처럼 손에 쥘 수 있는 디바이스 안에 카메라를 내장시켜 카메라에 입력되는 배경영상을 분석하여 디바이스 자체의 움직임을 알아내는 방식이다. DeMenthon 등은 세 점을 이용한 근사적 원근법을 이용하여 자세를 추정하는 방법을 제시하였는데[10], 이 방법을 이용하여 고정된 카메라 앞에서 세 삼각점에 LED를 부착한 형태의 포인팅 디바이스를 구현한 바 있다[11]. 비슷한 방법들이 현재까지 응용되어오고 있으나, 대부분의 연구에서는 특정한 형태의 대상체를 트래킹[12~14]하는 대신, 얼굴, 손, 몸짓을 포함한 사람의 움직임을 트래킹 하는 경우[15~17]가 많다. 전경영상기반 포인팅 디바이스들은 고정된 카메라의 한정된 시야 내에서만 사용이 가능하고 제스쳐를 해석하기위한 별도의 규정[16]이 필요하기 때문에 제품화되지 못하거나 장애인용[15] 등 용도가 국한되고 있다.

저가의 소형 카메라 모듈이 등장함에 따라 카메라를 내장한 배경영상 기반의 포인팅 디바이스가 활발하게 연구되기 시작하였다. Hinckley 등은 카메라 기반의 다자유도 포인팅 디바이스를 고안하였으나[18], 이차원상의 격자 패턴이 있는 지지면이 있어야 동작이 가능하기 때문에 공중공간에서는 동작이 불가능하다. Cantzler 등도 카메라를 내장하여 손에 쥐고 동작시키는 포인팅 디바이스를 개발하였는데[19], 모니터를 향해 포인팅 디바이스를 움직이는 방법으로 커서를 제어할 수 있는 방식이었지만, 스크린 상에서 사각 코너 점들을 찾은 후 가장 큰 사각 코너 점의 중심위치를 구하여 스크린 상에서 카메라가 향하고 있는 위치를 계산하기 때문에 몇 가지 중요한 문제점을 야기할 수 있다. 우선 기준으로 선정한 사각이 변하면 커서위치도 변하므로 화면의 내용이 자주 바뀌는 컴퓨팅 환경에서는 커서의 위치가 매우 불안정해진다. 또한 모니터와 매우 가까운 위치에서 디바이스를 사용할 경우 카메라의 시야에 스크린이 꽉 들어차게 되므로 디바이스를 조금만 움직여도 사각점이 카메라의 시야를 벗어나게 되어 디바이스를 움직일 수 있는 범위를 심하게 제한받게 된다.

보다 최근에는 휴대폰이나 PDA와 같은 모바일기기에 장착된 카메라를 이용하여 기존의 기기에 포인팅 기능을 구현하는 연구가 진행되었다[20~22]. Vartiainen 등은 모바일 카메라 기기를 이용하여 포인팅 시스템을 구현하였는데[20], 스크린 위치를 알아내기 위한 특정한 형태의 비주얼 코드를 이용하고 있다. Jiang 등이 개발한 Direct Pointer[21]는 카메라로부터의 영상프레임의 중심에 항상 커서가 위치하도록 커서의 움직임을 제어함으로써 마치 이 장치의 위치변화에 따라 커서가 움직이는 것처럼 피드백 제어하는 방식인데, 카메라가 커서의 위치를 알아내지 못하면 동작이 불가능한 단점이 있다. 따라서 이 장치는 커서의 형태가 확연히 구분되는 대형 스크린을 이용할 경우에 한해 유용한 방식으로 볼 수 있다. 개발중인 포인팅 디바이스들 중에는 카메라와 카메라 화면에 잡히는 대상물과의 거리를 고려하지 않고 마우스 포인터를 제어하는 것들도 있는데[22], 이런 경우에는 마우스 포인터의 이동속도를 적절하게 정하기가

어려워 카메라와 대상물과의 거리가 가까운 경우에는 포인터의 이동속도가 비정상적으로 빨라질 수 있다. 배경영상을 기반으로 하는 포인팅 디바이스들 중에서 상용화된 모델로는 닌텐도사의 Wii-remote[25]가 있다. Wii-remote에 탑재된 적외선 카메라는 TV의 상단에 설치된 10개의 적외선 LED영상을 검출하고 이를 이용하여 커서의 위치를 계산하는 방식이다. 이 시스템은 손에 쥐는 포인팅 디바이스 외에 별도의 센서 시스템을 설치해야 한다는 단점이 생긴다.

본 논문에서 제안된 방법은 현재 일반적으로 널리 사용되는 여러 종류의 컴퓨터 디스플레이에서 사용가능하며 디스플레이상의 비주얼 코드나 별도의 센서 없이 기존에 발표된 연구보다는 스크린의 위치를 좀 더 가깝거나 멀리 두고서도 사용할 수 있도록 했고 포인팅 디바이스의 움직임 영역도 확장하였다.

