Journal of Digital Contents Society Vol. 12 No. 1 Mar. 2011(pp. 57-67)

이미지 내 텍스트 추출 및 검색을 위한 안드로이드 모바일 시스템 구현

고은비*, 하유진**, 최수렴***, 이기훈****, 박영호*****

요 약

스마트폰의 보급량이 증가하면서 이동성과 휴대성이 강조된 모바일 검색이 주목 받고 있다. 그러나 모바일 검색을 위한 일반적인 키워드 입력 수단은 키패드로 제한되어 있다. 키패드는 모바일 기기가 갖는 이동성과 휴대성에 적합하지 않다는 단점을 갖는다. 이를 보완하기 위해 음성을 이용한 검색이 등장하였지만, 이 또한 단점을 지니고 있다. 따라서, 본 논문에서는 다양한 상황에서의 정보 접근성을 향상시키기 위해 이미지를 검색 수단으로 사용하는 검색 시스템을 제안한다. 본 콘텐츠는 안드로이드 플랫폼 기반의 스마트폰에서 이미지를 얻어 텍스트를 추출하고, 이를 검색 엔진의 키워드로 입력하여 그 결과를 출력하는 과정을 거친다. 또한, 검색 결과를 스마트폰의 내장 데이터베이스에 저장하고, 이를 관리하여 추후에 재사용할 수 있도록 한다. 실험을 통해 인식 가능한 이미지의 특성을 분석하고, 본 콘텐츠의 기능을 소개한다.

An Implementation of an Android Mobile System for Extracting and Retrieving Texts from Images

Eun-Bi Go*, Yu-Jin Ha**, Soo-Ryum Choi***, Ki-Hoon Lee****, Young-Ho Park*****

Abstract

Recently, an interest in a mobile search is increasing according to the growing propagation of smart phones. However, a keypad, which is not appropriate for mobile environment, is the only input media for the mobile search. As an alternative, voice emerged as a new media for the mobile search, but this also has weaknesses. Thus, in the paper, we propose a mobile content called Orthros for searching the Internet using images as an input. Orthros extracts texts from images, and then inserts the texts to public search engines as a keyword. Also, Orthros can repeat searching with the extracted texts by storing result URL to internal databases. As an experiment, we analyze properties of recognizable images and present the implementation method in details.

Keywords : OCR(Optical Character Recognition), Tesseract, WeOCR, Mobile Search, Android

1. 서론

모바일 네트워크 고도화 및 단말기의 비약적인 발전으로 스마트폰 보급이 확산됨에 따라[1] 모바일 검색이 주목받고 있다. 스마트폰의 상시 휴대성으로 인해 정보 접근성이 증대되어 언제, 어디서나 검색이 가능하기 때문이다.

그러나, 일반적으로 사용하는 모바일 검색 입력 수단은 키패드가 유일하다. 또한, 스마트폰의 사용량의 증가로 터치패널(touch-panel)이 보편화되고 있다. 이러한 인터페이스는 사용자에게 익숙하지만, 시각에 의존하므로 모바일 검색에는 부적합하다는 단점이 있다.

※ 제일저자(First Author) : 고은비
접수일:2010년 12월 22일, 수정일:2011년 3월 23일, 완료일:2011년 3월 30일
* 숙명여자대학교 멀티미디어과학과
ebstory@naver.com
** 숙명여자대학교 멀티미디어과학과
*** 숙명여자대학교 멀티미디어과학과
**** KT 연구소
***** 숙명여자대학교 멀티미디어과학과 교신저자

음성은 키패드와 터치패널을 이용한 검색의 단점을 보완하는 입력 수단이다. 음성은 인간에게 가장 자연스러운 통신 방법[2]으로, 시간이나 장소에 대한 제약이 적다. 그러나 음성을 이용한 모바일 검색은 주변 소음이나 사용자의 억양, 발음 등 검색 환경에 따라 인식률이 급변한다는 문제점을 갖는다. 또한, 음성을 사용할 수 없는 상황이 존재하므로 음성을 보완하는 입력 매체가 제안되어야 한다.

