안드로이드 모바일 기기를 이용한 터치패드 구현

이민구[0] 김슬기 박신애 한경숙 오용철
한국산업기술대학교 컴퓨터공학과
{pepsi815815[0], ksk0910, midrang, khan, oh}@kpu.ac.kr

Implementation of Touchpad by using Android Mobile Device

Min-Goo Lee[0], Seul-Ki Kim, Shin-Ae Park, Kyung-Sook Han, Yong-Chul Oh
Dept. of Computer Engineering, Korea Polytechnic University

요 약

국내 스마트폰 시장은 사용자 수 2000만 명 시대를 향해가고 있다. 사람들은 PC를 사용했던 것과 같이 모바일 기기도 최대한 활용하고자 노력하고 있다. 본 논문에서는 안드로이드 기반의 모바일 기기를 PC와 연결된 터치패드로 활용할 수 있도록 하는 것을 목표로 한다. 이는 기존 터치패드가 가지는 휴대성 부분의 단점을 개선할 방안이 된다. 또한 터치패드 구현에서 그치는 것이 아니라 불안정한 네트워크 환경에서도 효과적으로 사용할 수 있도록 연구를 진행하였다.

1. 서 론

스마트폰의 보급률이 높아지면서 사람들은 이 기기를 최대한 활용하기 위해 많은 노력을 기울여왔다. 실생활의 대부분이 어플리케이션이라는 영역에 적용되었고, 모바일 기기는 일상과 더욱 밀접해졌다. 모바일 기기는 휴대전화의 성격을 그대로 흡수했기에 사용자가 항상 휴대하고 다니는 성격을 가지고 있는 것은 물론 네트워크 사용이 용이하므로 지역에 구애없이 컴퓨터와의 통신이 용이하다.

터치패드는 대부분 노트북에 내장되어있거나 별도의 외장형 기기의 형태로 사용되고 있는데 노트북에 내장된 터치패드는 키보드 아래에 위치하여 사용하기가 불편하며 별도 사용을 위한 위치 이동은 불가능하다. 터치패드 기능만을 위해서 외장형 기기를 사용할 수도 있겠지만 추가적인 비용이 발생한다.

본 논문에서는 이러한 장단점을 고려하여 휴대용 단말기를 이용하여 사용자가 휴대하기 편리하면서 데이터 전송 또한 안정적으로 이루어질 수 있는 터치패드의 구현을 목표로 한다.

2. 관련 연구

기존에 모바일폰에 구현되었던 터치패드로 다음과 같은 것이 있다.

Padroid[1]는 별도의 서버 프로그램을 두지 않고, 사용자가 직접 모바일 기기의 IP와 포트번호를 입력하여 접속하도록 하였다. 띄워진 웹 화면에 있는 파란색 전원버튼을 누르면 바로 모바일 기기를 터치패드로 이용할 수 있다. 터치패드 외의 다른 기능은 제공하지 않으며 어플리케이션을 실행하면 검은 화면이 나타난다. 화면 위에서 사용자가 원하는 방향으로 손가락을 움직이면 PC의 마우스가 움직인다.

gPad[2]는 Padroid보다는 좀 더 발전한 형태로, 서버 프로그램을 통해 사용자가 좀 더 편리하게 접속할 수 있도록 하였다. 어플리케이션 또한 Padroid의 빈 화면보다는 사용자를 고려한 모습을 가진다. 버튼과 휠의 기능을 이용할 수 있다.

LIGHTENING社의 LP100[3]은 평소에는 무선마우스로, 프레젠테이션 때는 프리젠터와 레이저포인터로 사용할 수 있도록 만들어진 제품이다. 프레젠테이션을 하기 위해서 별도의 마우스와 프리젠터, 레이저포인터를 각각 휴대할 필요없이 LP100만으로 가능하게 해주는 제품이다.

이처럼 각기 다른 역할을 하던 여러 개의 제품이 점차 하나의 제품으로 합쳐지는 것이 요즘의 추세인 만큼, 모바일 기기를 무선 터치패드로 활용할 수 있도록 하는 본 연구의 잠재수요는 크다고 볼 수 있다.

기존의 제품을 살펴본 결과, 모두 같은 AP 상에서 PC와 모바일 기기를 각각 서버와 클라이언트로 이용하는 방식이다. Padroid는 컴퓨터에서 모바일 기기로, gPad는 모바일 기기에서 컴퓨터로 접속했지만 어느 한 기기에서 다른 기기로 접속함으로써 데이터를 주고받는 환경을 만든 것은 동일하였다.

