개선된 터치점 검출과 제스쳐 인식에 의한 DI 멀티터치 디스플레이 구현

Implementation of a DI Multi-Touch Display Using an Improved Touch-Points Detection and Gesture Recognition

이 우 범*
Woo-Beom Lee

요 약

멀티터치 관련 연구는 전반사 장애 현상(FTIR: Frustrated Total Internal Reflection)의 원리를 기반으로 기존 방법을 이용하여 단지 구현하는 것이 대부분이다. 또한 멀티 터치점(Blob-Points) 검출이나 사용자 제스쳐 인식에 있어서 성능 향상을 위한 소프트웨어적 해법에 관한 연구는 드문 실정이다. 따라서 본 논문에서는 확산 투광(DI: Diffused Illumination) 방식을 기반으로 개선된 터치점 검출과 사용자 제스쳐 인식에 의한 멀티터치 테이블-탑 디스플레이를 구현한다. 제안된 방법은 실행 중인 어플리케이션 내의 객체들을 위한 동시 변형 멀티터치 명령을 지원하며, 제안한 사전 테스팅(Pre-Testing) 방법에 의해서 멀티 터치점 검출 과정에서 시스템 지연 시간의 감소가 가능하다. 구현된 멀티터치 테이블-탑 디스플레이 장치는 OSC(Open Sound Control) 프로토콜을 기반으로 하는 TUIO(Tangible User Interface Object) 환경에서 Flash AS3 어플리케이션을 제작하여 시뮬레이션 한 결과 최대 37% 시스템 지연 시간의 감소와 멀티터치 제스쳐 인식에서 성공적인 결과를 보였다.

Abstract

Most of the research in the multi-touch area is based on the FTIR(Frustrated Total Internal Reflection), which is just implemented by using the previous approach. Moreover, there are not the software solutions to improve a performance in the multi touch-blobs detection or the user gesture recognition. Therefore, we implement a multi-touch table-top display that is based on the DI(Diffused Illumination), the improved touch-points detection and user gesture recognition. The proposed method supports a simultaneous transformation multi-touch command for objects in the running application. Also, the system latency time is reduced by the proposed pre-testing method in the multi touch-blobs detection processing. Implemented device is simulated by programming the Flash AS3 application in the TUIO(Tangible User Interface Object) environment that is based on the OSC(Open Sound Control) protocol. As a result, Our system shows the 37% system latency reduction, and is successful in the multi-touch gestures recognition.

Keywords : Multi-Touch Display, Diffused Illumination, Touch-points extraction, Gesture recognition, Multi-objects transformation, System Latency Time Reduction

I. 서 론

멀티터치(Multi-touch)란 컴퓨터 사용자가 터치스크린 또는 터치 패드를 입력으로 하는 환경에서 마우스 포인트와 같은 하나의 터치 포인트만을 인식하는 것이 아니라, 사용자의 여러개의 손가락 접촉 포인트를 동시에 인식함으로서 사용자 제스쳐에 의한 다중 명령(Multi-command)의 수행이 가능한 인터페이스 기술을 의미한다.

2006년 TED(Technology Entertainment Design) 컨퍼런스에서 제프 한(Jefferson Y. Han)에 의해서 발표되었던 전반사 장애(FTIR) 현상을 이용한 멀티터치 기술은 최근 기계-사용자 인터페이스 분야에서 컴퓨팅 환경 접근성의 유용성과 편리성 때문에 가장 주목받고 있는 인터페이스 기술 분야 중에 하나이다[1].

현재 멀치터치 기술은 애플사의 아이폰(iPhone), 아이팟 터치(iPod Touch), 맥북(MacBook) 등으로 대중에게 소개되었을 뿐만 아니라, 차세대 윈도우 시스템으로 개발된

*상지대학교
투고 일자 : 2009. 9. 10 수정완료일자 : 2010. 1. 26
게재확정일자 : 2010. 1. 29
* 이 논문은 2009년도 상지대학교 교내 연구비 지원에 의한 것임.

MS사의 Windows 7 OS에도 사용 친화적 인터페이스를 위하여 탑재되어 있다.

그러나 기존 FTIR 기술을 이용한 멀티터치 테이블-탑 디스플레이 장치의 구현은 적지 않은 제약과 단점들이 존재한다. 또한 대부분의 연구가 기존에 구현된 방법의 재현에 의해서 하드웨어 장치를 설계하여 완성하고 있다. 그리고 완성된 디스플레이 장치의 구동에 있어서도 대상 어플리케이션에 최적화된 장치 소프트웨어의 구현은 고려하지 않고 Touclib나 Gesturelib[2]와 같은 범용 라이브러리를 적용하는 것에 그치고 있다.