따라서, 본 논문에서는 이미지를 이용하여 모바일 검색을 실시하는 오르트로스(Orthros)를 소개한다. Orthros는 그리스어로 똑바르다, 바르다는 의미[3]를 지니는 단어로, 모바일 환경에서의 정보 접근성 향상을 위한 콘텐츠를 명명한 것이다. Orthros는 모바일 기기를 통해 얻은 이미지에서 텍스트를 추출한 뒤, 추출한 텍스트를 상용 검색 엔진의 키워드로 활용하여 정보 검색을 하는 단계로 진행된다. 본 논문에서는 오르트로스를 Orthros로 표현한다.

본 논문은 기존 상용 기능인 OCR을 직접 개발하지 않고 개발된 오픈 소스 OCR 엔진을 사용하였으며, 안드로이드 플랫폼 기반의 모바일 검색을 위한 콘텐츠를 구현하였다. 안드로이드(Android)는 무료로 제공하는 최초의 오픈 소스 모바일 플랫폼[4]으로, 사용자가 지속적으로 증가할 것으로 기대되며 그 근거는 다음과 같다.

본 논문은 다음과 같은 공헌을 제시한다.

- 스마트폰을 사용하여 이미지 내에 존재하는 텍스트를 추출하고, 추출한 텍스트를 이용하여 즉각적인 모바일 검색을 실시하는 콘텐츠인 Orthros를 제안한다.
- Orthros는 안드로이드 플랫폼에서 제공하는 오토 포커스 기능을 사용하고, 이미지 내의 텍스트 영역을 지정하여 인식 범위를 축소하는 텍스트 크롭 기능을 구현하여 텍스트 인식률을 높인다.
- Orthros는 검색 결과를 스마트폰의 내장 데이터베이스에 저장하고, 이를 추후에 재검색하여 학습할 수 있도록 한다.

본 논문에서 사용하는 용어는 다음과 같다. 첫 번째, OCR 서버(OCR Server)는 이미지에서 텍스트를 추출하기 위해 구축된 PC를 의미한다. OCR 서버는 오픈 소스 OCR 엔진인 Tesseract를 가져와 컴파일 하였다. 이후, 네트워크 상에서 Teseraact를 구동시키기 위한 플랫폼인 WeOCR를 구동한다. 두 번째, 모바일 클라이언트(Mobile Client)는 안드로이드 플랫폼 기반의 스마트폰 어플리케이션을 의미한다. 클라이언트를 탑재하는 스마트폰은 무선 인터넷을 지원하고 카메라를 내장한다. 세 번째, 오토 포커스(Auto Focus)는 스마트폰 화면 내의 버튼을 선택하여 이미지 내의 노이즈를 제거하고 빈번하게 나타나는 부분을 추출하는 수동 기능이다[6]. 마지막으로, 텍스트 크롭(Text Crop)은 모바일 단말기의 화면에서 대상 이미지의 특정 영역을 터치 방식으로 조절하여 이미지 내의 문자 인식률을 향상시킬 수 있도록 영역을 축소하는 수동 기능이다.

본 논문의 구성은 다음과 같다. 2 장에서는 OCR 서버에서 구동하는 Tesseract, WeOCR과 모바일 클라이언트에서 구현한 텍스트 크롭 기능을 소개한다. 3 장에서는 Orthros의 구조, 프로토콜, OCR 서버와 모바일 클라이언트의 주요 기능을 설명한다. 4 장에서는 Orthros의 주요 알고리즘을 설명하고, 5 장에서는 Orthros의 구현을 서버와 클라이언트로 나누어 설명한다. 6 장에서는 실험을 통해 Orthros의 인식 가능한 이미지를 분석한다. 마지막으로 7 장에서는 결론과 기대 효과를 설명한다.

