동작기반의 체험형 리모트 콘트롤

Motion-based Remote Control for the Contents of Experiences

정재훈, Jaehoon Jung*, 황인욱, Inwook Hwang*, 이인, In Lee*, 이채현, Chaehyun Lee*,
박건혁, Gunhyuk Park*, 황재인, Jane Hwang*, 최승문, Seungmoon Choi*, 김정현, Jounghyun Gerard Kim**
*포항공과대학교 컴퓨터공학과 가상현실 연구실, **고려대학교 컴퓨터학과

요약 이 논문에서는 동작기반 체험형 리모트 콘트롤러가 사용자와 상호작용을 하기 위해 갖춰야 할 기본적인 요소가 무엇인지 설명하고 체험형 리모트 콘트롤러의 동작 정보를 처리하는 시스템의 구성을 제안한다. 동작기반의 체험형 리모트 컨트롤러는 블루투스로 서버와 양방향 통신을 하게 되며 가속도 센서가 장착되어 기존의 버튼입력 기반의 리모트 컨트롤러와는 달리 동작을 이용한 입력을 할 수 있고 사용자에게 진동모터를 통한 촉감 피드백을 전해주어 사용자의 몰입감을 증가시킬 수 있도록 하였다. 가속도 센서를 이용한 동작 인식을 위해서는 두 가지 알고리즘이 개발되어 적용되었다. 동작기반의 체험형 리모트 컨트롤은 논문에 제시된 예 이외에도 기존의 버튼 기반으로 만들어진 응용프로그램에 쉽게 적용시킬 수 있으며 특히 엔터테인먼트 분야에서 사용자의 흥미와 몰입감을 더해주기 위해 사용될 것으로 기대된다.

핵심어: Handheld VR, Motion Recognition, Remote Controller

1. 서론

기존의 리모트 콘트롤러는 단방향 통신과 버튼 입력 방식만을 제공하였다. 또한 버튼 입력에 대한 반응으로 시각적인 요소와 청각적인 요소만을 제공하였다. 이 논문에서는 체험형 리모트 콘트롤러를 이용하여 사용자의 동작을 통해 버튼 입력을 대체하거나 사용자가 두 가지(버튼, 동작) 입력 방식을 자유롭게 선택하고 확장할 수 있는 입력방식을 제안한다. 또한 다양한 상호작용을 위해 강도 조절이 가능한 햅틱피드백을 줄 수 있는 시스템을 제안한다. 마지막으로 체험형 리모트 콘트롤러를 이용한 어플리케이션을 소개한다.

핸드헬드 체험형 인터페이스의 종합적인 연구는 우리나라에서 아직 찾아 볼 수 없다. 일부 대학교에서 가상현실이나 데스크탑/아케이드 게임과 관련된 체험형 인터페이스, 비전 인터페이스 등의 연구는 되고 있으나, 핸드헬드 플랫폼에 대한 종합적인 연구는 매우 드물다. 그러나 국외의 경우, 핀란드의 노키아는 이미 다양한 형태의 다중양식 인터페이스 기반의 체험형 휴대폰기술에 열을 올리고 있다. 2004년부터 진행되고 있는 MONA(Mobile multi-mOdal Next generation Applications) 프로젝트는 기존의 휴대폰에 각종 센서와 다양한 디스플레이를 통하여 길 안내, 게임, 멀티모달 메세징서비스와 같은 응용소프트웨어 개발을 시도하고 있다[1]. 또한 Symbian OS를 이용한 컴퓨터 비전 라이브러리와 증강현실 엔진이 뉴질랜드 캔터버리 대학의 Mark Billinghurst 박사, 독일 바이마르 대학 Oliver Bimber 교수팀에 의해 개발되었다[2][3].

이 논문의 2장에서는 동작기반의 체험형 리모트 콘트롤러를 위한 요구조건과 시스템 구성방법, 위치추적과 동작인식에 적용된 소프트웨어 알고리즘에 대해 설명하고 3장에서는 구성한 체험형 리모트 컨트롤러를 이용한 응용프로그램의 예를 소개하도록 한다.

2. 동작기반 체험형 리모트 콘트롤러를 위한 요구조건

동작기반의 체험형 리모트 컨트롤러에서 리모트 컨트롤러 그 자체는 사용자의 입력을 받아들이기 위한 장치로서의 기능만을 수행하며 입력신호(버튼, 동작으로부터 얻어낸 가속도)를 서버로 전송한다. 이 신호를 전송받은 서버는 신호처리와 동작인식과정을 거쳐 사용자에게 그래픽, 택타일 렌더링을 출력으로서 보내게 된다. 체험형 리모트 컨트롤러의 시스템 개념도는 <그림 1>과 같다.

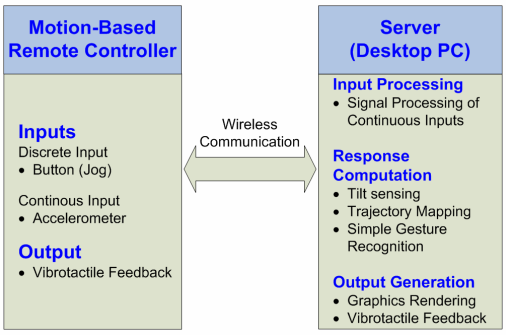


그림 1 <체험형 리모트 컨트롤러의 시스템 개념도>

2.1. 하드웨어 플랫폼

체험형 리모트 콘트롤러를 구성하기 위하여 갖추어야 할 하드웨어의 조건은 아래와 같다. 첫째, 사용자가 쉽게 들고 동작으로 입력을 취할 수 있도록 소형화가 가능해야 한다. 둘째, 배터리의 부피와 무게를 줄이고 긴 연속사용시간을 보장하기 위해서 소비전력이 낮아야 된다. 셋째, 사용자의 동작을 인식하기 위한 센서를 갖추어야 된다. 넷째, 센서로부터 얻은 데이터를 서버로 보내고 진동피드백 정보를 서버로부터 받기 위해서 양방향 무선통신이 가능해야 한다. 다섯째, 동작입력이 어려운 상황에서의 입력을 위해서 기존의 리모트 콘트롤러의 기능(버튼 입력)을 포함해야 한다. 이 다섯 가지의 기본 요건을 만족하는 체험형 리모트 콘트롤러의 구조는 <그림 2>과 같다.