따라서 본 논문에서는 기존의 대부분의 멀치터치 방식인 FTIR 방식의 수정 및 보완이 가능한 확산 투광 (DI: Diffused Illumination) 기술을 이용한 멀티터치 테이블-탑 디스플레이 장치를 구현한다. 또한 소프트웨어적 측면으로 멀티터치 인터페이스의 성능 향상을 위한 핵심 기술인 터치점(Blob-points) 검출이나 사용자 제스쳐 인식에 있어서 기존의 범용 라이브러리를 분석하여 새로운 방법의 적용이 가능하도록 터치점 검출을 위한 단계별 적용 과정을 명시하고, 적용하고자 할 어플리케이션에 최적화된 제스쳐 인식을 위한 방법을 제안한다.

제안된 방법은 터치 프레임 간의 사전 테스팅 연산을 이용하여 터치점 검출에 있어서 불필요한 영상처리 작업의 제거에 의한 시스템 지연 시간의 감소가 가능하다. 또한 제스쳐 인식 알고리즘도 멀티터치 디스플레이 장치 상의 변형이 요구되는 객체를 단위로 수행되기 때문에 기존의 방법에서는 불가능한 다중 객체의 동시 변형이 가능하다.

그리고 구현된 멀치터치 테이블-탑 디스플레이 장치는 OSC(Open Sound Control) 프로토콜을 기반으로 하는 TUIO(Tangible User Interface Object) 환경에서 Flash AS3 어플리케이션을 제작하여 시뮬레이션하여 시스템 지연 시간을 측정하고 멀티 터치 제스쳐 인식의 성공적인 결과를 보인다.

II. 멀티터치 테이블-탑 디스플레이 구조

디스플레이 장치로부터 사용자의 손가락 터치 제스쳐를 획득하기 위한 방법으로는 크게 전반사 장애 현상(FTIR)을 이용한 방식과 빛의 확산 투광(DI)에 의한 방식이 있다[3].

임계각 이상의 입사각을 가진 빛이 매질 경계면에서 모두 매질 내부로 반사되는 전반사 장애 현상을 이용하는 FTIR 방식은 사용자 손가락과 디스플레이 장치와의 접촉에 의해서 발생하는 장애 현상에 의해서 분산되어 나오는 적외선 LED 빛을 대역통과 필터(Band-pass filter)를 부착한 적외선 카메라를 사용하여 접촉점을 검출한다.

그리고 DI 방식은 디스플레이 장치의 표면 아래로부터 적외선 투광기(IR Illuminator)에 의해서 투광되는 빛으로부터 사용자 터치에 의해서 반사되는 적외선 빛을 적외선 카메라로 추적하여 터치점을 검출한다. 일반적으로 DI 방식은 FTIR 방식에 비해서 구현 비용 및 하드웨어 제작에 있어서는 우수하지만, 터치 강도나 터치 추적에 있어서는 약간 미흡하다. 따라서 본 논문에서는 구현에 있어서 용이한 DI 방식을 채택하여 터치점 검출에 있어서의 문제점을 영상처리 작업을 통해서 보완한다.

본 논문에서 사용자 멀티터치 입력을 위한 멀티터치 테이블-탑 디스플레이 장치의 하드웨어 구조는 그림 1과 같다. 멀티터치 테이블은 120x90x80cm(LxWxH) 크기의 중질섬유판(MDF: Medium Density Fiberboard)으로 제작하였다. 그림에서 빛의 확산에 의한 적외선 탐지용 카메라는 일반 웹캠으로는 터치점 검출에 충분한 적외선 검출이 미흡하기 때문에 Philips SPC900NC 웹캠을 4.3mm CCTV 적외선 렌즈와 Negatives 필름을 이용하여 개조하였다.

멀치터치 패널은 120x90x1cm 강화유리 위에 트랜싱지를 디퓨져(Diffuser)로 붙여서 프로젝터의 화면이 투영되게 하였다. 확산 투광을 위해서는 적외선 투광기 두 개를 사용하고 인식률을 높이기 위해서 멀티터치 테이블-탑 내부 사방 벽면에 알루미늄 호일을 부착하였다. 그리고 화면 투영을 위해서 1024x768 화소 이상의 DLP(Digital Light Processing) 방식의 프로젝터를 사용한다.