2. 관련 연구

본 장에서는 OCR 서버에서 구동하는 오픈 소스 OCR 엔진인 Tesseract와 웹에서 OCR엔진을 구동시키는 WeOCR에 대해 소개하고, 모바일 클라이언트에서 텍스트 인식률을 높이기 위한 텍스트 크롭 기능을 설명한다. OCR(Optical Character Recognition)은 이미지를 텍스트 기반의 수정 가능한 파일로 변환하는 기술이다[7].

2.1 Tesseract[8]

Tesseract는 1984년부터 약 10년간 Hewlett and Packard(HP)에서 개발한 오픈 소스 OCR 엔진이다[8]. 개발 이후 지속적인 성능 개선을 통해 2005년에 Tesseract를 오픈 소스로 발표하였다. 현재는 구글(Google) 사가 Tesseract의 일부를 지원하고 있다[9].

Tesseract의 작동 과정은 그림 1[10]과 같다. 첫째, 회색 이미지나 색상 이미지를 OCR 엔진의 입력으로 입력하고 난 뒤, 임계값을 이용하여 입력 이미지를 이진화한다. 둘째, 이진화한 이미지의 연결된 구성 요소를 분석하여 각 구성 요소의 외곽선을 추출하고, 이를 이진 데이터의 집합인 블랍(blob) 형태로 저장한다. 셋째, 텍스트 라인(text line)을 분석하여 블랍을 체계화 한 뒤, 문자의 자간에 따라 단어 단위로 나눈다. 마지막으로, 단어 단위로 나뉜 블랍을 단어 단위와 페이지 단위로 인식한다[8]. Tesseract는 출력 파일의 형식이 존재하지 않으므로[11] 출력 파일을 생성하기 위한 단계는 존재하지 않는다.

Tesseract는 레이아웃에 대한 분석이 존재하지 않아 다수의 텍스트 라인으로 이루어진 이미지는 왜곡될 수 있다. 그러므로, 입력 이미지는 반드시 TIFF 형식이어야 한다[11].

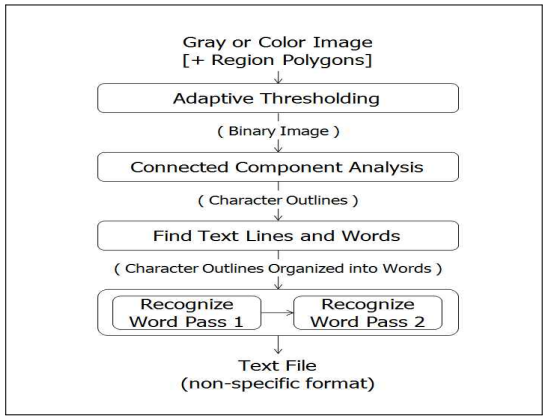
본 논문에서 Tesseract는 OCR 서버의 한 부품으로 사용되며, 텍스트를 추출하는 주요 기능을 수행한다.

2.2 WeOCR[12]

WeOCR은 네트워크 상에서 문자 인식을 가능하도록 하는 OCR 시스템의 플랫폼이다.

WeOCR은 사용자로부터 이미지를 입력받아 텍스트를 인식한 뒤, 그 결과를 사용자에게 전송한다[12]. 그러나 WeOCR은 자체의 OCR 엔진을 사용하지 않으므로, 오픈 소스 OCR 엔진과 연동하여 텍스트를 인식한다.

본 논문에서 WeOCR은 OCR 엔진을 웹에서 편리하게 사용할 수 있도록 하는 툴킷 역할을 하며, Tesseract와 함께 OCR 서버를 구성한다. 또한, 모바일 클라이언트와 Tesseract 간의 파일 변환과 데이터 전송을 실시한다.



(그림 1) Tesseract의 동작 과정

3. 오르트로스(Orthros)

본 장에서는 Orthros의 개요와 시스템 구조, 프로토콜에 대해 설명한 뒤, Orthros를 OCR 서버와 모바일 클라이언트로 나누어 역할과 주요 기능에 대해 설명한다.