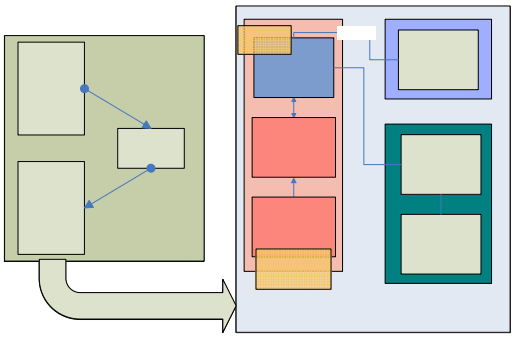


그림 2 <체험형 리모트 콘트롤러 하드웨어 구조>

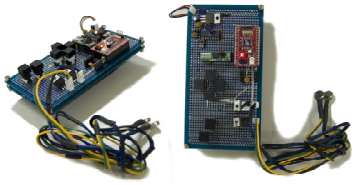


그림 3 <체험형 리모트 콘트롤러 프로토타입>

2.1.1. 하드웨어 상세설명

가. 블루투스

시리얼 통신형 블루투스 모듈을 이용에 서버의 블루투스 동글과 통신한다. 통신속도는 115200bps로 고정되어 있으며 윈도 API에서 지원하는 시리얼통신 라이브러리와 호환된다. 또한 지연시간와 입출력대역폭의 우선순위 조정이 가능하다. 사람이 동작과 디스플레이상의 차이를 인식하기 시작하는 지연시간은 83ms인데 비해서[4] 블루투스의 통신 지연시간은 평균 24ms로 그 이하이므로 동작인식에 문제없이 활용할 수 있다.

나. MCU

ATMEGA48을 사용하였다. 20Mhz로 동작하며 20MIPS의 처리속도를 가진다. 1 USART와 디지털 센서와의 통신을 위한 I2C, SPI 통신을 지원한다. 3개의 카운터와 이후 진동모터를 콘트롤하기 위한 6개의 PWM 채널을 가지고 있다. 디지털필터를 프로그래밍하여 내장하였다.

다. 진동피드백

전원 트랜지스터와 MCU의 PWM 채널을 이용하여 총 3개의 진동모터를 제어한다. 분해능은 8bit로 0에서 255까지의 단계를 가지고 진동모터의 물리적 세기를 조절 할 수 있다. 제어방식은 서버와의 통신을 통해서 진동모터를 개별적으로 제어한다.

라. 전원부

휴대용 건전지에 적합한 마이크로칩의 TC1264-3.3V 레귤레이터를 사용하였다. 매우 적은 전압강하 특성을 지녔고 최대 출력전류는 800mA 이다. 출력전압의 정확도는 3.3V+-0.5%를 보장한다.

마. 가속도계

LIS3LV02DQ 3축 가속도계를 이용하였다. 게임서버와의 통신을 통해 디지털 필터링을 제어할 수 있고 중력가속도의 범위도 2g와 6g로 어플리케이션에 맞게 변경이 가능하다. ADC 분해능 또한 12bit와 16bit중 선택하여 사용 가능하다. 마지막으로 가속도 센서의 대역폭 또한 통신을 통하여 제어할 수 있다.

바. 버튼

버튼은 총 8개를 연결 가능하다 아답터의 형태로 별도의 버튼(조이스틱, 조그, 다이얼)이 연결 가능하다. 버튼의 동시입력을 확인 가능하며 버튼의 상태가 서버로 실시간 전송된다.

2.2. 소프트웨어 구조

소프트웨어는 크게 두 부분으로 나누어진다. 콘트롤러에 들어갈 임베디드 프로그램과 서버에 들어갈 메인 프로그램이다. 임베디드 프로그램에서는 가속도 센서로부터 값을 읽어 블루투스를 통해 보내고, 서버로부터 데이터를 받아 진동모터를 작동시켜 사용자에게 햅틱 피드백을 준다. 서버의 메인 프로그램은 멀티 쓰레드 프로그램으로 구성되며, 콘트롤러로부터 데이터를 받아서 처리하는 업데이트 쓰레드와 처리된 데이터를 사용하여 게임에 적용하는 게임 쓰레드로 나누어진다. 소프트웨어의 구조는 아래의 <그림 4>과 같다.

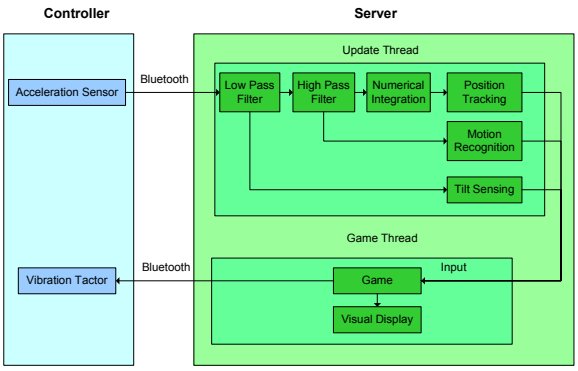


그림 4 <체험형 리모트 콘트롤러 소프트웨어 구조>

2.2.1. 업데이트 쓰레드

업데이트 쓰레드는 우선 콘트롤러로부터 네 개의 값 (3축의 가속도와 버튼의 상태)을 전달받는다. 이때 전달받은 가속도 값은 콘트롤러 프로그램 자체에서 어느 정도 필터링을 한 상태지만, 센서의 특성상 값이 불안정하게 될 수도 있다. 이러한 값을 그대로 사용하면, 위치 추적이나, 동작인식에서 문제가 발생할 수도 있으므로, 로우패스필터링을 통해서 오동작의 가능성을 줄여 준다.