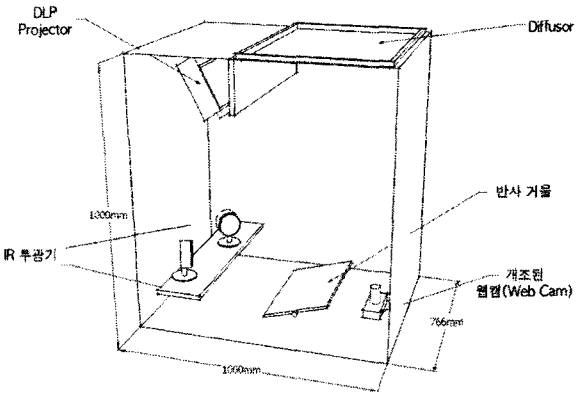


그림 1. 멀티터치 테이블-탑 하드웨어 구조
Fig. 1 The Proposed Multi-Touch Table-Top Hardware

III. 멀티 터치점(Touch-points) 검출

멀티터치 테이블-탑 디스플레이 하드웨어 장치인 터치 패널에서 발생하는 멀티터치 제스쳐를 인식하기 위해서는 카메라로부터 전송되는 터치 패널 영상으로부터 사용자가 멀티터치 패널과 접촉한 터치흔적(Touch-Blobs)을 추출하고 그 흔적으로부터 터치점을 검출해야만 한다.

그림 2에 나타낸 바와 같이 터치점 검출을 위해서는 전송된 카메라 영상에 대해서 그레이 변환, 사전 테스팅, 윤곽 강조, 배율 조정, 임계 처리, 라벨링 등의 영상 처리가 단계적으로 요구된다.

그레이 변환 단계에서는 카메라로부터 전송된 RGB 24bit 프레임을 아래 식 (1)에 의해서 8bit 그레이(Gray) 영상으로 변환하였다

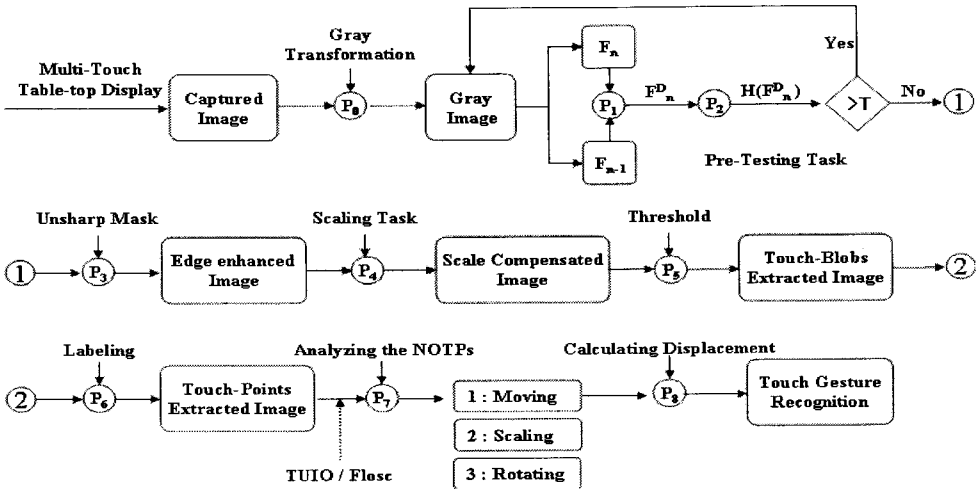


그림 2. 본 논문에서 제안하는 멀티-터치 처리 과정
Fig. 2 Multi-Touch Processing Tasks in Our approach

$$Y=0.299*R+0.587*G+0.114*B \quad (1)$$

그레이 레벨로 변환된 영상은 터치흔적 추출을 위한 사전 테스팅 작업을 수행한다. 본 논문에서는 사용자 터치 제스쳐 인식을 위해서 디스플레이 상에서 발생하는 터치 행위 프레임들로부터 처음과 끝 프레임에서만 터치흔적을 추출한다. 이것은 하나의 터치 제스쳐 인식에 있어서 터치점의 시작과 끝 사이에 존재하는 터치 흔적은 불필요한 정보로 정의하여 터치 이동 중의 프레임에서 발생하는 터치 흔적에 대해서는 부가적이 영상처리 과정을 생략함으로서 시스템의 반응 시간을 향상 시킬 수 있다.