3.1 개요

Orthros는 이미지에서 텍스트를 추출하고, 이를 키워드로 이용하여 모바일 환경에서 정보 검색을 실시하는 콘텐츠이다.

Orthros는 OCR 서버와 모바일 클라이언트로 구성되어 있다. OCR 서버는 모바일 클라이언트가 전송하는 이미지에서 텍스트를 인식, 추출한다. 모바일 클라이언트는 사용자의 요청을 받아 OCR 서버에 전달하고 OCR 서버의 응답을 사용자에게 보여준다.

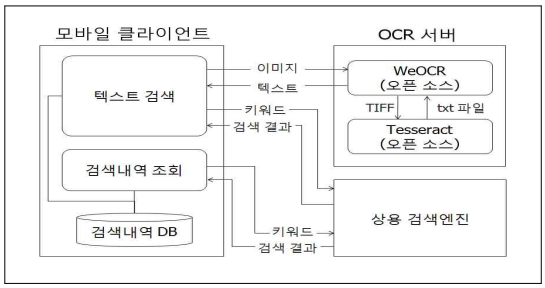
모바일 클라이언트는 OCR 서버의 텍스트 인식률을 높이기 위해 오토 포커스 기능과 텍스트 크롭 기능을 제공한다. 오토 포커스 기능은 모바일 클라이언트에 내장되어 있는 카메라 초점을 맞추는 역할을 수행하고, 텍스트 크롭 기능은 이미지의 텍스트 영역을 축소하여 텍스트 인식률을 높인다.

Orthros는 이미지를 이용하여 모바일 검색을 실시하기 때문에 다양한 상황에서의 검색이 가능하도록 한다. 이를 통해 모바일 환경에서의 정보 접근성을 높일 수 있다.

3.2 시스템 구조

그림 2는 Orthros의 시스템 구조와 송수신 데이터를 나타낸다. 모바일 클라이언트는 텍스트 검색을 통해 OCR 서버로 이미지를 전송하여 추출한 결과 텍스트를 상용 검색 엔진의 키워드로 이용하여 모바일 검색을 한다. 키워드로 사용된 텍스트는 모바일 클라이언트의 내부 데이터베이스인 검색내역 DB에 저장되고, 검색내역 조회를 통해 열람 및 재검색이 가능하다.

OCR 서버는 모바일 클라이언트로부터 받은 이미지에서 인식, 추출한 텍스트를 반환하며, WeOCR과 Tesseract로 구성된다. Tesseract는 이미지 내의 텍스트를 인식, 추출하고, WeOCR은 Tesseract와 모바일 클라이언트를 연결하는 역할을 수행한다.

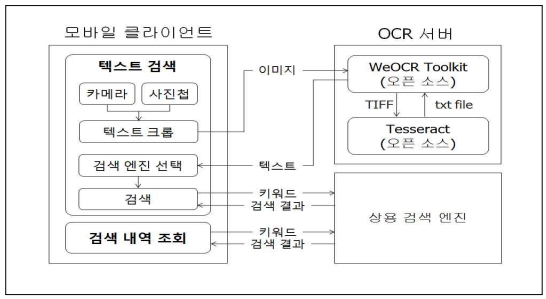


(그림 2) Orthros의 시스템 구성도

3.3 프로토콜

그림 3은 Orthros의 작동 과정을 텍스트 검색 기능과 검색내역 조회 기능으로 나누어 나타내며, 다음과 같은 순서로 진행된다. 모바일 클라이언트가 내장 카메라나 사진첩을 통해 이미지를 얻은 뒤, 텍스트 크롭을 사용하여 이미지 내의 텍스트 영역을 잘라내고, 이를 OCR 서버로 전송한다. OCR 서버의 WeOCR은 이미지를 TIFF 형태로 변환하여 Tesseract에 전송하면, Tesseract는 이미지에서 추출한 텍스트를 텍스트 파일 형태로 WeOCR에 전송한다. WeOCR은 파일 내의 텍스트를 모바일 클라이언트에 전달한다. 모바일 클라이언트는 검색에 사용할 사용 검색엔진을 선택하고, OCR 서버로부터 얻은 텍스트를 이용하여 모바일 검색을 실시한다.