Ⅲ. 구현방법

TV 리모컨을 이용할 때처럼 공중 공간의 포인팅 디바이스는 스크린을 향해 사용되는 경향이 있으며 스크린은 주위에 비해 명도가 높고 정해진 크기를 갖고 있기 때문에 디바이스에 내장된 하나의 카메라만 가지고도 모니터까지의 거리와 위치변화를 알아낼 수 있다. 비디오 스트림을 입력받아 연속적으로 3차원적 모니터 위치를 갱신시키기 위해 시퀀스 상의 모든 이미지마다 기준위치를 계산할 경우에는 여러 가지 문제가 생기게 된다. 특히, 실제 모니터 주변에서 모니터와 비슷한 물체로 인하여 잘못 인식되는 경우, 마우스 포인터의 위치가 급작스럽게 엉뚱한 곳으로 이동할 수 있다. 따라서 모니터를 검출한 후에는 잘 알려진 Lucas-Kanade optical flow 방법[23]을 이용하여 모니터 영상의 에지 점 4군데를 트래킹 하였고, 이러한 추적점이 화면에서 사라지는 경우에만 재검출하도록 하였다. 이렇게 하면 물체의 인식과정에서 오류가 발생하더라도 트래킹 결과에 의해서만 포인터를 움직이므로 포인터가 엉뚱한 곳으로 이동하지 않는 장점이 있다.

1. 시스템 개요

포인팅 동작을 구현하기위해 구성한 시스템은 PC, 카메라가 내장된 무선 포인팅 디바이스, 모니터, 클릭 이벤트 리시버 (OMW4UL-R), 이미지 수신장치 (RX2400S) 등으로 구성된다. 그림 2는 알고리즘의 구현에 이용된 시스템의 구성을 보여준다. 포인팅 디바이스

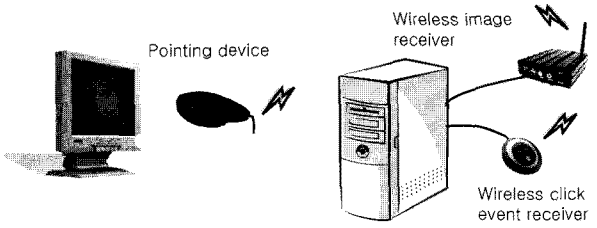


그림 2. 테스트 시스템의 구성
Fig. 2. Setting of the experimental system.

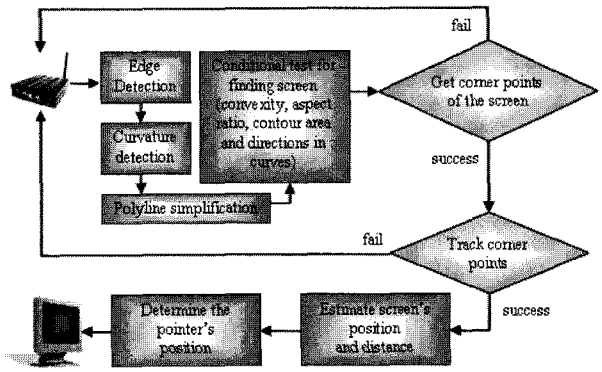


그림 3. 드라이버 프로그램의 구조
Fig. 3. Flowchart of the device driver.

는 두 개의 무선 수신기를 통해 PC에 연결된다.

마우스 포인터를 제어하기위한 비전알고리즘은 PC에서 구현되기 때문에 개발된 드라이버 프로그램을 설치해야하며, 드라이버 프로그램의 구조는 그림 3과 같다. 입력된 이미지에서 스크린의 모서리에서 4개의 특징 점들을 추출하고 이것의 위치변화를 마우스 포인터를 움직이기 위한 근거로 사용하였다. 만일 4개의 특징 점들이 유효적절하지 않다면 조건에 맞는 점들이 추출될 때까지 검출작업을 반복한다. 트래킹 프로세스는 인식과정과 분리되어 스크린 인식오류로 인한 마우스포인터의 잘못된 움직임을 방지할 수 있다.

2. 스크린의 코너 점 검출

모니터 스크린의 코너 점 검출방법은 컨투어 기반의 특징점, 컨벡스, 에지의 연속성, 에지 접합부의 각도, 컨투어 면적, 종횡비 등을 근거로 검출한다. 컨투어 기반의 방법은 텍스쳐에 근거한 방법에 비해 색과 조명의 변화가 검출과정에 영향을 덜 미치기 때문에 영상의 공간적 구조를 좀 더 효율적으로 잡아낼 수 있다. 먼저 스크린 후보 영역을 검출하기 위해 Canny edge 알고리즘[24]에 기반을 둔 에지 검출기를 입력이미지의 휘도(luminance) 성분에 적용하였다. 사물의 닫힌 경계영역을 검출하기 위한 Canny edge 검출기의 출력은 그림 4(b)와 같다.

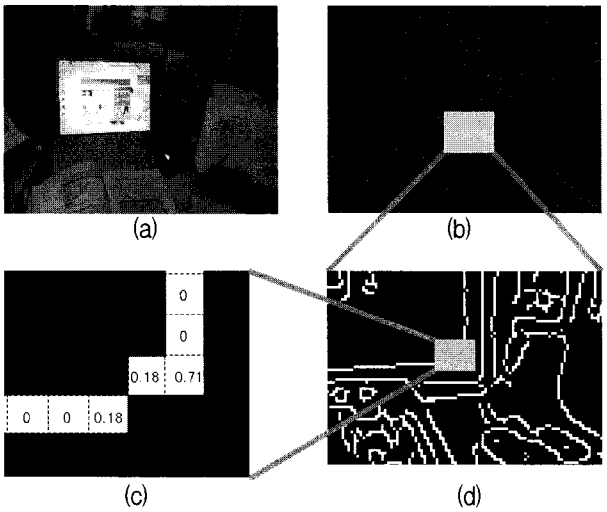


그림 4. (a) 소스이미지 (b) canny detector를 이용해 얻은 컨투어 이미지 (c) 스크린 코너부근의 커버쳐 값 (d) 컨투어 이미지의 부분 확대

Fig. 4. (a) Source image (b) Contours from the canny detector (c) Curvature values around the screen corner (d) Partial enlargement view.