본 논문에서 제안하는 시스템 또한 두 제품과 마찬가지로, 동일 AP에서 PC와 모바일 기기를 연결하여 PC는 서버, 모바일 기기는 클라이언트 역할을 한다. 사용자가 직접 IP와 포트번호를 입력하도록 하는 방식은 간단하다는 장점이 있으나 네트워크 환경에 익숙하지 않은 사용자라면 모바일 기기의 IP를 알아내는 것이 어려울 수 있다. 때문에 gPad와 같이 별도의 PC 어플리케이션을 두어 사용자의 입력 없이도 손쉬운 연결이 가능하도록 하였다. Padroid, gPad와는 다르게 부가적으로 PC 내의 멀티미디어 파일을 모바일 기기에서 재생하는 멀티미디어 리모콘을 구현하고자 한다.

두 제품 모두 프로그램 자체의 실행에는 문제가 없었으나 네트워크 환경에 따라 딜레이 현상이 나타났다. 본 연구를 진행하는 과정에서도 이는 큰 문제가 되었으므로 딜레이 현상을 최소화하기 위한 알고리즘을 연구, 시스템에 적용하였다.

3. 시스템 구성

3.1 개발환경

클라이언트 어플리케이션에서의 프레임은 Android 2.2 Froyo 버전을 기본으로 Android 2.2 SDK v.2.2 Windows Platform과 JDK6 SE 1.8을 사용하였다[4].

서버에서는 자바 플랫폼 SE6의 API를 사용하였고, 터치패드라는 특성상 마우스 제어 권한을 얻어야하므로 Robot 클래스를 뼈대로 구성하였다[5].

그림 1은 시스템 구성도를 나타낸다. PC와 모바일 기기가 같은 무선네트워크 상에 연결되어 있으면 데이터의 송수신이 가능해진다.

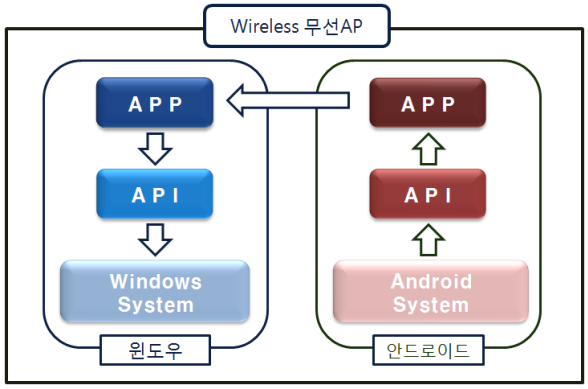

그림 1. 시스템 구성도

모바일 기기에 터치이벤트가 전달되면, API는 시스템에서 좌표 데이터를 받아 어플리케이션에 전달한다. 어플리케이션은 일련의 과정을 거쳐 데이터를 가공하고, 송신한다. PC는 가공된 데이터를 받아 다시 한 번 PC 환경에 맞게 데이터를 가공하여 시스템으로 전달한다.

이 과정에 필요한 주요 API는 GestureDetector 클래스와 MotionEvent 클래스이다[6]. GestureDetector 클래스의 gestureListener는 사용자의 동작을 기다렸다가 움직임이 일어나면 움직임의 종류를 구분한다. 한 번 클릭이 일어났는지, 두 번 클릭이 일어났는지 또는 길게 눌러졌는지 등을 구분하여 클릭, 더블클릭, 오른쪽 클릭 등의 제스처 구분 값을 PC에 전달하는 것이다. 이와 같은 제스처들은 앞서 언급한 Robot 클래스를 통해 마우스 제어로 이어진다.

제스처를 제외한 움직임일 경우 사용자 손가락의 움직임에 따라 마우스를 이동시켜야 한다. 이를 위해 API로부터 모바일 기기의 좌표데이터를 받아 움직임을 파악하고 경로대로 PC 마우스의 좌표를 변환한다. 이 때 단순히 좌표의 증가량만을 적용시킬 수는 없다. 외장형 터치패드의 경우 크기가 크므로 터치패드의 모서리와 PC화면의 모서리를 맞추는 것이 가능하지만 일반적인 스마트폰의 화면은 PC에 비해 차이가 크기 때문이다. 따라서 PC는 모바일 기기로부터 좌표의 증가량을 전달받은 후 마우스의 현재 좌표를 얻어 연산, 적용한다.