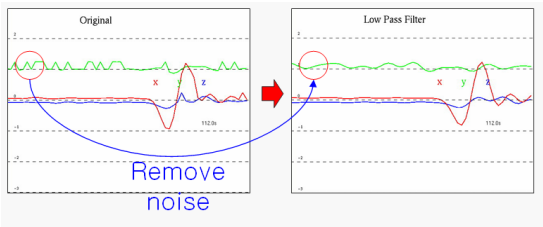


그림 5 <로우패스필터링을 통한 노이즈 제거>

위의 <그림 5>에서와 같이 로우패스필터링을 거치고 나면, 노이즈가 현저하게 줄어드는 것을 확인할 수 있다.

로우패스필터링을 할 경우, 필터의 계수를 선택하는 문제가 중요하다. 계수의 차수를 높이면 거의 완벽하게 노이즈를 제거할 수 있지만, 반응속도, 즉, 실제 움직임 뒤에 프로그램에서 결과로 반영되는 데까지의 시간(Phase)이 오래 걸리게 되므로, 적절한 값을 찾는 것이 중요하다. 본 연구에서는 반응속도가 크게 떨어지지 않으면서 적절하게 노이즈를 제거할 수 있도록, 응용 프로그램에 따라 필터의 종류와 계수를 선택할 수 있도록 하였다.

로우패스필터를 거친 후에는 가속도 값을 사용하여 콘트롤러의 기울기를 측정하거나, 다른 모듈들에 적용시키기 위해 하이패스필터를 적용시킨다.

일반적으로 콘트롤러가 전송하는 가속도 값을 살펴보면, 사용자로 인해 발생하는 가속도와 중력가속도로 인해 발생하는 가속도가 합쳐져 있다. 동작인식과 위치 추적을 위해서는 중력가속도를 제외하고 사용자로 인해 발생하는 가속도만을 뽑아낼 필요가 있다. 그러므로 하이패스필터를 통해 중력가속도 성분을 제거한 값을 활용하였다.

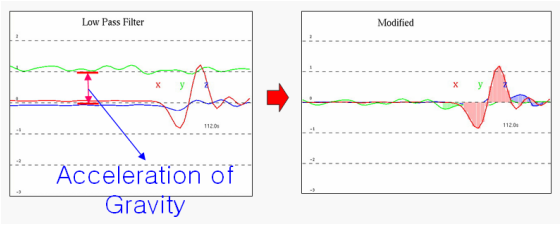


그림 6 <하이패스필터링을 통한 중력가속도 제거>

위의 <그림 6>에서 볼 수 있듯이, 하이패스필터를 거치고 나면, 중력가속도로 인한 가속도 값이 사라지는 것을 알 수 있다.하이패스필터를 통과한 가속도 값은 동작인식에 사용되거나, 적분을 통해 위치추적을 하는데 사용될 수 있다.

콘트롤러로부터 전달된 가속도 값은 위와 같이 적절한 필터링을 거쳐 기울기 측정, 동작인식, 위치 추적에 사용되며, 각각의 결과는 게임 쓰레드에 전달되어, 실제 게임에 적용되게 된다.

2.2.2. 게임 쓰레드

업데이트 쓰레드에서 게임 쓰레드로 값을 전달하는 방법은 크게 두 가지로 분류된다. 위치 추적과 기울기 측정 결과는 미리 정의된 전역 변수에 값이 저장되어 각각의 쓰레드에서 값을 저장하고 읽어올 수 있다. 동작인식 결과는 하나의 동작이 발생하였을 때, 이에 매칭되는 키보드 값을 이벤트로 발생시키는 방법을 사용한다.

게임 쓰레드에서는 이렇게 전달받은 위치 정보, 동작정보, 기울기 값을 사용하여 적절하게 현재 게임에 적용시키고, 필요한 경우, 현재 게임의 시각 디스플레이를 수행하고, 블루투스를 통해 콘트롤러에 진동 값을 전달하여 사용자에게 필요한 순간에 진동 피드백을 느낄 수 있도록 한다.

2.3. 위치 추적

컨트롤러로부터 전송되는 가속도를 이용하여, 컨트롤러의 현재 위치를 추적할 수 있다. 우선 컨트롤러로부터 전송된 가속도에 로우패스필터를 적용하여 센서 자체에서 발생하는 노이즈 성분을 제거한 뒤에 다시 하이패스필터를 적용하여 중력가속도 성분을 제거하면 남게되는 순수하게 사용자의 움직임으로 인해 발생된 가속도를 사용하여 컨트롤러의 위치를 추적한다.

컨트롤러의 가속도로부터 위치를 추적하기 위해서 기본적으로 아래 식과 같이 가속도를 시간에 대해서 두 번 적분하여 컨트롤러의 위치를 찾아내게 된다.

하지만 로우패스필터를 적용한 뒤에도 가속도 값에서 노이즈 성분만이 완전히 제거되는 것이 아니라, 일부 노이즈 성분은 남고 움직임에서 발생된 데이터는 일부가 소실되는 것이기 때문에, 필터링된 가속도 값을 바로 적분하여 위치를 계산하면, 많은 오차가 발생한다. 그러므로, 노이즈의 값을 분석한 뒤, 노이즈의 최대값을 임계값으로 두고, 가속도가 이 임계값을 넘어선 경우에만, 적분시켜 위치 값을 계산한다. 현재는 노이즈의 임계값이 중력가속도의 0.15배로 설정되어 있

으며 이 값은 변수를 통해 변경할 수 있다.