따라서 터치흔적 추출을 위해서 NxM 화소 크기의 터치 흔적 프레임들로부터 아래의 식 (2)에 의해서 현재 프레임 F_n과 이전 프레임 F_{n-1}과의 차영상 F_n^D를 구하고, 차영상의 모든 화소에 대한 차의 합 히스토그램 $H(F^D)$를 계산한다.

$$H(F_n^D)=\sum|F_n-F_{n-1}|=\sum_{x=0}^{N-1}\sum_{y=0}^{M-1}|f_n(x,y)-f_{n-1}(x,y)| \quad (2)$$

여기서 n은 프레임 인덱스를 의미하고, $f_n(x,y)$는 n번째 프레임의 그레이 영상에서 (x,y) 화소의 그레이 수준 값을 의미한다. 이 때 $H(F^D)$의 값이 일정 임계값 이하가 될 때, 즉 터치 프레임에서 터치 흔적의 시간적 변화가 거의 없는 경우에는 사용자의 손가락 움직임이 없는 사용자 제스쳐가 정지한 경우로 간주하고 터치 흔적 추출을 위한 영상 처리를 수행하고, 만약 그 이상이라면 사용자 터치가 이동 중이라고 간주하여 터치 흔적 검출을 위한 영상 처리를 수행하지 않고 제스쳐의 시작과 끝 프레임 검출을 위한 사전 태스팅 작업을 계속 반복한다. 그러나 사용자가 테이블 터치의 시작과 끝에서 움직임이 없는 정지 상태를 유지할 경우에는 제스쳐의 시작 또는 끝 프레임으로 인식되기 때문에 터치 흔적 검출을 위한 다음의 영상처리를 수행하게 된다. 따라서 이 문제의 해결을 위해서 이전에 검출된 터치점을 계산하여 위치 비교에 의해서 그 차이가 없다면 움직임이 없는 경우라도 제스쳐의 시작 또는 끝 프레임이 아닌 단순 정지 프레임으로 인식하여 계속적인 영상처리 작업을 수행하지 않도록 한다.

사전 테스팅을 통과한 프레임에 대해서는 터치 흔적 추출을 위해서 먼저 윤곽(Edge) 강조를 목적으로 언샾 마스크(Unsharp mask) 처리를 한다. 언샾 마스크 처리는 아래 식 (3)과 같이 원영상(Original image)에서 노이즈 제거를 위해서 블러링된 영상을 뺀다.

$$f_{us}(x,y)=f(x,y)\ominus f_b(x,y) \quad (3)$$

식 (3)에서 $f_{us}(\cdot)$은 언샾 필터링된 영상을 의미하고, $f(\cdot)$와 $f_b(\cdot)$는 각각 원영상과 블러링된 영상을 나타낸다. 그리고 $\ominus$는 영상차 연산을 의미한다. 이때 $f_b(\cdot)$를 구하기 위해서 사용하는 저주파 필터는 식 (4)과 같이 모든 필터의 계수 $c(k,l)$가 필터 내의 화소 수의 역수인, n × n 크기의 평균값 평활화(Smoothing) 필터를 사용한다[4].

$$c(k,l)=\frac{1}{(2N+1)^2} \quad -N\le k,l\le N \quad (4)$$

만약 평활화 필터의 효과에도 제거되지 않은 임펄스 잡음이 있다면 비선형 필터인 메디안(median) 필터를 적용한다. 그리고 보다 완전한 언샾 필터링 처리된 영상 $f_{cus}(\cdot)$을 구하기 위해서는 아래 식 (5)와 같이 처리한다.

$$f_{cus}(x,y)=f(x,y)\oplus(k\times f_{us}(x,y)) \quad (5)$$

식 (5)에서 $\oplus$는 영상합 연산이고, k는 필터 상수를 나타낸다. 이때 k값이 1보다 클 경우에는 High-boost 고주파 필터로 사용된다.

다음 처리로는 윤곽 강조에 의해서 감소된 터치점 신호를 확장시키기 위해서 식 (6)과 같이 그레이 레벨 값에 따른 논리곱 연산을 통해서 선명한 터치 흔적을 얻어 낸다.

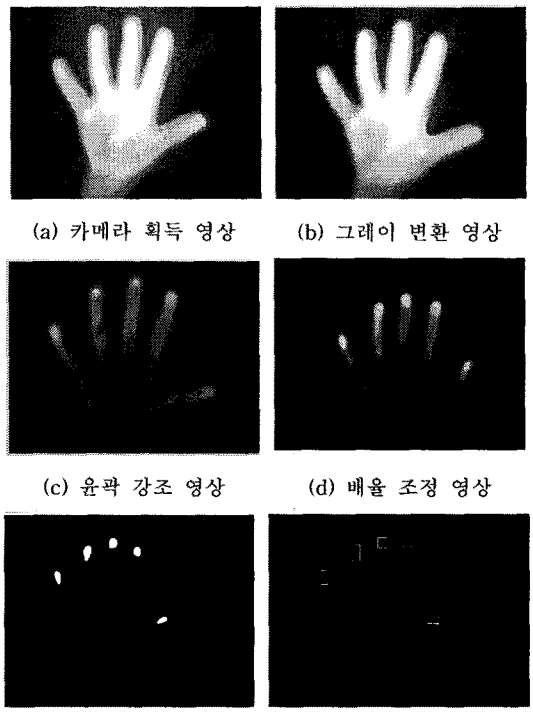
$$f_s(x,y) = S \times f_{us}(x,y) \quad (6)$$