3.2 주요 알고리즘

본 연구는 네트워크 환경에 큰 영향을 받지 않도록 하여 결과적으로 딜레이를 최소화하는 것에 중점을 두었다. 딜레이 발생의 가장 큰 원인이 불안정한 네트워크이므로 안정적인 네트워크를 제공받음으로써 해결할 수 있지만, 모든 사용자가 안정적인 네트워크를 누릴 수 있는 것은 아니다. 따라서 본 연구에는 이 문제의 해결방안으로 두 가지를 적용하였다.

첫째, 데이터 전송을 최소화하였다. PC와 모바일 기기는 연결된 상태이므로 데이터 전송 후 좌표 값을 변환할 수도 있고, 변환한 후에 전송할 수도 있다. 그러나 네트워크가 불안정하여 데이터 전송이 제대로 이루어지지 않으면 그 후 과정인 변환도 불가능하므로 모든 데이터의 연산이 완료된 후의 최종 결과 값만 전송하도록 설계하였다.

둘째, 화면을 구역화하였다. 마우스의 미세한 변화에 대한 좌표데이터를 배제하고 이동 후 좌표 값만을 PC에 송신하는 방법이다.

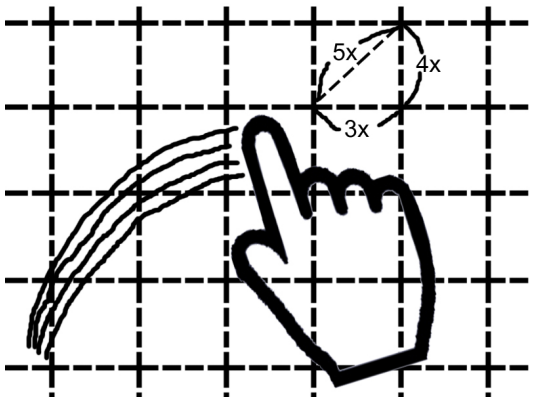

그림 2. 화면의 구역화

그림 2와 같이 화면을 구역으로 나누고 피타고라스 정리를 이용하여 대각선 길이를 계산한다. 스마트폰의 화면은 대부분 가로 : 세로 = 3 : 5의 비율을 가지고 있으나 가로 : 세로 : 대각선 = 3 : 4 : 5 비율로 나누면 자연수 연산이 가능하다. 위아래 오차범위를 제외한 정상범위 구역은 이차원배열 형식으로 번호를 매긴 후 구역을 벗어난 좌표 이동이 있었을 경우에만 변화량을 계산하여 전송하도록 하였다. PC는 모바일 기기로부터 가공된 데이터를 받아 현재 좌표로부터 증가량만큼 움직이도록 한다. 이는 중간과정에서 발생하는 데이터들을 저장하지 않기 때문에 메모리를 효율적으로 사용하는 효과를 거둘 수 있다.

의사코드(pseudocode)는 다음과 같다.

```
설정()
{
  가로 = 화면 가로 픽셀 읽어오기()
  세로 = 화면 세로 픽셀 읽어오기()
  밀도 = 화면 픽셀 밀도 읽어오기()
  구역화 비율 계산(가로, 세로, 밀도)
  {
    비율 = 밀도÷80
    가로비 = 3×비율
    세로비 = 4×비율
    대각선비 = 5×비율
  }
}
```

```
터치이벤트 처리()
{
  한 번 짧은 터치이면
    클릭 제스처 구분 값 전송()
  두 번 짧은 터치이면
    더블클릭 제스처 구분 값 전송()
  한 번 긴 터치이면
    오른쪽클릭 제스처 구분 값 전송()
  구역 이동 시 // 이차원 배열(P[n][n])의 n값 확인
  {
    P[n+1][n]이면
      +가로비와 가로 구분 값 전송()
    P[n-1][n]이면
      -가로비와 가로 구분 값 전송()
    P[n][n+1]이면
      +세로비와 세로 구분 값 전송()
    P[n][n-1]이면
      -세로비와 세로 구분 값 전송()
    P[n+1][n+1]이거나 P[n-1][n-1]이면
      +대각선비와 대각선 구분 값 전송()
    P[n+1][n-1]이거나 P[n-1][n+1]이면
      -대각선비와 대각선 구분 값 전송()
  }
}
```

4. 구현 결과

본 연구에서는 기본적인 마우스 기능이 가능한 터치패드를 구현하였다. 이를 통해 모바일 기기의 화면 위에서 사용자가 움직이는 대로 연결된 PC의 마우스포인터가 움직이게 할 수 있다. 두 기기가 무선으로 연결되어 있으므로 무선 마우스뿐만 아니라 프리젠터로도 활용이 가능하다. 마우스 기능을 통해 멀티미디어를 제어할 수 있으므로 이러한 점을 활용하고자 추가 기능으로 멀티미디어 리모콘을 구현하였다.