임계값이 넘은 가속도를 사용하여 적분을 하더라도, 가속도 값 자체가 정확하지 않기 때문에 두 번의 적분을 통해 계산되는 위치 값은 많은 오차를 가진다. 특히, 사용자가 정지 상태에서 시작하여 동작을 실행한 뒤에 다시 멈추는 과정에서 생기는 가속도를 모두 합하면 이론상 0이 되어야 한다. 하지만 실제 측정되는 가속도를 모두 더해보면, 가속도 센서 자체의 오류와 임계값을 넘은 가속도 값만이 적분에 포함되기 때문에 생기는 오류로 인해 실질적으로는 0이 아닌 값이 된다. 이렇게 사용자가 움직임을 멈췄을 때, 가속도의 합이 0이 아니면, 속도가 초기값(0)으로 돌아가지 못하게 되고, 그로 인해 위치 값은 무한대로 가게 된다. 이러한 가속도 센서 자체의 한계를 극복하고, 컨트롤러의 위치를 적절하게 추적하기 위하여 초기 가속도 보상 알고리즘을 사용하였다.

사용자의 움직임에 따른 가속도 값을 살펴보면, 대부분 <그림 7>와 같은 움직임을 보인다. 즉, 움직임이 시작될 때 가속도 값이 증가하고, 움직임이 멈춰질 때 가속도 값이 반대 방향으로 증가한다.

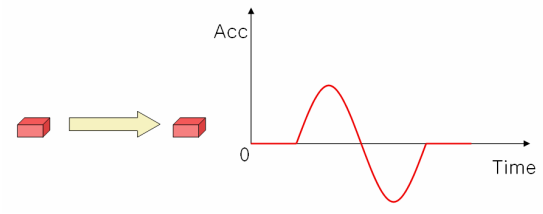


그림 7 <단방향 동작에서의 이상적인 가속도 변화>

하지만, 앞서 말한 센서 자체의 오류로 인해 대부분 <그림 8>과 같이 움직임이 시작될 때의 가속도의 합과 움직임이 끝날 때의 가속도의 합이 다르게 관찰되었다. 그러므로 실제 컨트롤러의 움직임을 멈추었을 때도, 계산된 위치 값은 계속 변화하여, 결국은 무한대로 증가하는 오류가 발생하였다.

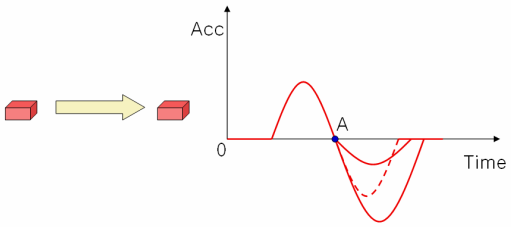


그림 8 <간단한 단방향 동작에서의 실제 가속도의 변화>

초기 가속도 보상 알고리즘이란, 움직임이 시작될 때 생기는 가속도의 궤적을 저장하고, 가속도가 반대 방향으로 전환되는 순간(점A)부터 센서로부터 전달되는 가속도(실선)는 무시하고, 이전에 저장했던 가속도의 궤적을 반대방향(점선)으로 바꾸어 그대로 적용하는 방법이다. 이 알고리즘을 사용하면, 어떤 식으로든 가속도가 발생한 경우, 똑같은 가속도가 반대방향으로 적용이 되므로, 속도가 초기 속도(0)로 돌아가게 되고, 위치값이 무한대로 가는 경우는 없어진다. 그리고 한방향으로 움직이고 멈추는 간단한 동작의 경우, 초기 가속도만 제대로 측정된다면, 실제 움직임과 비슷한 위치를 추적해 낼 수 있다.

하지만 <그림 9>와 같이 단방향의 간단한 동작이 아닌, 두 동작이 연속적으로 일어나는 복합적인 움직임의 경우, 실제 가속도는 점선과 비슷하게 나타나지만, 초기 가속도 보상 알고리즘에 의해 적용되는 값은 실선이므로, 실제 움직임과는 다른 위치를 예측하게 된다. 이러한 오차가 계속 쌓이게 되면, 결국 실제 위치와는 완전히 다른 위치를 예측하게 된다. 그러므로 초기 가속도 보상 알고리즘은 단방향의 간단한 움직임에만 적용될 수 있는 한계를 가진다.

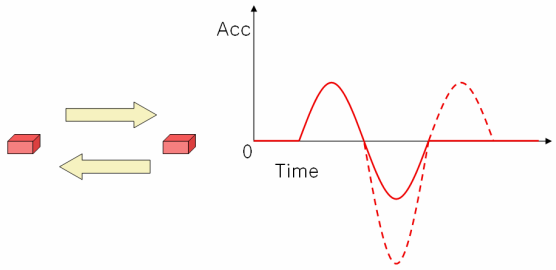


그림 9 <복합적 동작에서의 알고리즘의 적용>

2.4. 동작인식

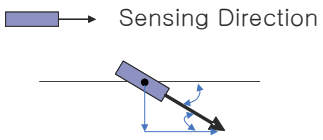
응용 프로그램의 성격에 따라 사용자가 자연스럽게 느끼며 몰입감을 느낄 수 있는 동작도 달라지게 된다. 예를 들면 야구 게임에서는 공을 던지는 동작이나 배트를 휘두르는 동작을 가장 자연스럽게 느낄 것이고, 펜싱 게임이라면 손에 쥐고 찌르는 동작을 가장 자연스럽게 느낄 것이다. 입력을 위한 동작이 얼마나 자연스럽고 사용자가 쉽게 받아들일 수 있는지에 따라 사용자의 응용 프로그램에 대한 몰입도가 결정된다.