(1) 커버쳐의 계산

코너점 및 단순화된 컨투어 선을 얻기 위해 커버쳐를 계산하였다. u점에서의 커버쳐는 다음과 같은 커버쳐 함수를 이용하여 계산할 수 있다.

$$k(u)=\frac{\dot{x}(u)\ddot{y}(u)-\ddot{x}(u)\dot{y}(u)}{\left[\dot{x}(u)^2+\dot{y}(u)^2\right]^{3/2}} \tag{1}$$

위 수식은 컨투어의 기울기함수 $(\tan^{-1}[\dot{y}(u)/\dot{x}(u)])$를 미분하여 구할 수 있는데, 에지 방향으로의 기울기를 의미한다. 소프트웨어로 구현하기 위하여 (1)식은 (2)식과 같이 근사화될 수 있다.

$$k(u)\approx\frac{[x(u+1)-x(u-1)]]y(u+1)-2y(u)+y(u-1)]}{\{[x(u+1)-x(u-1)]^2+[y(u+1)-y(u-1)]^2\}^{3/2}} \\ \frac{-[x(u+1)-2x(u)+x(u-1)][y(u+1)-y(u-1)]}{\{[x(u+1)-x(u-1)]^2+[y(u+1)-y(u-1)]^2\}^{3/2}} \tag{2}$$

그림 4(c)는 스크린의 한 코너부근의 각 픽셀에서 계산된 커버쳐 값의 예를 보여준다.

(2) 코너점의 선정

코너점의 가능성이 있는 점들을 가려내기 위해 작은 윈도우를 만들어 그 내부의 연속적인 버택스(vertex)들 중 극대 값을 갖는 픽셀만 골라내는 작업을 컨투어 맵을 따라 윈도우를 이동시키면서 행하였다. 그림 5(a)는 이렇게 하여 선택된 버택스들을 작은 사각점 형태로 나타낸 것이다. 직사각내부에서 인접한 변과의 각은 90°에 가깝다는 사실을 이용하면 식(3)을 이용하여 사각을 구성하는 버택스들을 가려낼 수 있다.

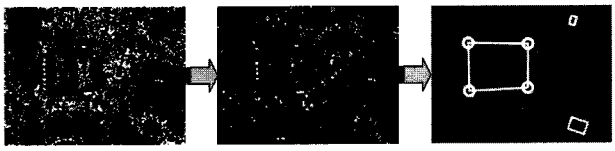


그림 5. (a) 지역적 최대 커버쳐 (b) 식(3)을 이용하여 가려낸 후보 버택스 (c) 컨투어면적, 컨투어내 버택스 개수를 이용하여 가려낸 3개의 4각 버택스쌍과 종횡비를 고려하여 최종 선택한 버택스쌍

Fig. 5. (a) Local maximum curvatures (b) Candidate vertices selected using the equation (3) (c) Three pairs of rectangular vertices selected using conditions of the contour area and the number of vertices within a contour and a finally chosen pair using the aspect ration.

$$\left|\frac{dx_1 \cdot dx_2+dy_1 \cdot dy_2}{\sqrt{\left(dx_1^2+dy_1^2\right)\cdot\left(dx_2^2+dy_2^2\right)}}\right|\approx 0 \tag{3}$$

여기서, $dx_1=x(u-1)-x(u),\ dy_1=y(u-1)-y(u)$
$dx_2=x(u+1)-x(u),\ dy_2=y(u+1)-y(u)$

의 관계가 있다.

식 (3)을 제약조건으로 하여 얻어낸 점들을 그림 5(b)에 나타내었다. 다음 단계로 각 컨투어 내에서 버택스들의 개수가 4개인 것만 남기고 컨투어의 면적이 너무 작은 것들을 제거하면 그림5(c)처럼 샘플 영상의 경우 3개의 사각을 구성하는 에지 점들을 얻을 수 있다. 최종적으로 스크린의 종횡비(4:3 또는 16:9)를 고려한다면 대부분의 경우 스크린상의 코너점만 얻어낼 수 있다. 그림 5(c)에서는 이렇게 찾아진 네 점을 흰 원으로 표기하였다. 만일 최종단계에서도 2개 이상의 스크린 에지 후보들이 남아있다면 화면의 중심에 보다 가까운 것을 선택하도록 하였는데, 이는 다음 단계에서 화면의 중심에서 트래킹을 시작하는 것이 유리하기 때문이다.

3. 모션 트래킹

입력 영상에서 일단 스크린의 코너 점들이 찾아지면 이어지는 영상부터는 트래킹 프로세스를 수행한다. 스크린에 대한 포인팅 디바이스의 상대적인 움직임은 코너 점에 대한 트래킹 결과에 따라 추정된다. 코너 점들은 수직하는 두 방향에 대해 이미지의 공간적 기울기가 높기 때문에 트래킹이 매우 용이하다. 실험에 사용한 시스템에서 새로운 프레임은 초당 약 30번 갱신되는데,

현재의 모니터 이미지는 다음 프레임의 명암패턴과 매우 유사하기 때문에 옵티컬 플로우(optical flow)알고리즘에 의한 트래킹이 가능하다.

