

안드로이드 모바일 플랫폼에서의 동적 해상도 자동전환을 위한 프레임버퍼 구조 개발

(Development of Frame Buffer Structure for Automatic Dynamic Resolution Switching on Android Mobile Platform)

김 주 성 * 권 오 철 †
(Jusung Kim) (Oh-Chul Kwon)

이 창 건 **
(Chang-Gun Lee)

요 약 최근 모바일 장치의 기능이 발전하면서, 스마트폰에 대한 관심이 증가하고 있다. 스마트폰은 휴대성이라는 장점을 가지고 있음에도 불구하고, 그 기기의 크기가 작아 사용자 인터페이스 관점에서 불편함이 많이 있었다. 본 논문에서는 이 문제점을 해결하기 위해 안드로이드 모바일 플랫폼에서의 프레임버퍼 구조를 수정하여, 동적으로 해상도를 전환시켜 크기가 큰 외부 디스플레이 장치로 디스플레이를 전환하고, 기존의 휴대폰은 리모콘으로 활용할 수 있는 사용자 인터페이스 전환 기술을 개발하였다.

키워드 : 해상도 자동전환, 안드로이드, 스마트폰, 프레임버퍼

- 이 연구를 위해 연구장비를 지원하고 공간을 제공한 서울대학교 컴퓨터 연구소에 감사드립니다.
- 본 연구는 지식경제부 및 정보통신산업진흥원의 대학 IT연구센터 지원 사업의 연구결과로 수행되었음(NIPA-2010-C1090-1011-0008)
- 이 논문은 2010 한국컴퓨터종합학술대회에서 '안드로이드 모바일 플랫폼에서의 동적 해상도 자동전환을 위한 프레임버퍼 구조 개발'의 제목으로 발표된 논문을 확장한 것임

* 학생회원 : 서울대학교 컴퓨터공학부
jskim@rubis.snu.ac.kr
ockwon@rubis.snu.ac.kr

** 정 회 원 : 서울대학교 컴퓨터공학부 교수
cglee@snu.ac.kr

논문접수 : 2010년 8월 10일
심사완료 : 2010년 10월 27일

Copyright©2010 한국정보과학회 : 개인 목적이거나 교육 목적인 경우, 이 저작물의 전체 또는 일부에 대한 복사본 혹은 디지털 사본의 제작을 허가합니다. 이 때, 사본은 상업적 수단으로 사용할 수 없으며 첫 페이지에 본 문구와 출처를 반드시 명시해야 합니다. 이 외의 목적으로 복제, 배포, 출판, 전송 등 모든 유형의 사용행위를 하는 경우에 대하여는 사전에 허가를 얻고 비용을 지불해야 합니다.

정보과학회논문지: 컴퓨터의 설계 및 테더 제16권 제12호(2010.12)

Abstract This paper presents a technique for an Android mobile device to best utilize its surrounding external display device such as high-definition TV, in order to overcome the size and resolution limitation of the small LCD in the mobile device. Unlike current techniques that simply expand the LCD screen image to an external large display, our technique dynamically changes the resolution adapting to capability of the external display. And, the original screen become a remote controller.

Key words : Resolution Switching, Android, Smart Phone, Framebuffer

1. 서 론

최근에 모바일 프로세서와 무선 인터넷 기술이 점점 발전하면서, 웹 서핑, 영화 감상, 문서 편집 등과 같은 다양한 작업들을 스마트폰에서 하는 것이 가능해 지게 되었다. 기존에는 집에서 PC를 통해서 밖에 할 수 없었던 작업들을 언제 어디서나 휴대용 스마트폰을 이용하여서 할 수 있게 된 것이다.

이렇게 스마트폰이 휴대성이라는 장점을 바탕으로 빠르게 발전하고 있음에도 불구하고, 이러한 휴대성 때문에 UI 관점에서는 많은 문제점을 나타내고 있다. 휴대성을 높이기 위해서는 휴대폰의 크기는 작아져야 하는데, 그렇게 되면 휴대폰의 디스플레이 장치나 입력장치 등의 크기도 작아질 수밖에 없다. 이러한 스마트폰의 한계 때문에 사용자들은 많은 불편함을 감수할 수밖에 없었다. 이러한 단점이 해결되지 않는다면 스마트폰은 PC의 기능을 완전히 대체할 수 없을 뿐만 아니라, 스마트폰의 발전에는 한계가 생길 수밖에 없다.