이번 연구에서는 리모트 콘트롤러에 내장된 3축 가속도 센서에서 측정된 데이터를 블루투스 모듈을 통해 실시간으로 전송받아 사용자의 동작을 인식하는 데에 이용하였다. 정확한 위치 정보를 얻어낼 수 있다면 쉽게 높은 신뢰도를 가지는 동작 인식을 할 수 있겠지만 높은 가격과 추가로 외부 장치를 필요로 하는 문제가 있으므로 적절한 알고리즘을 이용하여 가속도 데이터만으로 사용자의 동작을 정확히 파악하는 것이 관건이라 하겠다.

따라서 자동차 드라이빙 시뮬레이션과 같이 연속적으로 매끄럽게 이어지는 입력이 필요한 어플리케이션에 적용할 수 있는 동작인식 방법으로 중력 가속도를 이용한 기울기 감지를, 테트리스 게임과 같이 명령이 각각 구분되는 비연속적인 입력이 필요한 어플리케이션에 적용할 수 있는 동작인식 방법으로 가속도의 쓰레시홀딩에 기반한 스테이트 머신을 각각 개발하여 사용하였다.

2.4.1. 중력 가속도를 이용한 기울기 감지

지표면에서는 항상 지면과 수직 방향으로 약 9.8m/s^2 크기의 중력 가속도가 작용하고 있기 때문에 이 중력 가속도의 성분이 리모콘에 내장된 가속도 센서의 x, y, z 세 축에 어떻게 나뉘어 들어가는지 살펴보는 것으로 리모콘이 지면에 대해서 기울어진 각도를 알아낼 수 있다.



$$\sin\theta = \frac{g}{acc}$$

$$\theta = \arcsin(\frac{g}{acc})$$

그림 10 <중력가속도를 이용한 기울기 감지>

위의 <그림 10>에서 리모콘의 좌우 방향을 x축, 상하 방향을 y축이라고 할 때 리모콘이 지면과 평행하게 정지한 상태에서는 y축 자체가 지면에 수직한 방향이 되므로 y축에서 측정되는 가속도의 크기가 약 9.8m/s^{2}이 되고 다른 축의 가속도는 0이 된다. 그러나 위의 <그림 10>에서와 같이 리모콘이 상하로 기울어지게 되면 가속도는 x축과 y축 성분으로 나눌 수 있게 되어 그 기울어진 각도를 Θ라고 했을 때 가속도 센서의 y축에서 측정되는 가속도의 크기는 약 9.8m/s^2 × cosΘ가 되고, x축에서 측정되는 가속도의 크기는 약 9.8m/s^2 × sinΘ가 된다. 따라서 이 측정된 가속도를 가지고 삼각함수의 역함수로 계산하면 기울어진 각도 Θ를 계산해 낼 수 있다.

단, 사용자가 급격하게 리모콘의 위치나 각도를 변화시키는 경우에 중력 가속도 이외에 사용자의 움직임에 의한 가속도 성분이 측정될 수 있다. 이러한 경우 계산해 낸 기울어진 각도의 값이 실제보다 크거나 작게 나올 수 있는데 사용자의 움직임에서는 움직임을 시작할 때와 끝마칠 때 순간적으로 가속도가 크게 나오고 움직이는 도중에는 사용자의 움직임에 의한 가속도는 거의 측정되지 않는 것이 보통이므로, 빠르게 변화하는 고주파수 성분을 걸러내는 로우패스필터를 이용하여 이러한 원하지 않는 값들을 제거하고 안정적인 데이터를 얻어낼 수 있다.

기울기 감지 방법은 중력가속도를 이용하는 방법이므로 리모콘 전후의 상하 기울기, 리모콘 좌우의 상하 기울기의, 2축 기울기 값을 알아내어 입력에 활용할 수 있다. 기울기 값이 일정 각도를 넘어서면 입력한 것으로 판단하게 하여 디지털 버튼을 대체할 수도 있고 기울기 각도가 크면 큰 값의 입력을, 기울기 각도가 작으면 작은 값의 입력을 하도록 하여 아날로그 버튼의 입력을 대체할 수도 있다.

2.4.2. 가속도의 쓰레시홀딩에 기반한 스테이트 머신

상하좌우의 네 방향의 비연속적인 입력이 필요한 응용 프로그램에서 사용자가 자연스럽게 리모콘을 들고 움직임을 통해 입력하는 실험을 해 본 결과 각각의 입력에 대한 가속도의 크기는 일반적으로 <그림 11>의 그래프와 같이 나타난다.

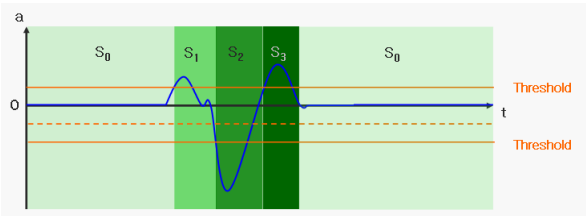


그림 11 <스테이트 머신을 이용한 동작인식>

그래프의 위쪽은 (+)방향의 가속도를 나타내고 아래쪽이 (-)방향의 가속도를 나타낸다. 그래프의 좌우 축은 시간을 나타내며 오른쪽으로 갈수록 시간이 많이 경과되었음을 나타낸다. 처음 나타나는 (+)방향의 언덕은 사용자가 원하는 방향으로 움직임을 시작할 때에 나타나는 가속도이다. 이후에 이어서 나타나는 (-)방향의 큰 골은 사용자가 움직임을 멈추고 다음 동작을 입력하기 위해서 원위치로 돌아오기 위해 힘을 줄 때에 나타나는 것이다. 하나의 입력에 대하여 사용자가 움직이는 폭이 크지 않기 때문에 앞서 움직임을 시작할 때 나타나는 언덕과 멈추고 돌아올 때 나타나는 골 사이에 중간 과정이 거의 없이 바로 이어서 나타난다. (-)방향의 골이 끝나고 나타나는 (+)방향의 언덕은 다음 동작을 준비하기 위해 사용자가 입력을 시작하기 전의 원위치로 돌아와서 움직임을 멈출 때에 나타나는 것이다. 다만 이 언덕은 사용자의 동작 성향에 따라 (돌아오는 움직임이 빠르지 않은 경우) 그 형태가 잘 보이지 않기도 한다.