물체를 트래킹하기 위한 옵티컬 플로우 알고리즘들이 다수 존재하지만, 본 논문에서는 Lucas-Kanade의 방법[25]을 이용하고 있는데, 이는 다른 방법들에 비해 간결하기 때문에 비교적 정확하고 빠른 특징이 있다. 기본적으로 Lucas-Kanade 알고리즘은 식(4)의 잘 알려진 모션 제약 식에 기반을 둔다.

$$\frac{dI}{dt}=(\nabla I)^T v+\frac{\partial I}{\partial t}=0 \tag{4}$$

여기서 I는 (x,y,t)에서 명도(intensity) 이며, $v=(\frac{dx}{dt},\frac{dy}{dt})^T$는 이미지 속도이다. 그러나, 이러한 관계만으로는 옵티컬 플로우의 국소에 한정된 해석에 기인하는 렌즈구경문제(aperture problem)로 인하여 이동위치추정이 어렵다. 이 때문에 Lucas와 Kanade는 국소 윈도우내의 옵티컬 플로우 v를 상수로 가정하고 그 윈도우 내에서 식(5)와 같이 정의되는 에러 함수 E의 가중 제곱 합(weighted sum-of-squares)을 최소화하는 방법을 제안하였다.

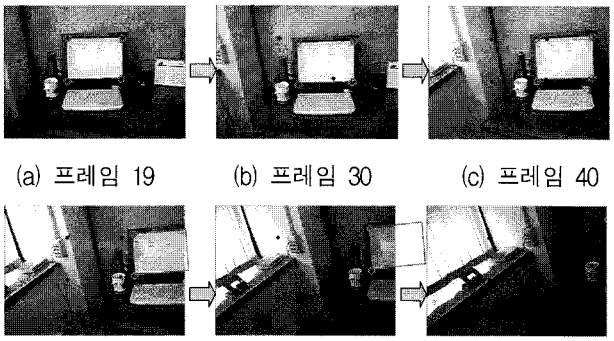
$$E=\sum_w\left\{(\nabla I\cdot v)+\frac{\partial I}{\partial t}\right\}^2 \tag{5}$$

여기서 w는 가중함수이다. 본 논문에서는 윈도우의 중심에 가중치를 부여하기 위하여 w를 가우시안 함수로 하였다. 옵티컬 플로우 v는 최소제곱법으로 구할 수 있다.

$$v=\left[\sum_w(\nabla I)^T(\nabla I)\right]^{-1}\left(\sum_w-(\nabla I)^T\frac{\partial I}{\partial t}\right) \tag{6}$$

$$\begin{bmatrix}\sum_w\left(\frac{\partial I}{\partial x}\right)^2 & \sum_w\frac{\partial I}{\partial x}\frac{\partial I}{\partial y}\\ \sum_w\frac{\partial I}{\partial x}\frac{\partial I}{\partial y} & \sum_w\left(\frac{\partial I}{\partial y}\right)^2\end{bmatrix}^{-1}\begin{bmatrix}-\sum_w\frac{\partial I}{\partial x}\frac{\partial I}{\partial t}\\ -\sum_w\frac{\partial I}{\partial y}\frac{\partial I}{\partial t}\end{bmatrix}$$

그림 6은 Lucas-Kanade 포인트 추적 알고리즘에 의한 결과의 한 예를 보이고 있다. 프레임 19에서 스크린의 네 코너점이 검출되면 프레임 92와 같이 4개의 코너점이 모두 화면에서 사라질 때까지 계속해서 트래킹한다. 프레임 50과 70처럼 트래킹중인 점들의 일부가 시야에서 벗어났을 경우에는 이전에 기억하고 있던 각각의 코너점간의 거리정보를 이용하여 포인팅 디바이스의 위치변화를 추정한다. 즉 그림 6(d-e)에 표시한 바와 같



(a) 프레임 19 (b) 프레임 30 (c) 프레임 40

(d) 프레임 50 (e) 프레임 70 (f) 프레임 92

그림 6. 코너 점의 트래킹 - 각 프레임에서 붉은 점은 스크린 코너 점 검출 알고리즘에 의하여 찾아낸 네 코너 점의 위치이며 녹색원은 이동한 코너 점을 트래킹한 것이다

Fig. 6. Tracking of the corner points - In each frame, the red dots indicate the position of the four corners from the screen corner detection process and the green circles plot the tracked points.

은 가상의 선을 그려 스크린의 위치를 추측하게 된다. 이러한 방법으로 포인팅 디바이스의 인식 가능한 팬 또는 틸트 각을 넓힐 수 있으며 동작하는 위치가 모니터와 가까운 경우 그 효과는 크다. 프레임 92와 같이 트래킹하던 점이 모두 사라지면 스크린의 코너 점 검출과정을 다시 시작한다.