이런 문제점이 생기면서 HDTV, 키보드 등과 같은 외부 사용자 인터페이스 장치를 휴대폰에서 사용하려는 노력이 계속 되고 있다. 본 연구에서는 디스플레이 장치에 초점을 맞추고 HDTV 등으로 스마트폰의 디스플레이를 전환하는 기술을 제안한다.

기존에도 외부 디스플레이 장치로 스마트폰 등의 모바일 장치의 화면을 전송하는 기술이 존재하였으나 이러한 기술들은 단순히 TV-OUT을 통해서 스마트폰의 화면을 전송하는 등의 방법을 사용하였다. 하지만 이러한 방법은 유선으로 전송하는 방법이기 때문에 스마트폰의 휴대성을 최대한 살릴 수 없다. 뿐만 아니라 TV-OUT 등의 기술은 스마트폰의 화면을 그대로 전송하여 크기만 키운 것이기 때문에 화면의 해상도는 변하지 않는다. 그렇게 되면 한 화면에 나타나는 정보의 양은 변하지 않기 때문에 스마트폰의 문제점을 해결하는 데에는 무리가 있다.

본 연구에서는, 안드로이드의 프레임버퍼 구조를 개선하여 스마트폰의 해상도를 동적으로 변환시켜 화면을

Wi-Fi를 통해 외부 디스플레이 장치로 전송시키고, 기존의 스마트폰의 LCD는 입력장치로 활용할 수 있는 방법을 제안하고자 한다.

본 논문은 다음과 같은 순서로 기술된다. 먼저 2장에서는 관련 연구에 대해 기술 된다. 3장은 기존의 안드로이드 플랫폼의 해상도와, 프레임버퍼 관리 구조에 대해 기술된다. 4장에서는 기존의 안드로이드 플랫폼 구조를 수정하는 것을 제안하고, 5장에서는 실제로 구현한 것이 기술된다. 마지막으로 6장에서는 본 논문의 결론이 기술 된다.

2. 관련 연구

서론에서 언급한 것과 같이 기존에도 이러한 UI의 한계점을 극복하기 위해서 외부의 다른 UI 장치와 스마트폰을 연결하려는 노력이 많이 있었다.

[1,2]에서는 모바일 장치를 TV나 PC의 리모콘으로 사용할 수 있는 가능성에 대해 언급하고 있다. 실제로 [3,4]와 같이 PC의 인터넷 브라우저나 e-mail 등의 특정 프로그램에 한해, PDA를 통해 PC를 제어하는 연구도 진행 되었다. 그리고 안드로이드의 경우 "Gmote"[5]라는 PC에 들어있는 영상 파일을 스마트폰을 통하여 재생하고 제어할 수 있는 어플리케이션이 출시되기도 하였다. 이러한 연구들은 단지 모바일 장치를 리모콘으로 사용하는데 제한되기 때문에 스마트폰의 모든 기능을 사용한다고 보기에는 무리가 있다.

그리고 많은 연구에서 스마트폰의 화면을 외부의 큰 디스플레이 장치로 전환하는 기술을 제안하였다. 노키아

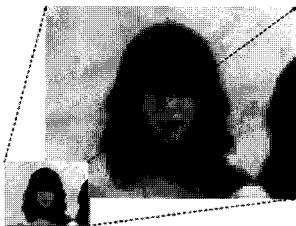


그림 1 단순 화면 확대



그림 2 해상도 전환

에서는 TV, 프로젝터와 같은 큰 장치로 스마트폰의 화면을 전송하는 smart docking station[6]을 만들었다. 그리고 애플의 iPhone[7]과 삼성의 Instinct HD Phone [8]에서는 TV-OUT을 통해 화면을 전송할 수 있게 해주었다. 이러한 기술은 그림 1과 같이 것처럼 화면을 단순히 키우는데 불과하다. 그리고 [9]에서는 제한적으로 해상도를 전환시켜 TV로 전송시키는 기술을 제안하고 있으나, 이 연구에서는 스마트폰의 LCD보다 큰 해상도로 전환시킬 수 없을 뿐만 아니라 해상도를 전환시킨 후에는 기존의 스마트폰의 입력장치를 활용할 수 없어 입력에 큰 문제를 나타내고 있다.