<그림 11>의 가속도 그래프에 따라 하나의 입력을 크게 네 단계로 나누고 이 단계의 변환에 의해서 사용자의 동작을 인식하도록 하였다. 각각의 상태는 <그림 12>에서와 같이 전환되며 다음과 같은 의미를 갖는다.

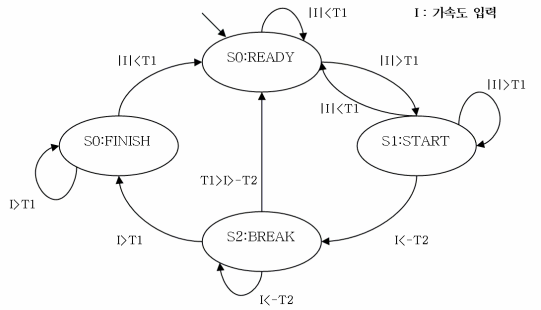


그림 12 <사용된 스테이트 머신의 다이어그램>

S0: 동작을 기다리는 대기상태(Ready)로 프로그램의 시작이나 다른 동작의 끝에서부터 정해진 임계값 T1을 넘는 가속도 입력이 들어와서 S1 상태로 넘어가기 전까지의 상태이다.

S1: 동작하려는 방향으로 가속도를 주는 상태(Start)로 S0에서 임계값 T1을 넘는 가속도 입력이 들어오면 이 상태로 넘어오게 된다. 처음 임계값을 넘어서 S1 상태에 들어왔을 때와 같은 방향으로 임계값 T1을 넘는 가속도 입력이 들어오는 동안 이 상태가 유지되며, 일정시간 동안 이 조건을 유지하지 못하면 사용자의 손떨림이나 센서에서 나오는 노이즈로 인해 발상한 데이터로 판단하여 대기상태인 S0 상태로 돌아가게 된다.

S2: 움직임을 멈추고 원위치로 돌아오기 위해 반대방향으로 가속도를 주는 상태(Break)로 S1 상태로 들어갈 때 인식된 가속도의 반대방향으로 임계값 T2를 넘는 가속도 입력이 들어오면 이 상태로 넘어오게 된다. S2 상태에서는 S1 상태에서 들어온 사용자의 움직임을 실제 어플리케이션에 방향키 입력 이벤트로 전달해 준다. 역시 같은 방향으로 임계값을 넘는 가속도 입력이 들어오는 동안 이 상태가 유지되며, 일정시간 동안 임계값을 넘지 못하면 대기상태인 S0 상태로 돌

아가게 된다.

S3: 원위치에서 움직임을 멈추기 위해 가속도를 주는 상태(Finish)로 이미 사용자의 동작은 입력으로 응용 프로그램에 전달되었지만 입력을 위한 동작이 연속되는 경우에 다음 동작과 혼동되는 것을 피하기 위하여 가속도의 크기가 임계값 T1 아래로 내려갈 때까지 S3 상태에서 기다리게 된다. 가속도가 임계값 T1 아래로 내려가면 다음 입력을 받아들이기 위해 S0, 즉 대기 상태로 넘어가게 된다.

S2 상태까지 진행되면 사용자가 의도한 정상적인 입력으로 인식하고 실제 키보드 이벤트를 발생시켜 응용프로그램에 방향키 입력이 들어가게 하며 S0를 제외한 각각의 단계에서 일정 시간 동안 상태가 유지되는 조건이나 다음 단계로 넘어갈 조건을 만족시키지 못하는 경우에는 S0 상태로 돌아가도록 하였다.

일반적으로 사용자가 방향 입력 동작을 시작할 때보다 멈추고 원위치로 돌아올 때 가속도의 크기가 더 크므로 임계값 T1보다는 T2의 크기를 크게 하여 보다 분명하게 입력을 받아들이도록 하였다. 임계값 T1과 T2의 크기는 실제 사용성 평가를 통해 중력가속도의 0.2배 정도로 결정되었으나 보다 쉽고 정확한 동작 인식을 위해서는 아직 약간의 조정이 필요한 것으로 생각한다. 혹은 사용자에 따라 동작에서 발생하는 가속도의 크기가 다를 수 있으므로 프로그램을 시작할 때에 사용자의 동작을 미리 받아들이고 여기에서 측정된 값에 따라 임계값을 조정하는 것도 좋을 것이다.

현재 개발된 프로그램은 키보드의 방향키를 입력했을 때와 같은 이벤트를 발생시키는 형태이므로 방향키 입력을 사용하는 어떠한 프로그램에도 사용할 수 있으며 방향키뿐만 아니라 마우스나 키보드의 다른 키 입력 이벤트를 발생시키는 것으로 변환도 가능하다.