텍스트 검색 기능을 수행하면서 저장한 추출 텍스트 관련 정보를 이용하여 재검색이 가능하도록 하였다. 재검색은 추출 결과 텍스트를 상용 검색 엔진의 키워드로 전송하면 검색 결과가 모바일 클라이언트에 전송되는 과정을 거친다.



(그림 3) Orthros의 동작 과정

3.4 OCR 서버

OCR 서버는 Apache[13], PHP[14], 요청처리 프로그램, WeOCR, Tesseract로 구성된다. 각 구성요소의 역할과 동작 과정은 다음과 같다.

Apache는 HTTP를 이용하여 모바일 클라이언트와의 통신을 지원하고, PHP는 동적 웹 프로그램인 요청 처리 프로그램을 구현하기 위하여 사용된다.

요청 처리 프로그램은 본 논문에서 직접 구현한 PHP 프로그램으로 모바일 클라이언트로부터 요청으로 받은 이미지를 저장하고 이를 WeOCR의 입력 인자로 넘겨주고 출력 결과를 응답으로 생성하는 기능을 수행한다.

Tesseract는 오픈소스로 제공되는 OCR 엔진이며, 이미지에서 텍스트를 인식, 추출하는 중추적 역할을 수행한다. Tesseract의 입력 이미지는 TIFF형태로 제한되어 있기 때문에 모바일 클라이언트로부터 전달받은 이미지를 적합한 형태로 변환하는 과정이 필요하다.

WeOCR은 요청 처리 프로그램으로부터 입력받은 이미지를 Tesseract에서 사용가능한 형태로 변환해 주고, Tesseract의 텍스트 추출 결과를 요청 처리 프로그램에 전달하는 중간 연결부의 역할을 한다.

Tesseract의 텍스트 추출 결과는 임시 텍스트 파일로 저장되고, OCR 서버 프로그램은 이 파일을 파싱하여 모바일 클라이언트에 텍스트 형태로 응답한다. 이 과정에서 텍스트 추출 결과 파일을 임시로 저장함으로써 저장 공간의 낭비를 줄이는 효과를 얻을 수 있다.

3.5 모바일 클라이언트

본 절에서는 모바일 클라이언트의 주요 기능을 텍스트 검색 기능과 검색내역 조회 기능으로 나누어 설명한다.

3.5.1 텍스트 검색 기능

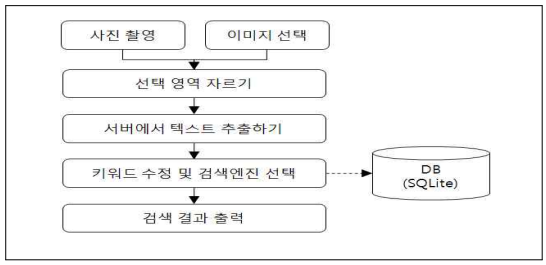
텍스트 검색 기능은 이미지에서 OCR 서버에서 추출한 텍스트를 키워드로 이용하여 모바일 검색을 수행한다.

텍스트 검색 기능은 그림 4와 같은 단계로 진행된다. 첫째, 모바일 클라이언트가 내장 카메라를 이용하거나 내부 메모리에서 이미지를 선택한다. 둘째, 이미지를 텍스트 크롭 한다. 그림 5는 텍스트 크롭의 동작 과정이다. 텍스트 크롭은 사용자가 가변 프레임을 반복적으로 사용하여 최종적으로 결정한 텍스트 영역을 새로운 파일로 저장한다. 셋째, 저장된 이미지를 OCR 서버로 전송하여 텍스트를 추출하고, 이를 모바일 클라이언트가 전달 받아 키워드로 설정한다. 넷째, 키워드가 사용자 의도와 다른 경우 수정하고, 구글(Google), Dictionary, 네이버(Naver), 위키피디아(Wikipedia) 중 사용자가 선택한 포털의 검색 결과를 출력한다. 추출한 텍스트의 관련 정보는 검색이 수행되는 시점에 모바일 클라이언트의 내장 데이터베이스인 검색내역 DB에 저장되어 검색 내역 조회 기능에 사용된다.