이러한 기술들과 달리 본 연구에서는 그림 2와 같이 해상도를 변환시켜 TV로 스마트폰의 화면을 전송하고, 기존의 스마트폰의 LCD는 리모콘 등의 역할을 할 수 있는 입력장치로 활용하는 기술을 제안하고자 한다.

3. 기존의 안드로이드 플랫폼 구조

그림 3은 안드로이드의 구조를 나타낸 그림이다. 이를 바탕으로 이 장에서는 안드로이드에서 해상도와 프레임 버퍼를 관리하는 방법이 기술된다.

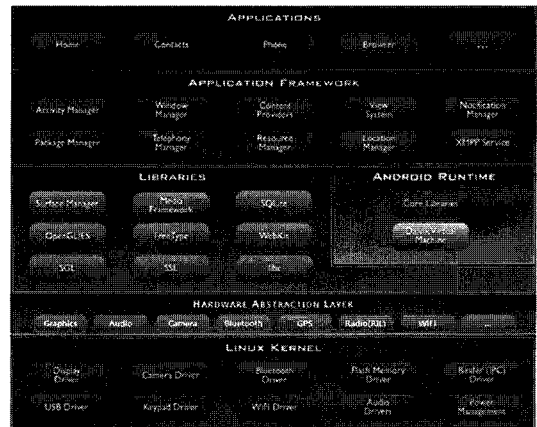


그림 3 안드로이드의 구조

3.1 안드로이드 해상도 관리

기존의 Android의 해상도 관리 방법에 대해 설명 하던 다음과 같다. 우선 부팅시 Linux Kernel Layer 에서는 디스플레이 드라이버를 통해 LCD 장치의 해상도 정보를 읽어온다. Hardware Abstraction Layer(HAL)에서는 그 정보를 SurfaceComposerClient Class에 저장하여 관리한다.

Android의 Application 들은 SurfaceComposerClient Class의 getWidth나 getHeight 등의 함수를 통해 해상도 정보를 가져가게 된다. 각 Application은 얻어진 해

상도 정보를 이용해 Layout을 정하며, 그에 따라 프레임버퍼에 화면을 그리게 된다.

3.2 안드로이드 프레임버퍼 관리

Linux Kernel Layer는 LCD 장치의 해상도 정보를 읽어올 때, 해상도에 맞도록 프레임버퍼의 메모리를 할당하게 된다. 그 때 할당되는 메모리의 크기는 (1)을 통해 구할 수 있다. 이 수식에서 xres는 x축 해상도, yres는 y축 해상도, bpp(bits per pixel)는 픽셀 포맷을 나타낸다. 한 화면을 나타내는 한 프레임의 크기는 xres*yres*(bpp/8)로 나타낼 수 있으나 안드로이드는 기본적으로 더블버퍼링을 사용하기 때문에 프레임 버퍼의 사이즈는 2배가 된다.

$$size = xres * yres * (bpp/8) * 2 \text{ byte} \quad (1)$$

안드로이드의 디스플레이 드라이버는 이렇게 할당된 메모리에서 어플리케이션들이 그려낸 화면의 정보를 LCD의 레지스터로 메모리를 복사하여서 LCD에 화면을 출력하도록 한다. 그림 4는 안드로이드의 기존 프레임버퍼 구조를 나타낸 것이다. 안드로이드는 더블버퍼링을 사용하고 있기 때문에, 어플리케이션이 Buffer1을 쓰고 있으면 LCD Display에는 Buffer2가 출력되고(실선), Buffer2를 쓰고 있으면 LCD에는 Buffer1이 출력되게 된다. (점선)

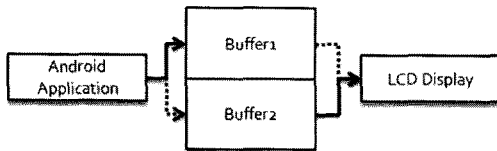


그림 4 안드로이드 프레임버퍼 구조

4. 새로운 안드로이드 플랫폼 구조 제안

이러한 기존의 안드로이드 플랫폼 구조에서는, 동적으로 해상도를 변경시킬 수 없다. 그렇기 때문에 아래와 같이 해상도와 프레임버퍼를 관리하는 것을 제안한다.

4.1 해상도 전환 모듈 구현

기존의 안드로이드는, SurfaceComposerClient가 드라이버에서 해상도 정보를 읽어와 저장한 후, 그것을 수정시킬 수 있는 메커니즘이 존재하지 않았다. 안드로이드의 어플리케이션들은, 해상도를 참조하여서 화면을 생성하기 때문에, 어플리케이션들이 참조하는 해상도의 값만 동적으로 변하게 된다면, 동적으로 해상도가 변하는 것이 가능해질 것이다.

