운전자를 위한 벅스 안드로이드 어플리케이션

김동우*, 임은주*, 이장수*, 무디샤오*, 최낙중*, 고석주*

*경북대학교 IT 대학 컴퓨터학부

e-mail : kdw9242@gmail.com

BUGS android application for driver

Dong-Woo Kim*, Eun-Ju Im*, Jang-Su Lee*, MudiShaOh*, Nak-Jung Choi*, Seok-Joo Koh*

*Dept. of Computer Engineering, Kyung-Pook University

요 약

많은 사람들이 자동차를 교통수단으로 이용함에 따라서 다양한 자동차 액세서리 및 관심을 가지게 되면서, 자동차에서 다양한 여가 활동을 즐길 수 있게 되었다. 그와 함께 스마트폰 보급이 활성화가 되게 되면서 음악을 스마트폰으로 이용하여 들으면서 차에서도 많이들 활용하게 되었다. 하지만, 이러한 스마트폰 사용을 하게 됨으로 인해서 자동차 안에서 스마트폰을 사용하여 사고가 일어나는 빈도가 점점 많아지게 되었다. 그로 인해서 운전자들이 안심하고 사용할 수 있는 어플리케이션 개발이 필요로 하게 되었다. 현재 나와 있는 어플리케이션 중에 Bugs Driver[1]라는 음악 재생 어플리케이션이 있지만, 이 어플리케이션은 차량 운전자를 위한 특별한 기능을 가지고 있지는 않고, 기존 음악 어플리케이션에서 기능을 축소한 음악 어플리케이션이다. 또한, 직관적이지 않은 UI(User Interface)를 가지고 있으며, 속도 또한 충분히 빠르지 않아 운전자가 이용하기에 적합한 어플리케이션이 아니다.

이에 본 연구에서는, 차량 안에서도 안전하고 편리하게 사용할 수 있는 음악 어플리케이션을 개발하였다. 이 어플리케이션은 움직임을 최소화 할 수 있는 UI 와 모션인식을 이용한 어플리케이션 조작, 일정 속도 이상으로 주행 시 어플리케이션을 조작하지 못하도록 하는 기능을 가지고 있다. 음악 관련 정보는 ㈜네오위즈의 벅스[2] 어플리케이션 API 를 이용하였고, 모션인식은 OPENCV[3]를 이용하여 구현 하였다.

1. 서론

현재 많은 스마트폰의 보급으로 인해서 언제 어떤 상황에서도 사용을 하는 경우가 많다. 그로 인해 차량 운전 중에 어플리케이션을 조작하는 경우도 발생을 하게 된다. 그로 인해 많은 사고가 발생하고 있다. 그로 인해 법으로 이를 금하고 벌금도 정해져 있지만, 이를 지키는 운전자는 매우 적고, 적발하기도 어려운 실정이다. 이를 위하여 운전자를 위한 벅스 안드로이드 어플리케이션인 BUGS Driver 가 이미 시중에 나와 있지만, 이 어플리케이션은 운전 중 조작이 어려움은 물론, 충분히 빠르지 않은 속도와 직관적이지 않은 UI 를 가지고 있어, 운전 중 사용하기에는 적합하지 않다.

이에 본 논문에서는 운전자들이 보다 쉽고 안전하게 사용할 수 있는 어플리케이션을 만들기 위해서, 사용자 모션이나, 속도측정을 통한 어플리케이션 사용 통제기능이나, 음성인식을 이용한 검색 기능을 이용하여 운전자 전용 어플리케이션을 만들었다. 이 어플리케이션은 운전 중이나 그 외 다른 작업들을 하면서 편리하게 활용할 수 있어 운전 중 기기 조작으로 인한 사고율을 줄임과 다른 작업을 하면서도 음악 감상을 편리하게 할 수 있도록 하였다.