식 (6)에서 S는 배율 상수이다. 멀티 터치 흔적을 추출할 만한 배율 조정이 끝난 다음에 사용자 제스쳐 인식에 사용될 손가락 끝 점들만을 얻어내기 위해서 식 (7)과 같이 임계값 T에 의해서 정류 처리된 이진영상 $f_t(\cdot)$을 생성한다.

$$f_t(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & otherwise \end{cases} \quad (7)$$

마지막으로 검출된 사용자 멀티터치 흔적들은 라벨링 작업에 의해서 사용자 제스쳐 인식에 요구되는 터치점들의 좌표를 계산한다. 라벨링 방법에는 여러 가지가 있으나 윤곽선 추출을 기반으로 하는 Freeman 체인 코드(Chain code) 알고리즘[5]을 사용하여 윤곽선을 얻어 낸 뒤에 윤곽선 상의 점을 체인 코드로부터 좌표로 변환하여 각 터치점을 포함하는 최소경계사각형(MBR: Minimum Bounding Rectangle)을 구한다. 이렇게 추출된 각 터치점들의 최소경계사각형 좌표로부터 각 터치점의 중점을 계산하여 TouchData 구조체에 저장하고 다음 절의 제스쳐 인식 연동을 위한 변수로 사용한다. 이때 사용되는 TouchData 구조체는 검출된 전체 터치점의 개수와 각 터치점의 멀티터치 디스플레이 상에서의 (x, y) 위치 좌표를 포함한다.

그림 3은 이상에서 설명된 멀티 터치점 검출을 위한 영상처리의 각 단계에서 생성되는 결과 영상을 나타낸다.



(a) 카메라 획득 영상 (b) 그레이 변환 영상

(c) 윤곽 강조 영상 (d) 배율 조정 영상

(e) 임계처리 영상 (f) 라벨링 영상

그림 3. 터치점 검출 과정

Fig. 3 Touch-Points Detection Processing

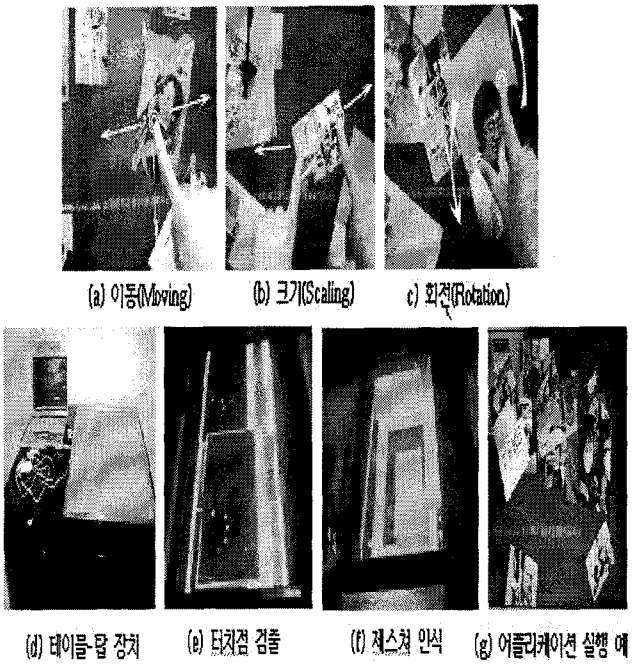
IV. 사용자 제스쳐 인식

본 논문에서는 TUIO 프로토콜을 통해 플래시로 전송된 터치 데이터를 이용하여 플래쉬 어플리케이션에서 적용 가능한 객체의 위치 이동(Moving), 크기(Scaling), 회전(Rotation) 등의 세 가지 제스쳐 인식 알고리즘을 구현하다. TUIO는 실제로 사용자가 물건과의 접촉에 의해서 디지털 정보를 조작하기 위한 인터페이스 기술로서 터치 이벤트와 접촉 물건의 상태 등과 같은 상호작용 표면에서 발생하는 추상적 묘사의 전송이 가능하다. 따라서 이 프로토콜은 멀티터치 디스플레이에서 발생하는 이벤트를 추적하는 어플리케이션으로부터 제어 데이터를 코드화하고, 그 프로토콜을 해독할 수 있는 클라이언트 프로그램으로 전송할 수 있다[6].