이에 착안하여서, 어플리케이션이 부팅될 때 값이 설정이 되는 SurfaceComposerClient의 값을 바로 참조하는 것이 아니고, 해상도 전환 모듈을 만들어 여기에서 제공하는 해상도의 값을 참조한다면 동적으로 해상도를

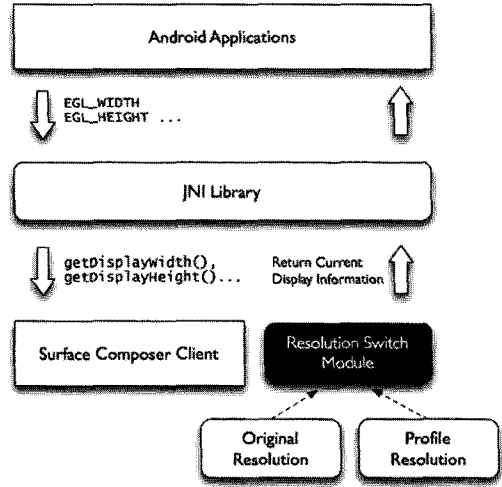


그림 5 해상도 전환 모듈

변하게 할 수 있을 것이다. 해상도 전환 모듈은 일반적인 경우에는 SurfaceComposerClient의 값을 리턴하고, 만약에 HDTV와 같은 외부 UI 장치로 전환을 할 때에는 외부 Device의 해상도를 리턴을 하도록 구현이 될 것이다. 그림 5는 해상도 전환 모듈의 구조를 나타낸 것이다.

4.2 프레임버퍼 관리 구조 개선

기존의 프레임버퍼 구조를 그대로 동적 해상도 전환에 사용한다면, 다음과 같은 문제점을 지니게 된다. 첫째, 부팅 시에 LCD의 해상도의 정보를 이용하여 메모리를 할당하게 되면 LCD보다 큰 해상도의 정보를 저장할 수 없게 된다. 둘째로 프레임버퍼의 내용을 스마트폰의 LCD에 복사하여서 출력되게 되는데, 해상도가 LCD보다 커지게 되면 LCD는 화면의 일부밖에 보일 수 없게 되어 외부의 디스플레이 장치로 화면을 전환하게 되면 스마트폰의 LCD는 더 이상 사용할 수 없게 된다. 마지막으로 외부 디스플레이 장치로 화면을 전환하게 되면, 출력의 주체는 외부 디스플레이 장치가 되는데 스마트폰에서 더블버퍼링을 하는 것은 메모리를 낭비하게 된다. 이런 문제점 들을 개선하기 위해 다음과 같은 방법을 제시한다.

첫째, 처음에 프레임버퍼 메모리를 할당할 때 LCD 만큼의 메모리를 할당하는 것이 아니고, 추후에 외부 장치로 디스플레이가 전환될 것을 고려하여서 메모리를 크게 할당한다. 안드로이드에서는 최대 2Mbyte의 프레임버퍼 사이즈를 제공하기 때문에, 부팅될 때 2Mbyte의 메모리를 미리 할당을 한다면, 첫 번째 문제는 해결할 수 있을 것이다.

둘째, 그림 6과 같이 프레임 버퍼를 외부 디스플레이

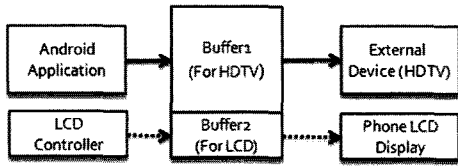


그림 6 제안하는 프레임 버퍼 구조

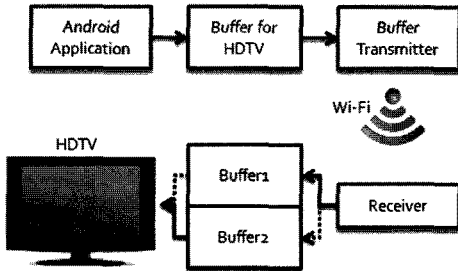


그림 7 HDTV에서의 더블버퍼링

장치로 전송할 부분과 스마트폰의 LCD에 복사할 부분, 2개로 나누어서 관리를 한다. 안드로이드 어플리케이션은 외부 디스플레이 장치로 전송할 영역에 데이터를 쓰게 되고, LCD에 출력할 부분은 새로운 컨트롤러가 데이터를 쓰게 하는 것이다. 그렇게 하면 외부 장치에 출력되는 화면과, 스마트폰의 화면을 분리할 수 있어 스마트폰을 리모콘 등으로 사용하는 것이 가능하게 된다.