2. 연구 배경

2.1. BUGS API

Bugs API 는 네오위즈사의 벅스 어플리케이션을 통해서 음악 정보를 전달해 주는 API 이다. BUGS API 는 안드로이드에 설치되어 있는 BUGS 어플리케이션을 통해서 BUGS 서버와 통신 하여 음악정보 등의 데이터에 접근 할 수 있다. BUGS 서버에서 가져올 수 있는 정보에는 재생 목록을 비롯하여 현재 셔플모드인지, 순차 모드인지에 대한 정보, 음악 검색 정보가 있다. 또한, 재생이나 정지, 다음 곡 재생, 이전 곡 재생 등의 명령을 API 를 통해 할 수 있으며, 음악을 검색할 때에는 앨범, 곡, 가수 등의 카테고리를 선택하고 키워드와 함께 검색을 수행 할 수 있다.

2.2. BUGS Driver

BUGS Driver 는 이미 시중에 나와 있는 차량 운전

자 전용 안드로이드 어플리케이션이다. 이 어플리케이션은 운전자 전용 어플리케이션이지만, 운전 중에 사용하기에 불편한 점이 많다. 운전자가 운전을 하면서도 사용하기 편리하려면, 어플리케이션에 대한 접근성이 좋아야 한다. 하지만 이 어플리케이션은 기본 음악 재생 어플리케이션이나, 벅스 음악 재생 어플리케이션에 비해서 운전자를 위한 기능이 별도로 추가되어 있지도 않고, 단지 음악 어플리케이션의 기능을 축소시켜, 버튼을 크게 만들어 놓은 어플리케이션이다. 또한 이 어플리케이션은 속도가 느림은 물론 직관적이지 않은 UI 를 가지고 있어 운전자가 사용하기엔 적합한 어플리케이션이 아니라는 문제를 가지고 있다.

3. 운전자용 음악 재생 어플리케이션 개발

3.1. 접근성을 높이기 위한 UI(User Interface) 디자인

운전자를 위한 어플리케이션은 운전 중에 사용되기 때문에, 접근성이 좋아야 한다. 접근성을 좋게 하기 위해서는 어플리케이션에 대한 조작이 최소화되어야 한다고 생각을 하였다. 최소한의 조작을 위해서 화면 전환은 스와이프(Swipe) 기능을 기반으로 만들었다. 가운데의 메인화면을 기준으로 상하좌우에 다른 필요한 화면을 배치하였다. 오른쪽에는 재생 목록을 두어, 스와이프 한 번으로 재생 목록을 확인 할 수 있게 하였고, 아래쪽에는 검색 기능을 두었다. 검색 과정이 끝나면 왼쪽의 검색결과가 오른쪽으로 이동하면서 이용자에게 검색 결과를 보여 주도록 하였다. 검색 결과를 모두 확인 한 후에는 오른쪽에서 왼쪽으로 스와이프 하면 메인화면을 볼 수 있도록 하여 다시 동작을 쉽게 수행할 수 있도록 하였다.

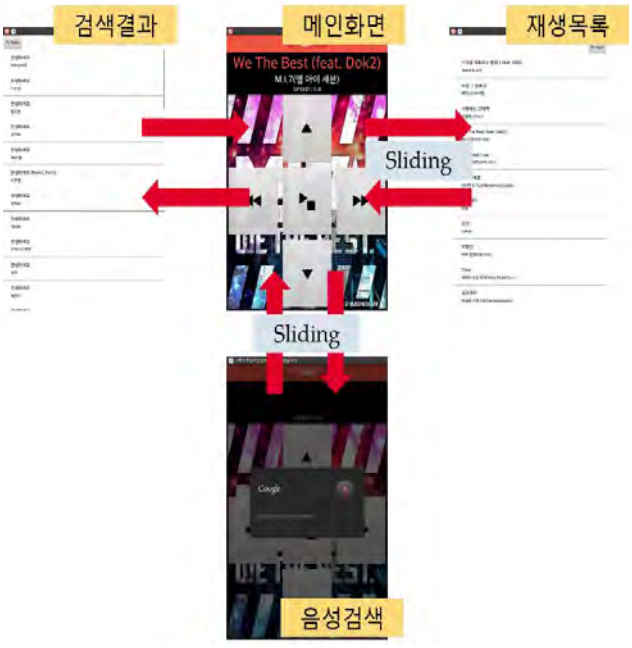

Figure 1. 운전자용 음악 재생 어플리케이션 UI

스와이프를 탐지해 내는 알고리즘은 터치 리스너를 이용하여 손이 닿았을 때의 위치와 손의 떨어진 위치를 측정하는 방법을 이용하였다. 손이 닿은 위치가 왼쪽 30% 영역이고, 손이 떨어진 위치가 오른쪽 30% 영역이면 좌에서 우로 스와이프 된 것으로 인식 하였다. 마찬가지로 우 30% 영역에서 좌 30% 영역까지 스와이프 한 것이면 우에서 좌로 스와이프 한 것으로 인식하였다.