구현하는 알고리즘은 그림 4와 같이 터치점의 개수에 따라서 적용하고자 하는 어플리케이션에 최적화된 제스쳐 인식 알고리즘을 제공한다. 사용자 제스쳐 인식 알고리즘에서 제일 먼저 처리되는 것은 한 변형을 목적으로 하는 각 객체 내에서 발생하는 터치점의 개수(NOBPs: Number Of Blob Points)이다. 본 논문에서는 터치점의 개수에 따라서 아래와 같이 제스쳐 인식을 수행한다.

NOBPs = 1(Moving) | 2(Scaling) | 3(Rotation)

이렇게 터치점의 개수에 의해서 제스쳐 인식을 위한 1차 분류를 수행하면 기존의 방법들에 존재하는 대칭성 크기변환 문제나 두 객체 동시 반응 등의 문제를 해결할 수 있다.



(a) 이동(Moving) (b) 크기(Scaling) c) 회전(Rotation)

(d) 테이블-탑 장치 (e) 터치점 검출 (f) 제스쳐 인식 (g) 어플리케이션 실행 예

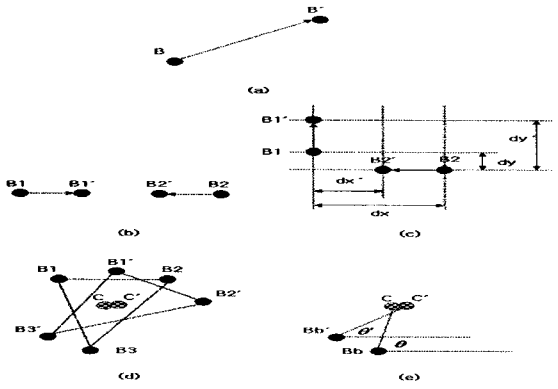


그림 4. 사용자 제스쳐 인식을 위한 터치점 인식
Fig. 4 Touch-Points Recognition for User Gesture Recognition

먼저 객체 위치 변환을 위한 이동(Moving) 동작(그림 4(a))은 한 손가락을 이용하여 객체를 이동시키는 동작을 의미하며 아래 식 (8)에 의해서 터치점 B의 이동 변위 dx, dy가 계산된다.

$$dx=(B_x^{'}-B_x),\quad dy=(B_y^{'}-B_y) \tag{8}$$

여기서 B와 $B^{'}$는 각각 위치 변환을 위한 이동의 시작점과 끝점의 좌표를 나타낸다. 다음으로 크기 변환을 위한 확대 및 축소 동작 인식은 두 손가락을 이용하여 객체를 확대 및 축소하는 동작(그림 4(b))으로서 아래 식 (9)과 같이 두 터치점 $B1$, $B2$의 이동 변위에 의해서 크기 배율 상수 Sx, Sy를 계산한다.

$$S_x=\frac{|B1_x^{'}-B2_x^{'}|}{|B1_x-B2_x|},\quad S_y=\frac{|B1_y^{'}-B2_y^{'}|}{|B1_y-B2_y|} \tag{9}$$

일반적으로 크기 변환은 두 손가락의 이동한 변위가 대칭성의 조건을 만족하는 경우 - 각자 반대 방향으로 향하거나 마주보는 방향으로 향하는 경우 - 확대 및 축소의 충분 조건을 만족한다. 따라서 이 조건을 만족하는 경우에 두 터치점 $B1$, $B2$의 변위 거리비를 구해서 배율값을 결정한다. 그러나 본 논문에서는 그림 4(c)의 경우와 같이 이 조건을 만족하지 않아도 한 객체 내에서 두 터치점의 이동 변위가 인지된다면 크기 변환은 수행된다. 마지막으로 그림 4(d)의 회전 변환은 세 손가락을 이용해서 객체를 회전하는 동작을 의미하며 그림 4(e)와 같이 세 터치점 $B1$, $B2$, $B3$의 중점 C와 가장 아래 터치점 Bb가 회전 변위 전과 후에 이루는 각 θ, $\theta^{'}$의 차이에 의해서 회전각 R_θ를 아래 식 (10)을 이용하여 계산한다.

$$R_\theta=\theta^{'}-\theta \tag{10}$$

$$\theta=\tan^{-1}\left(\frac{|C_y^{'}-Bb_y^{'}|}{|C_x^{'}-Bb_x^{'}|}\right),\quad \theta^{'}=\tan^{-1}\left(\frac{|C_y^{'}-Bb_y^{'}|}{|C_x^{'}-Bb_x^{'}|}\right)$$

$$C(x,y)=\left(\frac{B1_x+B2_x+B3_x}{3},\frac{B1_y+B2_y+B3_y}{3}\right)$$

V. 시뮬레이션

본 논문에서 제작 및 구현된 DI 방식의 멀티터치 테이블-탑 디스플레이 장치의 구동을 위한 터치점 검출 프로그램은 노트북 컴퓨터(1GHz, Intel CoreDuo, 1G RAM)의 Microsoft Visual Studio 2008 환경에서 OpenCV 라이브러리[7]를 기반으로 TouchLib[2]를 수정하여 사용하였다.