마지막으로 그림 7과 같이 스마트폰 내에서는 더블버퍼링을 사용하지 않고, HDTV의 수신부에서 더블버퍼링을 사용한다. 이렇게 사용한다면 HDTV보다 상대적으로 메모리가 작은 모바일 장치의 메모리를 효율적으로 사용할 수 있게 된다. 안드로이드 휴대폰에서는 HDTV로 전송될 버퍼 한 개만 두고, 안드로이드 어플리케이션은 이 버퍼에 내용을 쓰게 되고 이 버퍼를 HDTV로 전송을 하게 된다. 이 때 버퍼가 전송되는 동안에는 버퍼에 Lock을 걸어서 내용이 쓰여지는 동안 전송이 되지 않도록 구현하였다. 이 때 성능의 저하가 생길 수 있으나, 대부분의 성능 이슈는 Wi-Fi를 통한 전송에서 생기기 때문에 싱글버퍼링으로 인한 성능 저하는 매우 미미하다.

5. 구현

위에서 제시한 구조를 따라, 실제로 안드로이드 플랫폼을 수정하여서 아래와 같은 기능들을 구현하였다.

- Wi-Fi를 통해 주변의 디스플레이 장치 검색
- 검색된 장치의 해상도에 맞게 동적 해상도 전환
- 프레임 버퍼를 캡처하여 디스플레이 장치로 전송
- 기존의 스마트폰은 리모콘 등으로 활용할 수 있도록 LCD Controller 구현

표 1 구현 환경

Device	Specification
Target Board	Board : Odroid CPU : SAMSUNG S5PC100 (833Mhz) Memory : 512MB Mobile DDR2 Display : 3.5" LCD (320x480) Network : WiFi 802.11b
Target UI Device	CPU : Intel Core2 Duo E7200 2.53Ghz Memory : 2.00GB RAM OS : Microsoft Windows 7 Network : Realtek RTL8168/8111 PCI-E Gigabit Ethernet NIC (100Mbps)

구현은 표 1의 환경에서 하였다. 외부 디스플레이 장치는 HDTV를 사용하지 않고, PC로 HDTV의 환경을 가상하여서 구현하였다.

5.1 디스플레이 장치 구현

주변의 장치를 Wi-Fi를 통해 검색하고 해상도의 정보를 가져오는 안드로이드 어플리케이션을 제작하였다. 그리고 검색된 해상도 정보를 4.1절에서 언급한 해상도 전환 모듈에 전달해 주게 되면 해상도가 바뀌게 된다. 그리고 4.2절에서 언급한 대로 프레임 버퍼를 설계하고, HDTV의 버퍼를 Wi-Fi를 통해 전송시켜 주었다.

그림 8은 주변의 1024x768의 해상도를 가지는 Samsung LCD Monitor를 어플리케이션이 찾아낸 사진이다. 그리고 그림 9는 안드로이드로 해상도를 전환하여 화면을 전송한 후 정보과학회 홈페이지에 접속한 것을 찍은 사진이다.