하지만 아래, 위로 스와이프 하는 기능을 추가하면 좌, 우에 관해서만 스와이프 할 때와 달리 겹치는 부분이 있어 두 가지로 인식 되는 공간들이 발생하게 되었다.

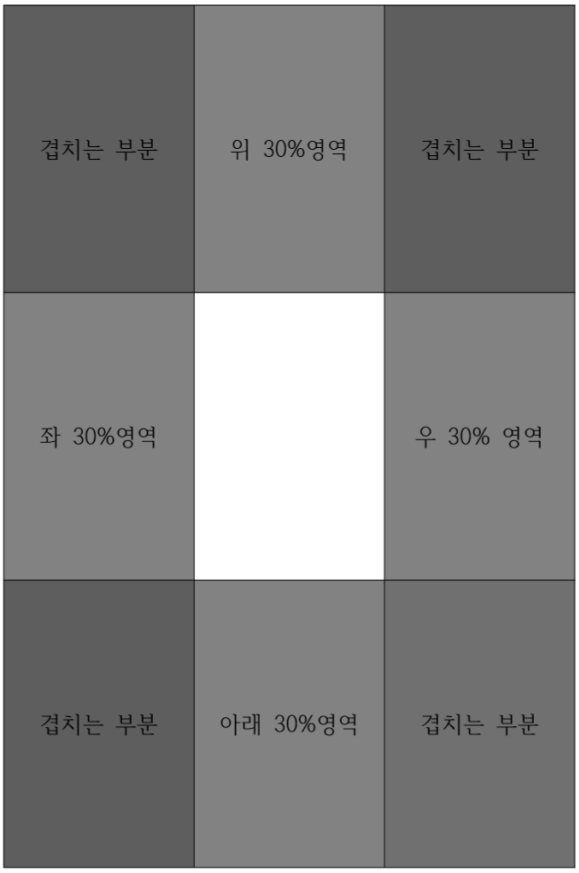

Figure 2. 영역을 선택하였을때 겹치는 영역들

이러한 문제가 발생되는 것을 해결하기 위하여, 스와이프 인식에 우선순위를 생각하였다. 좌우 스와이프를 상하 스와이프보다 우선순위를 주어 상하 스와이프와 좌우 스와이프가 동시에 인식 되었을 때는, 좌우 스와이프만 인식 된 것으로 하였다. 예를 들어 위의 왼쪽 아래의 공간에서 오른쪽 위의 공간으로 스와이프 하였을 때는, 좌에서 우로 스와이프도 인식되고, 아래쪽에서 위로 하는 스와이프도 인식된다. 이럴 경우 좌에서 우로 스와이프는 아래에서 위로 스와이프보다 우선시하여 인식한다.

좌우 스와이프를 상하 스와이프보다 우선시 한 이유는 좌우로 스와이프 하여 재생 목록을 보는 기능이, 상하로 스와이프하여 음성인식 기능을 이용하여 음악 검색을 하는 기능 보다 훨씬 자주 사용하는 기능으로 예측 되어 있기 때문이다.

스와이프 기능 이외에도 운전중 접근성을 올리기 위해서 커다란 버튼을 사용하여 조작이 가능 하도록 하였다. 음악 어플리케이션 조작에 필요한 재생, 정지, 이전 곡, 다음 곡, 볼륨조절 기능을 화면에 배치 하였다. 이외의 기능은 운전 중에 크게 필요치 않으므로, 버튼으로 추가하지 않았다.