검출된 터치점의 어플리케이션 전송은 TUIO 웹페이지[6]에서 프로토콜 지원을 위해서 제공되고 있는 OSC Library 및 소스를 이용하였다. 그리고 어플리케이션을 위한 시뮬레이션은 현재 OS 버전에서는 한 프로세스에서 오직 하나의 사용자 포커스(Single focus)를 지원하고 있기 때문에 멀티 포커스(Multi focus) 지원하는 Adobe사의 플래시(Flash)를 이용한 Flash AS3 프로그램을 제작하여 제스쳐 인식 알고리즘을 구현하고 시뮬레이션 하였다. 그림 5는 본 논문에서 구현한 멀티터치 테이블-탑 디스플레이어와 그 시스템에서 제공하는 제스쳐 종류와 시뮬레이션 예이다. 구현된 시스템의 시뮬레이션 결과 본 논문에서 사용된 멀티터치 사용자 제스쳐 인식은 표 1과 같이 기존의 연구에서 지원되지 않던 이동 및 회전 변환에 있어서 두 객체의 동시 변환 기능을 지원한다. 또한 크기 변환에 있어서 확대 및 축소를 위한 사용자 제스쳐의 대칭 이동 발생 조건을 만족하지 않아도 되기 때문에 임의 방향의 변형이 가능한 편의성을 제공한다. 그리고 멀티 터치점 추출을 위한 영상처리 작업에 있어서 시스템 지연 시간의 감소에 의한 시스템 반응 속도의 향상을 위해서 본 논문에서 제안하는 사전 테스팅 작업의 효율성 검사를 수행하였다.

실제 시스템 동작에서 많은 지연이 요구되는 실영상 캡쳐 작업의 지연 시간을 제외하고, 하나의 터치 제스쳐를 완성하는 데에 n개의 프레임이 발생한다고 하면, 사전 테스팅 작업을 수행하지 않는 경우의 터치 흔적 추출을 위해서 요구되는 최소 시스템 지연 시간은 아래 식 (12)와 같다.

$$\begin{aligned}T_{SL}&=2\times T_{NAB}+(n-2)\times T_{AB} \\ &=2\times\sum_{i=1}^{5}T_i^{NAB}+(n-2)\times\sum_{i=1}^{5}T_i^{AB} \\ &=2\times\sum_{i=1}^{5}T_i^{NAB}+(n-2)\times\left(\sum_{i=1}^{2}T_i^{AB}+\sum_{i=3}^{5}T_i^{AB}\right) \\ &\text{where, } T_{RSL}=\sum_{i=3}^{5}T_i^{AB}\end{aligned} \tag{12}$$

여기서 T_{SL}는 시스템 지연시간을 T_{NAB}, T_{AB}는 각각 활동 터치 흔적이 없는 경우와 활동 터치 흔적이 있는 경우의 시스템 지연시간을 의미한다. 그리고 T_i는 표 2에 나타낸 봐와 같이 터치 흔적 추출을 위한 영상 처리 단계 가운데 i번째 단계에서의 시스템 지연 시간을 의미한다. 따라서 터치 흔적을 추출하기 위한 시스템의 총 지연 시간은 터치 흔적의 활동이 없는 제스쳐의 시각과 끝에서의 지연과 그 사이의 이동 변위 동안에 발생하는 활동 흔적이 있는 경우의 시스템 지연 시간과의 총합을 의미하게 된다. 그러나 사전 테스팅 작업에 의해서 터치 제스

쳐의 중간 프레임, 즉 터치 흔적의 이동이 있는 경우에 대한 $T_3 \sim T_5$까지의 시스템 지연이 제외된다면, 식 (12)에서 T_{RSL}는 본 논문에서 제안한 방업에 의해서 감소 가능한 최대 시스템 지연 시간이 된다.

표 1. 관련 연구와의 성능비교

Table 1. Comparison of Our approach and the Related Works

제스쳐 유형		기존연구	본 연구
이 동	두객체동시이동	X	O
크 기	임의방향변형	X	O
회 전	두객체동시회전	X	O

만약 n=10 이라면, 표 2의 측정값에 의한 식 (12)의 계산에 의해서 일반적 기존 방법에 의한 시스템 지연 시간은 98.9382ms가 요구되며, 본 논문의 사전 테스팅에 의해서 T_{RSL}의 시스템 지연이 제거된다면 36.44082 ms의 시스템 지연이 요구된다. 따라서 본 논문에서 제안한 사전 테스팅의 효과는 산술적으로 최대 37%까지 시스템 지연 시간을 줄일 수 있다.