3.5.2 검색내역 조회 기능

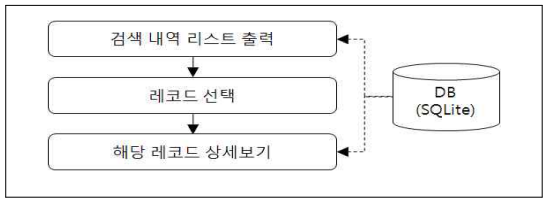
검색내역 조회 기능은 모바일 클라이언트의 내장 데이터베이스인 검색내역 DB에 접근하여 저장되어 있는 정보를 조회한다. 저장되어 있는 정보는 텍스트 검색 기능을 통해 추출한 텍스트에 대한 세부 정보(검색 날짜, 텍스트 크롭한 이미지, 검색 키워드, 검색결과 URL)이다.

그림 6은 검색내역 조회 기능의 동작 과정을 나타낸다. 검색내역 DB에 저장된 검색내역 리스트를 화면에 출력한 뒤, 사용자가 특정 아이템을 선택하면 해당 아이템에 대한 상세 정보를 조회할 수 있도록 출력한다.



(그림 4) 텍스트 검색 기능의 동작 과정

(그림 5) 텍스트 크롭 기능의 동작 과정



(그림 6) 검색내역 조회의 동작과정

4. 알고리즘

본 장에서는 OCR 서버와 모바일 클라이언트 간의 통신을 위한 알고리즘을 설명한다.

4.1 OCR 서버 알고리즘

본 절에서는 OCR 서버의 구성 요소 중 요청 처리 프로그램의 알고리즘을 설명한다. 요청 처리 프로그램은 모바일 클라이언트가 요청한 이미지 파일을 WeOCR의 입력으로 전달하고, 결과로 반환되는 텍스트 파일을 파싱하여 클라이언트에 전송한다.

그림 7은 요청 처리 알고리즘을 나타내며 다음과 같은 순서로 진행된다. 첫째, 모바일 클라이언트가 업로드한 이미지 파일을 로드하고, 날짜와 시간을 이용하여 새로운 파일 이름을 생성한다. 둘째, 이미지 파일을 저장할 디렉토리를 설정하고 전 단계에서 명시한 파일 이름으로 저장한다. 셋째, 이미지 파일이 성공적으로 저장된

경우에는 WeOCR 명령어에 해당 이미지 파일을 입력하고 결과 텍스트 파일을 저장할 경로를 지정한 후, 명령어를 수행한다. 이를 통해 생성된 결과 파일을 파싱하여 결과 텍스트를 모바일 클라이언트에 전송한다.

```
# Algorithm for Handling Client Requests
Input: Uploaded Image File uploadedFile
Output: Result text resultText

Algorithm Start:
Step 1. Load uploadedFile and create file name
Step 2. Set directory to store uploadedFile and
        save uploadedFile at the directory as
        new file name
Step 3. If (Saving uploadedFile succeeds)
            1. Set WeOCR command input as
               saved uploadedFile and set
               output file path
            2. Execute WeOCR command
            3. Parse the output file and set
               resultText as parsed string
        Else, set resultText as exception message
Step 4. Return resultText to client
END
```

(그림 7) OCR 서버의 요청 처리 알고리즘

4.2 모바일 클라이언트 알고리즘

본 절에서는 OCR 서버와 통신하기 위한 모바일 클라이언트의 알고리즘을 설명한다. 모바일 클라이언트는 이미지에서 텍스트를 추출하기 위하여 스마트폰의 내장 카메라나 내장 데이터베이스를 통해 텍스트 크롭 기능을 수행한 후에 OCR 서버와 통신 한다.