4. 거리측정

마우스 포인터의 움직임은 포인팅 디바이스의 각도 또는 위치변화에 대응된다. 디바이스의 상대적 위치변화를 찾기 위해서는 모니터 스크린의 크기와 동영상에서 이동되는 스크린의 중심점을 알아야한다. 단일 카메라를 이용하기 때문에 모니터까지의 상대적 거리를 알기위해서 원근법을 이용한다. 그림7에 영상면, 렌즈, 사물간의 기하학적 관계를 나타내었다. 디바이스 전면에서 Z만큼 떨어진 곳에 위치한 스크린의 코너 점 $P_n(X_n,$

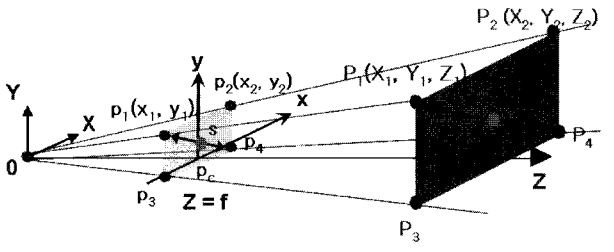


그림 7. 영상좌표 p(x,y)와 공간좌표 P(X,Y,Z)

Fig. 7. Image coordinates of p(x,y) and world coordinates of P(X,Y,Z).

Y_n, Z_n)은 2차원 이미지상의 $p_n(x_n, y_n)$에 투영되는데, 이점들은 식(7)에 의하여 주어진다.

$$x_n = \frac{f}{Z_n} X_n,\ y_n = \frac{f}{Z_n} Y_n \quad (n = 1 \sim 4) \tag{7}$$

여기서, f는 일반적으로 mm 단위로 표시되는 초점거리(focal length)이다. 포인팅 디바이스가 모니터의 전면과 제한된 시야각 이내에서 사용된다고 가정하면 각각의 버택스(Z_1-Z_4)와 포인팅 디바이스 사이의 거리들은 계산을 간략하게 하기 위하여 모두 동일하게 취급할 수 있다. 사전에 미리 알 수 있는 실제 모니터 스크린의 크기 S(15, 17, 19인치 등)와 이미지상의 버택스로부터 다음 식을 이용하여 거리 Z를 구할 수 있다.

$$s = \sqrt{(x_4 - x_1)^2 + (y_4 - y_1)^2} \tag{8}$$

$$Z = \frac{f}{s} S \tag{9}$$

또한 식(10)을 이용하여 스크린의 중심점을 구해 마우스 포인터를 움직일 때 기준위치로 사용하였다.

$$P_c = \left(\frac{X_1 + X_4}{2}, \frac{Y_1 + Y_4}{2}, Z \right) \text{ or } \left(\frac{X_2 + X_3}{2}, \frac{Y_2 + Y_3}{2}, Z \right) \tag{10}$$

그러면, 마우스 포인터가 이동할 위치는 P_c의 좌표에 의하여 결정될 수 있음을 알 수 있다. 예를 들어 1024×768 픽셀의 해상도를 갖는 15인치 LCD 모니터를 사용하고 포인팅 디바이스내의 카메라는 288×208 픽셀의 해상도로 동영상을 캡쳐하며 55mm의 초점거리를 가지고 있다고 하면, 식 7~9와 모니터의 물리적 픽셀 크기를 이용하여, s≒5.3cm, Z≒39.5cm를 계산할 수 있다. 여기서 모니터의 물리적 픽셀크기란 모니터에서의 단위 길이 당 픽셀의 수를 말하며 스크린의 가로축(또는 세로축)상의 픽셀수를 디스플레이 되고 있는 스크린의 물리적인 가로축 폭(또는 세로축 높이)으로 나누어 구할 수 있다.

5. 마우스 포인터 제어

버택스를 추적하는 동안 초점거리와 포인팅 디바이스-스크린 간의 거리가 변하지 않는다고 가정하면 Z와 f를 각각 Z′와 f′로 근사화시킬 수 있으므로 식(11)과 (12)의 C를 상수로 취급하여 마우스 포인터의 가속 파라미터로 사용할 수 있다. 만일 파라미터가 2C로 설정된다면, 다음 입력 프레임에서 마우스 포인터의 위치는 (x,y)에서 (x+2C, y+2C)로 이동하게 된다.

$$v_x = \frac{\Delta X}{\Delta t} = \frac{\frac{fx}{Z} - \frac{f'x'}{Z}}{\Delta t} \approx C(x - x'),\ C \approx \frac{f}{\Delta t \cdot Z} \tag{11}$$

$$v_y = \frac{\Delta Y}{\Delta t} = \frac{\frac{fy}{Z} - \frac{f'y'}{Z}}{\Delta t} \approx C(y - y') \tag{12}$$

마우스 포인터의 이동 범위를 확장하기 위해 클러치 메커니즘(clutch mechanism)을 고려하였다. 내재적인 클러치 매커니즘을 사용하는 광 또는 볼 마우스의 경우 마우스가 패드위에서 사용될 때 패드의 한쪽 끝에서 더 이상 이동이 어려우면 이를 들어 올려 다시 적당한 위치로 이동하는 식으로 클러칭이 이루어진다. 이러한 기존의 마우스와는 달리 공중에서 사용하는 부류의 마우스들[1~4, 6~18, 20~23]은 기계적으로 내재된 클러치 메커니즘이 없기 때문에 버튼이나 움직임 추적을 차단하기 위한 터치센서 등을 채용하고 있다. 본 논문에서 제시하고 있는 시스템에서는 별도의 버튼을 달아 이를 누른 상태에서만 트래킹이 이루어지도록 하고 여기서 손을 떼면 클러칭이 되도록 하였다.