2.4 진동 피드백

체험형 리모트 콘트롤러는 보다 풍부한 감각과 다양한 상호작용 방법을 제공하기 위해 진동모터를 사용하였다. 진동모터를 이용하면 사용자는 시각과 청각뿐만이 아니라 촉감을 통해서도 시스템의 정보를 얻을 수 있게 되고, 이를 통해 사용자가 보다 직관적이고 쉬운 방법으로 시스템의 현재 상황을 이해하고 상호작용할 수 있도록 할 수 있을 것이다. 또한, 보다 다양한 감각을 제공함으로써 사용자가 시스템에 보다 깊게 몰입하는데 도움을 줄 수 있다.

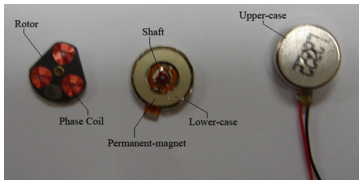


그림 13 <진동모터의 구조 (자화전자; JHV-12C1-L20)>

진동모터는 무게중심이 한쪽으로 치우친 회전자를 작은 직류모터를 이용해 회전시킴으로써 진동을 만들어내는 장치<그림 13>로서, 모터에 가하는 입력(전압)을 바꾸어 모터의 회전속도를 조절함으로써 진동강도를 조절한다. 즉, 높은 전압을 걸어주면 모터가 빠르게 회전하면서 큰 진동을 일으키고, 낮은 전압을 걸어주면 모터가 천천히 회전하면서 약한 진동을 일으키는 식이다. 때문에 진동모터는 진동모터는 진동주파수와 진동강도를 독립적으로 변화시킬 수 없다. 진동주파수는 낮지만 크기는 강한 진동이나, 즉, 주파수는 빠르지만 크기는 약한 진동은 만들어낼 수 없다. 아래 <그림 14>와 <그림 15>는 자화전자에서 만든 JHV-12C1-L20 진동모터의 입력 전압에 따른 진동주파수 변화와 진동강도 변화를 측정한 결과를 보여준다[5].

하지만 이러한 단점에도 불구하고, 크기가 작고 가격이 저렴하며 구조가 단순하므로 크기에 비해 상대적으로 강한 진동을 낼 수 있다는 장점 덕분에 크기와 사용시간이 중요하게 생각되는 모바일 기기 등에서 널리 쓰이고 있다.

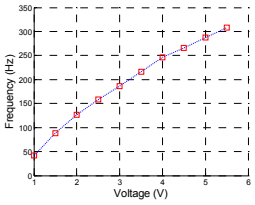


그림 14 <진동모터의 입력전압에 따른 진동주파수 변화>

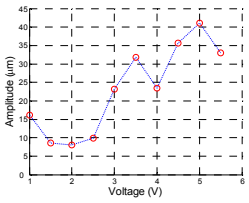


그림 15 <진동모터의 입력전압에 따른 진동강도 변화>

연구를 위해 제작된 체험형 리모트 콘트롤러의 프로토타입에는 2개의 진동 모터가 장착되었다. 두 개의 진동모터를 적절히 이용하면 스피커에서의 서라운드 효과와 같이 특정 방향에서의 진동이나 충돌 등을 표현할 수도 있을 것이다.

장착된 진동모터를 구동시키기 위해서는 구동시키기를 원하는 진동모터의 번호를 선택하고, 원하는 진동강도를 0에서 255 사이의 값으로 지정해주면 된다. 0은 정지상태를, 255는 최대진동상태를 나타낸다.

한 번 진동명령을 내리면, 다시 다른 진동명령을 내리기 전까지는 현재 상태를 계속 유지하며, 따라서 특정 진동을 일으킨 뒤에 진동모터를 멈추기 위해서는 원하는 진동 강도로 진동명령을 내리고 나서 일정 시간 뒤에 반드시 진동강도를 0으로 하여 다시 명령을 내려주어야 한다.

진동을 이용하기 위한 응용으로는 자동차 운전 시뮬레이션에서 자동차의 충돌효과를 예로 들 수 있다. 달리는 자동차가 건물의 벽 등에 부딪히면 달려오던 속도에 비례하여 진동을 주게 되고, 일정 시간이 지나면 자동차가 멈추면서 진동도 함께 멈추게 된다.

3. 동작기반 콘트롤러의 구현된 응용사례

3.1. 카 드라이빙 시뮬레이션

그림 16 <카 드라이빙 시뮬레이션에서의 동작인식 방법>

방향의 전환이나 가속, 감속에서 연속적인 크기를 갖는 값의 입력이 필요한 카 드라이빙 시뮬레이션에서는 2.5.1에서 설명한 기울기 감지를 이용하였다. <그림 16>에서와 같이 리모트 콘트롤러를 좌우로 기울임에 따라 차의 조향각을 좌우로 바꿔주도록 하였으며 기울인 각도가 클수록 조향각의 변화도 크도록 하였다. 리모트 콘트롤러를 앞으로 기울인 경우에는 자동차의 가속을, 뒤로 기울인 경우에는 감속의 효과가 나타나도록 하였으며 역시 기울인 각도가 클수록 그 효과가 더 크도록 하였다.

또한 사용자가 운전하는 차량이 다른 자동차나 건물에 부딪혔을 때에는 콘트롤러에 내장된 진동모터를 통해 충돌 당시 자동차의 속도에 비례하는 크기의 진동을 주고 진동의 크기는 시간에 따라 일정한 비율로 감쇠되도록 하였다.

3.2. 테트리스

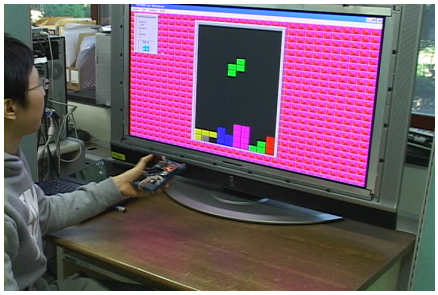


그림 17 <체험형 리모트 콘트롤러를 통한 테트리스 시연 장면>

테트리스를 하기 위해 필요한 입력은 좌우 이동, 떨어뜨리기와 회전 이렇게 총 네 가지가 필요하다. 이 입력에 대응되는 동작에는 가속도의 쓰레시홀딩에 기반한 스테이트 머신을 이용한 방식과 기울기 감지 방식을 적용할 수 있다.