멀티미디어 리모콘은 PC내에 있는 음악이나 비디오 등의 멀티미디어 파일을 제어하는 기능이다. PC에 있는 파일을 모바일로 전송하는 것이 아니라, 연결된 PC의 파일목록에 모바일에서 접근한 후, 파일을 찾아 재생 등의 기능을 하도록 하는 것이다. 파일전송이 일어나는 것이 아니므로 동작은 PC에서 이루어진다. 재생, 일시정지, 정지, 음소거, 음량조절 등의 기능이 있으며 사용자에게 익숙한 직관적인 UI를 통해 손쉬운 사용이 가능하도록 하였다.

따라서 PC와 모바일 기기가 동일 네트워크에 연결되어 있기만 한다면, 거리에 상관없이 마우스와 리모콘 역할을 수행하는 것이 가능하다.

또한 딜레이 현상을 최소화하기 위해 연구를 진행하고 알고리즘을 적용하였다. 스마트폰에서는 픽셀이 곧 화면의 좌표이므로 보다 정밀한 감도로 작업이 가능하다[7]. 그러나 앞서 언급한 바와 같이 마우스작업에서 픽셀 단위의 좌표는 배제할 수 있다. 이처럼 픽셀 하나에 대한 데이터를 모두 전송하는 것 대신 가로×세로 넓이, 즉 약 100픽셀을 대표하는 하나의 데이터만 전송하도록 함으로써 데이터 전송의 최소화를 이루었다.

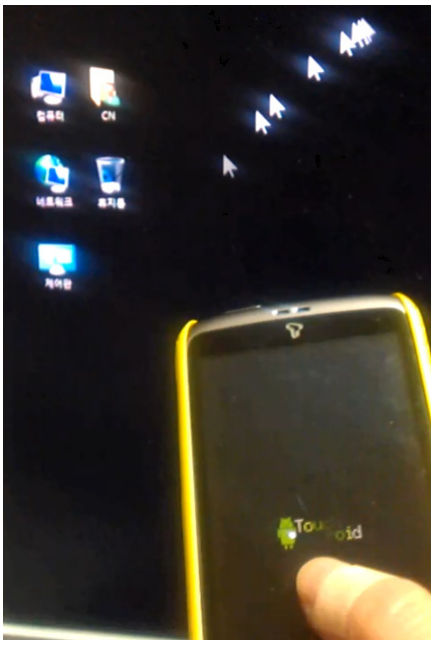

그림 3 구현결과

5. 결론 및 고찰

본 연구에서는 안드로이드 기반의 모바일 기기를 터치패드로 활용할 수 있도록 하는 어플리케이션을 구현하였다. 이를 통해 스마트폰 사용자는 추가비용을 들이지 않고도 간편히 휴대할 수 있는 터치패드를 얻을 수 있다. 터치패드는 무선 마우스 기능을 수행할 수 있으며, 프리젠터로 활용이 가능하다. PC 내부 파일목록에 접근하여 멀티미디어를 제어하는 멀티미디어 리모콘 기능도 갖추었다.

뿐만 아니라 화면을 구역으로 나누는 알고리즘을 연구하여 적용함으로써 불안정한 네트워크 환경에서도 딜레이를 최소화하기 위한 방안을 제안하였다. 딜레이가 발생하는 원인을 분석하였고, 환경 자체를 개선할 수 있도록 하였다.

추후 PC내에서 이동 직전 좌표와 이동 직후 좌표의 증가량과 소요시간을 함께 측정하는 기술을 추가하면 딜레이현상의 감소를 수치화할 수 있을 것이다.

참고문헌

[1] http://software2tech.com/tag/android-padroid/

[2] http://www.gpad.mobi/content/en/home

[3] http://www.hankyung.com/news/app/newsview.php?aid=2010070844951

[4] http://developer.android.com/sdk/android-2.2.html

[5] http://airyym.tistory.com/118

[6] http://download.oracle.com/javase/6/docs/api/

[7] http://developer.android.com/guide/practices/screens_support.html

[8] 마크 머피, 「알짜만 골라 배우는 안드로이드 프로그래밍」, 에이콘, 2009

[9] 에드 버넷, 「헬로, 안드로이드 2.2」, ITC, 2010