3.2. 접근성을 높이기 위한 UI(User Interface) 디자인

운전 중에 버튼을 활용하거나 터지 화면을 터치하는 행동을 하는 것 자체가 위험할 수 있다.그 위험성을 제거하고 조금 더 운전자가 쉽게 접근 할 수 있는 방법을 위해서 제스처 인식을 활용하기로 하였다. 제스처 인식은 사용자의 움직임에 따라 이전 곡, 다음 곡 재생과 소리의 크기를 조절 할 수 있도록 하였다.

손을 좌에서 우로 움직이면 다음 곡, 우에서 좌로 움직이면 이전 곡, 아래에서 위로 움직이면 소리를 크게, 위에서 아래로 움직이면 소리를 작게 하도록 하였다.

어플리케이션에서 손을 인식하는 방법은 색 검출을 활용하기로 하였다. 먼저 전방이나 후방 카메라에서 이미지를 읽어 온다. 손의 움직임을 읽는데 고해상도의 사진은 필요하지 않으므로, 저해상도로 이미지의 해상도를 낮춘다. 사람마다 손의 색이 다르고, 같은 사람일지라도 빛의 세기나 역광의 여부에 따라 색이 달라진다.

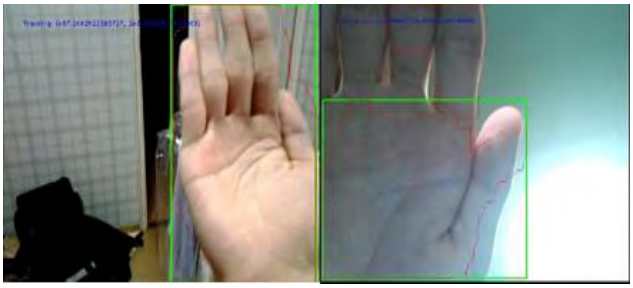

Figure 3. 빛에 따라 달라지는 손의 색

이러한 경우에도 손을 보다 잘 인식하기 위해서, 이미지를 HSV(Hue, Saturation Value) 모델로 읽는다. HSV 모델로 읽은 이미지에서 채도와 명도 정보는 제외하고, 색상 정보만을 이용하여 살색을 검출한다. 이렇게 함으로써 사람마다 손의 색이 다른 정도와 빛의 세기에 따른 색의 차이를 어느 정도 보정할 수 있다. 또한 이미지 중에서 검출할 색인 살색은 흰색으로 치환하고 살색을 제외하고 나머지는 모두 검은색으로 바꾸어 이진화한다. 어플리케이션을 사람들이 많은 곳에서 사용할 경우 살색인 부분이 여러 곳에서 검출 될 수 있다. 이렇게 여러 개의 살색 부분이 검출 될 경우 가장 큰 살색의 부분을 손으로 인식하고, 나머지 부분은 손으로 인식하지 않는다.

손의 움직임을 인식하기 위해서, 일정 시간이 지난 후에 카메라에서 또 다른 이미지를 읽어온다. 새로 읽어 온 이미지와 이전에 읽었던 이미지에서 살색 영역의 중심점을 계산한다. 이전에 읽은 이미지와 새로 읽은 이미지를 비교하여 손이 어느 방향으로 얼마나 움직였는지 계산한다. 측정한 손의 움직임이 일정치 이하면 조작이 아닌 것으로 판단하였다.

3.3. GPS 를 활용한 속도 측정 및 어플리케이션 사용 제약

어플리이케션의 제작 목적이 운전 중에 사용하기 편한 어플리케이션 이다 보니 운전중에 쉽게 조작 혹은 변조가 된다면 오히려 위험성이 더 심해질 것이라고 판단되었다. 그로 인해서 일어나는 문제가 있을 것으로 예상되어 속도가 높을 경우에 어플리케이션 동작에 제약을 두었다.