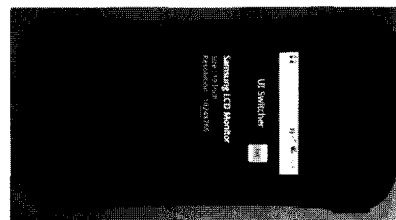


그림 8 주변의 장치를 감지한 어플리케이션

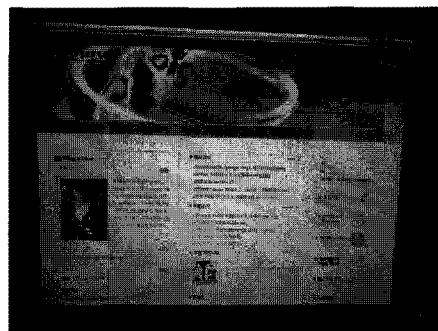


그림 9 주변의 장치로 화면을 전환한 사진

5.2 LCD Controller 구현

화면을 전송한 후에, 스마트폰의 LCD를 관리하는 LCD Controller를 구현하였다. (4.2절 참고) LCD Controller는 두 가지 기능을 지원하도록 구현하였다. 첫 째는, 스마트폰의 LCD에 리모콘의 역할을 할 수 있는 화면을 띄워서 리모콘의 동작을 할 수 있도록 하였고, 두 번째는 외부 장치에 전송한 화면을 드래그 하면서 볼 수 있도록 스크롤링 모드를 구현하였다. 리모콘 모드는 LCD side의 버퍼로 리모콘의 이미지 데이터를 복사하여 구현하였고, 스크롤링 모드는 외부 장치로 전송한 버퍼의 메모리에서 일부를 복사하여 구현하였다.

그림 10은 리모콘의 역할을 하는 스마트폰의 사진이고, 그림 11은 정보과학회의 홈페이지를 스크롤링 하여 보여주는 스마트폰의 사진이다.

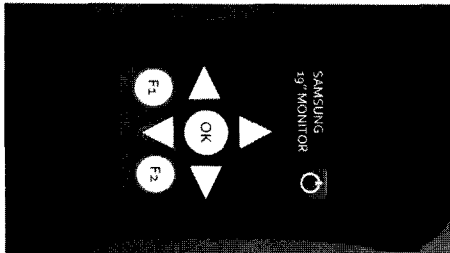


그림 10 리모콘 기능을 하는 스마트폰

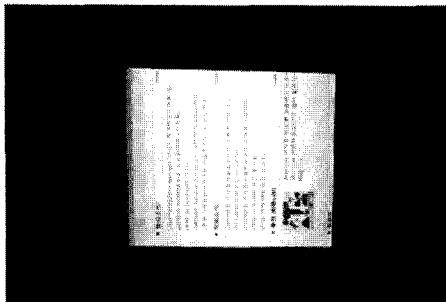


그림 11 스크롤링 기능을 하는 스마트폰

마트폰에서도 이러한 아이디어를 사용한다면 보다 좋은 서비스를 제공할 수 있을 것으로 기대된다.

참고 문헌

- [1] Myers 외 2인, "Using handhelds and PCs together," *Commun. ACM*, vol.44, no.11, pp.34-41, Nov. 2001.
- [2] L. Tarrini 외 4인, "Remote Control of Home Automation Systems with Mobile Devices," In *Proceedings of the 4th International Symposium on Mobile Human-Computer Interaction*, pp.364-368. Springer-Verlag, Sep. 2002.
- [3] L. Andreas 외 3인, "Using handheld devices for mobile interaction with displays in home environments," *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, pp.1-10, Sep. 2009.
- [4] J. Nichols 외 7인, "Generating remote control interfaces for complex appliances," In *Proceedings of the 15th annual ACM symposium on User interface software and technology*, pp.161-170, Oct. 2002.
- [5] M. Stogaitis and M. Sun, Gmote, <http://www.gmote.org/>, 2008.
- [6] Steinke and Bernd, "High definition multimedia display architecture for tiny mobile Smartphones," In *MobiMedia '07: Proceedings of the 3rd international conference on Mobile multimedia communications*, pp.1-4, Aug. 2007.
- [7] Apple. iPod and iPhone: TV out support. <http://support.apple.com/kb/ht1454/>, 2008.
- [8] Samsung. Samsung Instinct HD Phone. <http://www.samsungmobileusa.com/InstinctHD.aspx>, 2009.
- [9] O. C. Kim, J. S. Kim, C. G. Lee, E. Y. Ha, "The Development of Automatic User Interface Transform Technology on Android Mobile Platform," *Proc. of the KIISE Korea Computer Congress 2009*, vol.36, no.2(B), pp.436-440, 2009. (in Korean)

6. 결론

본 연구에서는 스마트폰에서 동적으로 해상도를 전환하여 외부 디스플레이 장치로 화면을 전환하는 기술을 제안하였다. 또한, 디스플레이가 전환된 이후에도 스마트폰의 기존 디스플레이 장치를 리모콘으로 활용함으로써 스마트폰의 활용성을 증가시켰다.

이러한 기술을 통해 스마트폰은 기존의 '휴대성'의 장점을 부각시키고, UI 측면의 단점을 해결할 수 있을 것으로 기대가 된다. 본 연구는 안드로이드 플랫폼 상에서 진행이 되었지만, 향후 출시되는 다른 플랫폼을 가진 스