Ⅳ. 실 험

구현된 프로그램의 동작 상태를 테스트하기 위하여 첫 번째로 사용자 테스트를 통하여 최적화된 마우스 포인터의 가속 파라미터 값을 찾고 간단한 포인팅 동작 속도 기준으로 기존의 광마우스와 효율성을 비교해보았으며, 둘째로 다양한 환경조건 하에서의 안정성을 확인해보았다. 실험은 윈도우즈 XP가 탑재된 PC를 이용하였으며 디스플레이 장치로 1024×768 픽셀을 갖는 15인치 LCD모니터, 100인치 프로젝션, 17인치 CRT 모니터 등이 이용되었다. 구동과 테스트에 필요한 프로그램은 모두 Visual C++를 이용하여 작성하였다.

1. 가속파라미터의 설정

그림 8에 보이는 테스트 프로그램은 마우스를 움직이며 클릭하는데 소요되는 시간을 측정하는 방법으로 포인팅 동작의 효율성을 간접적으로 확인하기 위한 것이다. 시작 버튼을 클릭하면 8개의 임의의 숫자들이 에디트 박스에 나타나고, 사용자는 일치되는 숫자버튼을 누르도록 하였다. 클릭 속도는 평균 클릭 간 소요시간

을 측정하였고, 분당 클릭속도의 단위로 환산하여 표시하였다. 마우스와 PC의 사용이 능숙한 열 명의 대학원생이 실험에 참가하여 개발된 디바이스를 가지고 네 가지의 가속 파라미터환경에서 각각 실험하였으며 기존의 광마우스를 가지고 측정한 것과 결과를 비교하였다. 가속 파라미터는 C에서 4C(식.11 참조)까지 설정하였고 각 조건 당 5회 시도하여 평균치를 기록하였다.

그림 9에 가속 파라미터 값에 따른 클릭 속도의 차이와 광마우스를 사용했을 때의 결과를 비교하였다. 개발된 디바이스의 전체적인 평균 클릭속도는 기존의 마우스에서 보다 조금 느린 편이지만, 그림 9에 보이는 바와 같이 가속 파라미터를 최적화하면 기존의 마우스보다 더 효율적으로 사용할 수도 있음을 보여준다.

개발된 디바이스를 이용한 실험에서 전체적인 클릭 속도가 느려지는 가장 큰 이유는 내장된 카메라가 사용하는 시야의 유효범위를 넘는 경우가 자주 발생하거나 클러치 메커니즘의 사용이 미숙하기 때문인 경우가 많다. 이러한 결점은 초점거리가 훨씬 더 짧은 카메라를 사용하거나 마우스 포인터 이동에 관련된 파라미터들을 최적화함으로써 해결할 수 있다.

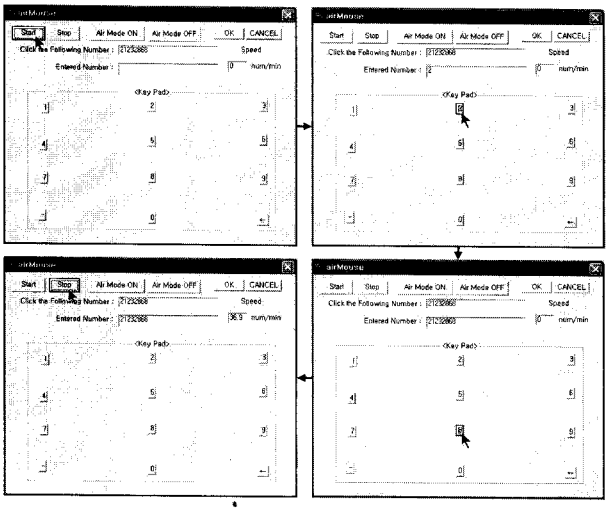


그림 8. 클릭속도를 계산하기위한 실험환경
Fig. 8. Experimental platform to calculate the click speed.

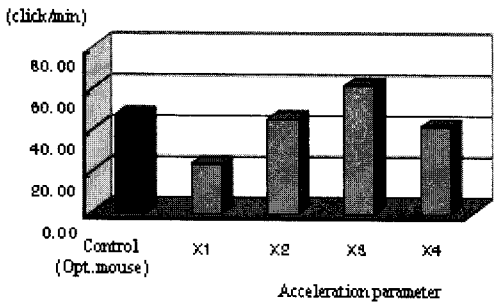


그림 9. 클릭속도 실험결과
Fig. 9. Test results of the click speed.

2. 환경적 신뢰성 테스트

여러 컴퓨터 사용 환경에서 동작의 신뢰성을 확인하기 위하여 조명과 모니터 사용 조건에 대한 동작성을 측정해 보았다. 시스템의 정확성과 신뢰성은 인식과 트래킹 과정의 에러발생률과 관련이 깊기 때문에 입력된 동영상으로부터 모니터의 인식오류나 추정오류의 비율을 가지고 평가하였다. 실제 디바이스에 장착된 카메라에서 얻어지는 영상을 실험 데이터로 활용하였다. 각 영상의 해상도는 288×208 픽셀이고, 평균 프레임 샘플링 주파수는 30Hz이다.