속도를 측정하기 위해서 GPS 기능을 사용하기로 하였다. 안드로이드에서는 GPS 기능을 사용할 수 있는 API 가 있었다. GPS API 에서 이용할 수 있는 정보는 GPS 에서 측정한 위치와 속도를 수치화된 데이터이다. 하지만 이 정보는 km/h 이 단위가 아니라 mile/h 단위이기 때문에, 환산을 위해 상수를 곱해 주어야 한다. 이 기능을 이용하여, 속도를 측정하고, 측정된 속도가 시속 40km 이상이 되면 모든 버튼의 조작을 하지 못하도록 하였다. 또한 측정된 속도가 시속 40km 미만이 되면 다시 버튼을 사용할 수 있게 하였다.

Figure 4. GPS 를 이용한 속도 측정 및 어플리케이션 사용 제약을 한 모습

Figure4 는 어플리케이션 통제 속도를 40km/h 에서 3km/h 로 줄이고, 달리면서 테스팅 하는 모습이다. 속도가 3km/h 보다 빠르면, 속도를 붉은 색으로 표시하고, 버튼을 사용 불가능 상태로 바꾼다. 다시 속도가 3km/h 보다 작아진다면, 속도를 하얀 색으로 표시하고, 버튼을 사용 가능 상태로 바꾼다.

3.4. 음성인식을 활용한 음악 검색

운전자가 음성인식을 이용해서 검색을 할 수 있다면, 별다른 움직임 없이 단 한 번의 조작만으로 검색

을 할 수 있다. 음성 인식 기능은 구글 안드로이드 STT[4](Speech To Text)를 활용하였다. 구글의 STT는 기본 내장 어플리케이션으로 사용하기에도 편리하고, 성능도 좋고, 스마트폰의 자원을 많이 소모하지 않아 선택하게 되었다.

음성 인식 기능은 음악 검색을 하기 위하여 활용된다. BUGS API 에는 음악 검색을 위한 기능이 포함되어 있다. Bugs API 로 음악 검색을 하기 위해서는 두 가지 정보가 필요하다. 첫 번째로, 곡, 앨범, 가수 등의 타입이 필요하다. 운전 중에 검색을 조금 더 빠르게 하기 위함과, 조작을 최소화하기 위해서 검색 타입은 기본적으로 곡이 되도록 하였다. 두 번째로는 검색어가 필요하다. 검색어는 음성인식을 이용하여 사용자로부터 입력 받는다. Google SST 는 입력 받은 음성을 분석하여, 가장 가까운 단어를 나열해준다. 이 중 첫 번째에 있는 단어가 가장 비슷한 단어이므로 기본적으로 첫 번째에 있는 단어를 선택 하도록 하였다. 이 두 가지 데이터를 Bugs API 를 이용하여 검색하면, Bugs API 는 관련된 여러 개의 검색 결과를 나열해 준다. 나열된 검색결과를 검색 결과 화면을 통해 사용자에 보여주고 사용자가 검색 결과 중 원하는 곡을 누르면, 선택 된 곡이 재생 되도록 하였다. 검색 결과 중 원하는 곡이 없다면 스와이프 기능을 활용하여 메인 화면으로 이동하거나, 메인 화면으로 돌아가는 버튼을 이용하여 메인화면으로 이동할 수 있다.

4. 결론

개발한 운전자를 위한 벅스 안드로이드 어플리케이션은 운전자들이 보다 안전하고 편리하게 음악 어플리케이션을 활용할 수 있도록 하는 많은 기능들을 가지고 있다. 이러한 기능들은 운전자의 어플리케이션 사용을 최소화 할 수 있도록 도와주며, 운전자가 보다 안전하게 어플리케이션을 사용할 수 있도록 많은 역할을 한다.

하지만 운전자가 운전 중에 음악을 감상하는 것이나, 어플리케이션을 사용하는 행위는 여전히 위험하다. 개발한 어플리케이션은 안전을 보장하지는 못하지만, 이러한 연구를 통해서 운전자의 안전을 보다 높일 수 있으므로 차후 연구가 필요하다.

참고문헌

[1] Bugs Driver. https://play.google.com/store/apps/details?id=com.neowiz.android.bugs.plugin.drive (2013)

[2] Bugs. http://www.bugs.co.kr/ (2014)

[3] OPENCV. http://opencv.org/ (2014)

[4] Google SST. http://developer.android.com/reference/android/speech/package-summary.html (2014)