표 2. 터치 흔적 추출을 위한 시스템 지연

Table 2. System latency for extracting Touch-blobs

Filter type	T^{NAB}	T^{AB}
T_1 : 그레이변환	1.490972 ms	1.492368 ms
T_2 : 사전테스팅	0.636114 ms	0.636952 ms
T_3 : 윤곽강조	4.997004 ms	5.247874 ms
T_4 : 배율조정	2.141334 ms	2.156699 ms
T_5 : 임계처리	0.437729 ms	0.407594 ms

VI. 결 론

본 논문에서는 대부분의 멀티터치 장치에서 적용하고 있는 FTIR 방식의 단점을 보완할 수 있는 확산 투광에 기반한 DI 멀티터치 테이블-탑 디스플레이어 장치를 제작하였다. 또한 제작된 멀티터치 하드웨어 장치는 터치점 검출이나 제스쳐 인식을 위해서 기존의 제공되는 범용 라이브러리를 수정하여 멀티터치 장치의 소프트웨어적 성능 개선 방향을 제시하였다. 제안한 방법은 멀티 터치 흔적의 검출을 위한 영상 처리 과정에 있어서 객체 변형 과정 중의 보간 프레임의 제거를 위한 사전 테스팅 작업을 적용함으로서 시스템 지연 시간의 감소가 가능하였다. 또한 사용자 제스쳐 인식에 있어서도 해당 멀티터치 지원 어플리케이션에 대한 최적화를 위해서 디스플레이 상의 변형을 목적으로 하는 객체들의 한 객체 내에서 발생하는 터치점 개수를 1차 부류로 하는 제스쳐 인식 알고리즘을 제안함으로서 동시 객체 변형이나 대칭 이동 조건 등의 문제점들에 대해서 새로운 해법을 보였다. 그러나 사용자 터치점의 강도 문제나 사용 주변 환경에 의존하지 않는 터치점 추출 알고리즘의 일반화 문제 등이 남아 있다. 향후 이들 문제의 해결과 함께 가변적 모바일 상황에서의 터치점 검출이나 특정 어플리케이션만의 최적화된 제스쳐 인식 모듈 설계가 수행된다면 제안된 논문의 활용성은 크다고 할 수 있다.

참 고 문 헌

[1] Jefferson Y. Han, "Low-cost multi-touch sensing through frustrated total internal reflection",Proceedings of the 18th annual ACM symposium on User interface software and technology, pp.115-118, New York, NY, USA, 2005.

[2] David Wallin, *Touchlib: an opensource multi-touch framework*, 2006. (http://www.whitenoiseaudio.com/touchlib/)

[3] L.Y.L.Muller,"Multi-touch display:design,application and performance evaluation", M.S. Thesis, Univ. of Amsterdam, 2008.

[4] Rafael G. Gonzalez, *Digital Image Processing 3/e*, Prentice Hall, 2007.

[5] Herbert Freeman, "Computer processing of line-drawing images", Computing Survey v6, no1, pp.57-97, 1974.

[6] http://www.tuio.org

[7] *Open Source Computer Vision Library: Reference Manual*, Intel, 2001. (http://www.intel.com/technology/computing/opencv/)

[8] Dan Saffer, *Designing Gestural Interfaces*, O'Reilly Media, Inc., 2008.

[9] 이우범, "변형 보정과 원형 추적법에 의한 교통 표지판 인식", 한국신호처리시스템학회논문지, Vol.9, No.3, pp.188-194, 2008.

[10] http://nuigroup.com/touchlib/

[11] http://opensoundcontrol.org/introduction-osc

[12] 이승재, 이우범, "확산 투광 방식에 의한 멀티터치 테이블-탑 디스플레이 구현", 한국신호처리시스템학회 하계학술대회, 제10권, 제1호, pp.126-129, 2009.

이 우 범(Woo-Beom Lee)

1995년 2월 영남대 컴퓨터공학과(공학사)
1997년 2월 영남대 컴퓨터공학과(공학석사)
2000년 8월 영남대 컴퓨터공학과(공학박사)

2000년 3월~2004년 2월 대구과학대 컴퓨터공학과 전임강사
2004년 3월~2007년 2월 영남대 전자정보공학부 객원교수
2007년 3월~현재 상지대 컴퓨터정보공학부 조교수
※ 주관심분야 : 시각인지, 컴퓨터비젼. 의료영상처리