첫 번째 실험은 15인치 LCD 모니터를 가지고 20와트 형광등이 천정에 매달린 상태의 인공조명 조건과 모든 불빛이 차단된 어두운 조건으로 나누어 실시하였다. 두 번째와 세 번째 실험은 각각 100인치 프로젝션과 17인치 CRT 모니터를 사용하여 첫 번째 실험과 같은 절차로 실시하였다. 그림10은 실험에 사용된 동영상중 각

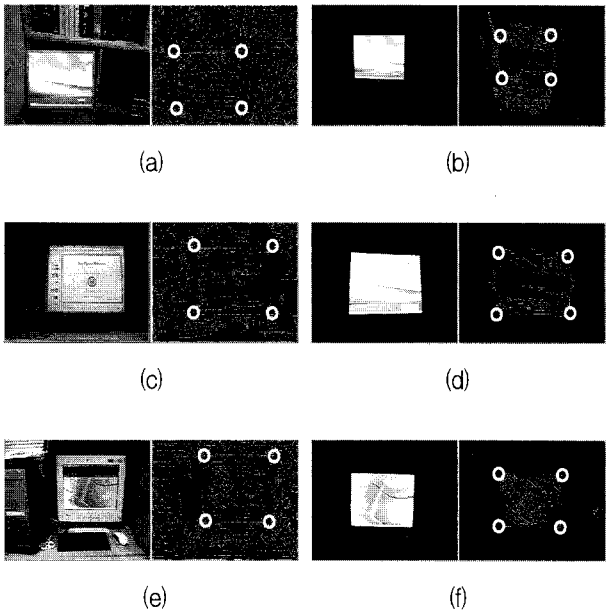


그림 10. 조명이 있는 환경 조건과 없는 환경 조건하에서 각각 포인팅 디바이스의 동작을 테스트하기 위한 스크린 코너의 검출 예: (a)-(b) LCD 모니터 (c)-(d) 프로젝션 (e)-(f) CRT 모니터
Fig. 10. Examples showing the screen corner detection results to test the performance of the pointing device under environmental conditions with and without illumination: (a)-(b) LCD monitor (c)-(d) Projection (e)-(f) CRT monitor.

표 1. 검출 및 트래킹 에러율
Table 1. The error rates of detection and tracking performance.

screen	LCD Monitor		Projection		CRT Monitor	
illumination	natural	dark	natural	dark	natural	dark
Total frame	324	350	505	475	470	469
Error frame	36	0	6	4	8	0
Error Rate	11.11%	0%	1.19%	0.84%	1.70%	0%

조건마다 하나의 샘플 영상만 골라 나타낸 것이다.

입력 프레임 중에서 모니터나 스크린을 포함하지 않는 것은 에러율의 계산에서 제외하였고, 에러율은 전체 프레임의 숫자에 대한 검출오류나 트래킹오류가 발생한 프레임의 비율로 계산하였다. 표 1에 보이는 결과로부터 컴퓨터 디스플레이 종류와는 상관없이 어두운 조건에서는 에러율이 작은 것을 알 수 있다. 컴퓨터 디스플레이 자체가 스스로 발광하기 때문에 조명조건에 유리하다.

V. 토의 및 향후과제

제안된 알고리즘은 모니터의 디스플레이 일부가 다른 물체에 의하여 가려진 경우 기준점을 검출하지 못한다는 점과 에지검출과 옵티컬 플로우를 이용하기 때문에 CPU에 부담을 주고 있다는 점은 개선이 필요하다. 또한 더 낮은 해상도에서 더 짧은 초점거리를 갖는 렌즈를 이용하는 방법도 필요하다. 왜냐하면 초점거리가 짧은 렌즈를 이용하면 포인팅 디바이스의 이동범위를 확장시킬 수 있고 낮은 해상도를 이용하면 CPU의 부담을 줄일 수 있기 때문이다. 그러나, 이미지의 해상도가 낮을수록 검출과 트래킹의 정확성이 떨어지므로 성능을 개선하는 것이 쉬운 작업은 아니다. 만일 제안된 알고리즘을 마우스 내부의 하드웨어에 구현한다면 응용 범위가 매우 넓을 것으로 보인다. 응용이 유리한 분야중 하나는 GUI방식의 멀티미디어를 위한 리모컨이며 현재 이러한 응용 연구도 진행 중이다.

VI. 결 론

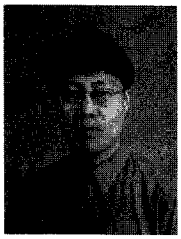
본 논문에서는 스크린 코너 점의 검출과 트래킹의 방법으로 공간에서 사용되는 비전기반의 포인팅 디바이스를 구동하기 위한 알고리즘을 소개하였다. 제안된 방법은 C++를 이용하여 구현하였고 테스트를 위한 프로토타입 시스템은 원거리에서 손에 쥐고 동작을 시킬 수 있도록 무선형식으로 제작하여 여러 환경적 조건에서 동작을 테스트하였다. 개발된 시스템은 특정한 형태의 마커나 비주얼 코드를 사용하지 않고도 기존에 개발된 비전기반의 시스템에 비해 안정된 포인팅 동작이 가능하다. 스크린의 모양에 대한 의존성, 연산 량에 대한 부담 등 많은 해결과제가 남아있지만 실제 상용 가능성에 대한 큰 기대를 얻을 수 있었다. 예를 들면, 어두운 환경에서도 안정된 동작, 쉬운 작동 법, 다양한 컴퓨팅 환경으로의 접목 성 등 기존에 개발되었거나 연구 중인 다자유도 마우스에 비하여 우수한 특징들이 많다.