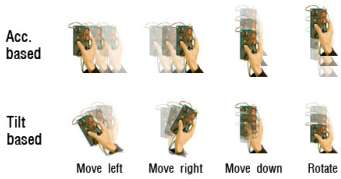


그림 18 <테트리스에서의 체험형 리모트 콘트롤러 조작법>

가속도의 쓰레시홀딩에 기반한 스테이트 머신을 이용한 방법에서는 리모트 콘트롤러를 상하좌우로 움직일 때 이동 방향으로 발생하는 가속도를 테트리스에 필요한 각각의 동작에 대응시켰다. 즉, 좌로 움직이면 좌로 이동, 우로 움직이면 우로 이동, 위로 움직이면 회전, 그리고 아래로 움직이면 떨어뜨리기를 수행하도록 할 수 있다.

기울기 감지 방식은 자동차 운전 시뮬레이터에서의 조작법과 비슷한 방식이다. 리모트 콘트롤러를 전후좌우로 기울이면 발생하는 경사의 정도를 가속도 센서에서 x, y, z 축 방향에 전해지는 중력가속도를 분석하여 측정할 수 있는데, 이를 위에서 가속도 기반 방식과 마찬가지로 각각의 동작에 대응시키면 리모트 콘트롤러를 좌로 기울이면 좌로 이동, 우로 기울이면 우로 이동, 뒤로 기울이면 회전, 앞으로 기울이면 떨어뜨리기를 수행하도록 할 수 있도록 하였다.

3.3. 권투 게임

그림 19 <동작인식을 이용한 권투 게임>

가속도의 쓰레시홀딩에 기반한 스테이트 머신을 이용한 동작인식 방법을 권투게임에 접목시켜 사람이 콘트롤러를 손에 쥐고 주먹을 앞으로 뻗을 때마다 화면에 나타나는 캐릭터에게 타격이 가해지도록 하였다. 또한 화면에 나타나는 캐릭터가 사용자에게 주먹을 날리면 콘트롤러에 장착된 진동모터를 통해 피드백을 주게 된다.

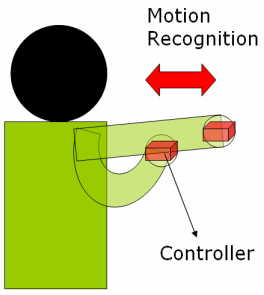


그림 20 <권투 게임에 적용된 동작인식 방법>

3.4. 탁구 게임

그림 21 <동작인식을 이용한 탁구게임>

가속도의 쓰레시홀딩과 기울기 정보를 모두 사용하여 스테이트없이 연속적인 동작을 판단하도록 하였다. 즉 콘트롤러를 잡고 스윙을 하거나 앞으로 기울이면 라켓이 앞으로(반시계방향) 돌아가고 뒤로 기울이면 뒤로(시계방향으로)돌아가 원위치가 된다.

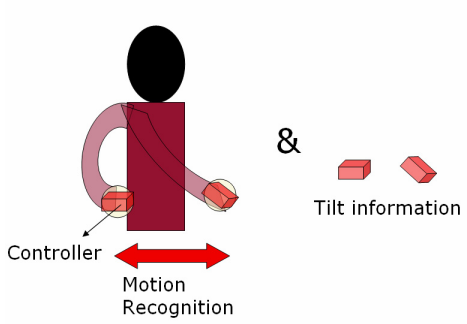


그림 22 <탁구 게임에 적용된 동작인식 방법>

4. 결론

이 연구를 통해 사용자가 기존의 단순한 버튼 입력 대신에 보다 직관적이고 친숙한 동작을 입력에 활용하도록 하여 사용자의 조작감과 몰입감을 높일 수 있도록 하였다. 또한 버튼과 동작을 조합하는 것으로 리모콘의 제한된 인터페이스에서 보다 다양한 입력을 가능케 하였으며 상황에 알맞은 햅틱 피드백을 사용자에게 전달함으로써 보다 다양한 감각기관을 활용한 양방향의 체험을 할 수 있도록 하였다. 앞으로 기존의 버튼 인터페이스와 비교한 사용성 평가가 있어야 할 것이다.

또한 가속도계를 이용한 위치 추적 방법과, 두 가지의 동작 인식 방법을 제시하였으며 앞으로 보다 정확한 동작인식과 위치 추적을 위해서는 지금의 가속도 센서만이 아닌 적외선 센서나 초음파 센서 등 추가적인 센서를 사용한 복합적인 알고리즘으로의 개선이 필요할 것이다.

감사의 글

본 연구는 KT-POSTECH 공동연구과제 지원에 의해 수행되었음.

참고문헌

[1] Ftw's project MONA webpage. http://mona.ftw.at

[2] A. Henrysson, M. Billinghurst, M. Ollila, "Face to Face Collaborative AR on Mobile Phones." In Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR 2005), 2005.

[3] P. Föckler, T. Zeidler, and O. Bimber, "Phoneguide: Museum guidance supported by on-device object recognition on mobile phones." Proceedings of the 4th international conference on Mobile and ubiquitous multimedia, 2005.

[4] C. Jay and R. Hubbold, "Delayed Visual and Haptic Feedback in a Reciprocal Tapping Task," IEEE Computer Society, 2005.

[5] 정재훈, "동작기반 리모트 컨트롤러 및 인식강도와 전력소비에 최적화된 모바일기기의 진동설계에 관한 연구", 석사학위논문, 포항공과대학교, 2007