참 고 문 헌

[1] G. M. Eom, K. S. Kim, C. S. Kim, J. Lee, S. C. Chung, B. S. Lee, H. Higa, N. Furuse, R. Futami, and T. Watanabe , "Gyro-Mouse for the disabled: click and position control of the mouse cursor," IJCAS, vol. 5, no. 2, pp. 147-154 , 2007.
[2] J.ReKimoto, "Tilting operations for small screen interfaces," UIST'96, pp.167-168.
[3] B. Fröhlich and J. Plate, "The Cubic Mouse. A new device for three-dimensional input," Proc. ACM CHI 2000, pp. 526-531, April 2000.
[4] M. S. Sohn and G. H. Lee, "SonarPen: An ultrasonic pointing device for an interactive TV," IEEE Transactions on Consumer Electronics, vol. 50, no. 2, pp. 413-419, May 2004.
[5] P. Baudisch, M. Sinclair, and A. Wilson, "Soap: a pointing device that works in mid-air," UIST'06, Oct.15 - 18, Montreux Switzerland, 2006.
[6] http://www.gyration.com
[7] http://www.sico.co.kr/Z3/index.asp
[8] I. S. MacKenzie and S. Jusoh, "An evaluation of two input devices for remote pointing," Proceedings of the eighth IFIP international conference on EHCI 2001, pp 235-249, Heidelberg, Germany, Springer-Verlag, 2001.
[9] I. S. MacKenzie, T. Kauppinen, and M. Silfverherg, "Accuracy measures for evaluating computer pointing devices," CHI 2001, pp.9-16, 2001.
[10] D. DeMenthon and L.S. Davis, "Exact and approximate solutions of the perspective-three-point problem," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.14(11), pp.1100-1105, Nov.1992.
[11] D. DeMenthon, "From artificial vision to

synthetic reality: System for interaction with a computer using video image analysis (in French)," PhD thesis, University Joseph Fourier, Grenoble, France, Oct.4. 1993.
[12] M. Munich and P. Perona, "Visual input for pen-based computers," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.24, no.3, pp.313-328, March 2002.
[13] Z. Zhang, Y. Wu, Y. Shan, and S. Shafer, "Visual panel: Virtual mouse, keyboard and 3D controller with an ordinary piece of paper," PUI 2001, Orlando, Florida, Nov. 15-16. 2001.
[14] P. Nesi and A. D. Bimbo, "A vision-based 3-D mouse," Int. J. Human-Computer Studies, vol.44, no.1, pp.73 - 91, 1996.
[15] M. Betke, J. Gips, and P Fleming, "The camera mouse: Visual tracking of body features to provide computer access for people with severe disabilities," IEEE Trans. Neural. Syst. Rehabil. Eng. v10(1), Mar. 2002.
[16] Y.Wu and T.S.Huang, "Hand modeling analysis, and recognition for vision-based human computer interaction," IEEE Signal Processing Magazine, pp.51-60, May 2001.
[17] 한승일, 황용현, 이병국, 이준재, "스테레오 비전을 이용한 포인팅 디바이스에 관한 연구," KSIAM IT series Vol.10, No.2, pp.67-80, 2006.
[18] K. Hinckley, M. Sinclair, E. Hanson, R. Szeliski, and M. Conway, "The VideoMouse: A camera-based multi-degree-of-freedom input device," ACM UIST'99, pp. 103-112, 1999.
[19] H. Cantzler and C. Hoile, "A novel form of a pointing device," in Proc. Vision, Video and Graphics, pp.57 - 62, 2003.
[20] P. Vartiainen, S. Chande, and K. Rämö, "Mobile visual interaction: enhancing local communication and collaboration with visual interactions," MUM'06, Stanford, California, Dec. 2006.
[21] H. Jiang , E. Ofek, N. Moraveji, and Y. Shi, "Direct Pointer: direct manipulation for large-display interaction using handheld cameras," CHI 2006, Canada, April 2006.
[22] J. Wang, S. Zhai, and J. Canny, "Camera phone based motion sensing: Interaction, techniques, applications and performance study," UIST'06, Montreux, Switzerland, pp.101-110, Oct. 2006.
[23] B.Lucas and T.Kanade, "An iterative image registration technique with an application to stereo vision," Proceedings of DARPA Image Understanding Workshop, pp.121-130, 1981.
[24] J. Canny, "A computational approach to edge detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.8, no.6, pp. 679-698, Nov. 1986.
[25] http://wii.nintendo.com

저 자 소 개



장 석 윤(학생회원)
1990년 충남대학교 전자공학과 학사.
2000년 동대학 의공학과 석사.
1995년~1997년 삼성전자 AV사업부 연구원.
2008년 현재 연세대학교 생체공학 협동과정 박사과정,
2008년 유한대학 컴퓨터제어과 강의전담교수
<주관심분야 : 컴퓨터비전, 로봇, 의용전자공학>

고 재 원(정회원)
1981년 연세대학교 전기공학과 학사.
1983년 동대학 전기공학과 석사.
1992년 동대학 전기공학과 박사.
1986년~1992년 삼성전자 자동화연구소 선임연구원.
1993년~1995년 고등기술연구원 책임연구원
2008년 현재 유한대학 컴퓨터제어과 부교수
<주관심분야 : 로봇제어, 지능제어, 컴퓨터비전>