모바일 클라이언트는 OCR 서버로 이미지를 전송하는 단계와 OCR 서버의 추출 결과 텍스트를 수신하는 단계로 구성되며, 그림 8과 같은 순서로 진행된다. 첫째, 모바일 클라이언트 내 파일 경로를 이용하여 이미지를 전송하기 위한 스트림을 생성한다. 둘째, URL을 이용하여 OCR 서버의 요청 처리 프로그램과 통신한다. 셋째, 파일 경로를 파일 이름으로 전송한다. 넷째, 이미지가 모두 전송될 때까지 지속적으로 데이터를 전송한다. 다섯째, 이미지 데이터가 모두 전송되면 스트림을 닫는다. 여섯째, 추출 결과 텍스트를 수신받기 위해 스트림을 생성한다. 일곱째, 추출 텍스트가 모두 수신될 때까지 데이터를 받는다. 마지막으로, 스트림과 연결을 차례로 닫고 결과 텍스트를 반환한다.

```
# Algorithm for Communicating with Server
Input: Server URL urlString, File Path filePath
Output: Result text resultText

Algorithm Start:
Step 1. Create an output stream of image using
        filePath
Step 2. Open new connection with server using
        urlString
Step 3. Send filePath to server as a filename
Step 4. While (Output stream exists)
            Send image
Step 5. Close output stream
Step 6. Create an input stream to get resultText
Step 7. While (Input stream exists)
            Receive resultText
Step 8. Close input stream
Step 9. Close connection
Step 10. Return resultText
END
```

(그림 8) 모바일 클라이언트의 요청 알고리즘

5. 구현

본 장에서는 Orthros의 OCR 서버의 설치 과정과 모바일 클라이언트의 세부 구현에 대해 설명한다.

5.1 OCR 서버 구현

본 절에서는 OCR 서버를 구성하는 OCR 서버를 구축하기 위한 설치 과정을 설명한다.

5.1.1 설치 과정

OCR 서버 구축은 웹 서버 환경의 구축이 선행된 후에 본격적으로 이루어진다. 웹 서버 환경은 Apache와 PHP를 설치함으로써 구축할 수 있으며, 이는 보편적으로 이루어지는 과정이므로 자세한 설명은 생략한다.

웹 서버 환경의 구축이 완료된 후에는 OCR 작업을 처리할 Tesseract와 WeOCR을 설치해야 한다. Tesseract는 WeOCR의 일부 모듈로 사용되므로 Tesseract의 설치가 완료된 후 WeOCR을 설치해야 한다.

첫째, Tesseract의 아카이브를 다운로드하고 원하는 경로에 빌드하여 설치한다. 설치 후에는 OCR 엔진을 통해 추출할 언어를 설정해야 한다. 현재 제공되고 있는 언어의 종류는 영어, 독일어, 프랑스어, 일본어, 중국어 등 다양한데, 각 언어를 추출하기 위해 언어 데이터인 tessdata가 필요하며 이는 프로젝트 홈페이지에서 언어 별로 제공되고 있다. 본 논문에서 제안하는 Orthros에서는 영어에 대한 텍스트 추출을 지원하므로 이에 해당하는 tessdata를 다운로드하여

설정했다. 다운로드 받은 tessdata를 Tesseract의 share 디렉토리에 위치시키고 모든 사용자가 읽을 수 있도록 권한을 변경함으로써 추출 언어를 설정하였다.

둘째, WeOCR 설치의 선행과정으로 이미지 변환과 이미지 이진화를 위한 영상처리 패키지인 Netpbm[16], O2-tools[5]와 JPEG 라이브러리인 libjpeg-devel[18]를 설치해야 한다. 각각은 해당 홈페이지에서 아카이브를 다운로드하여 원하는 경로에 빌드함으로써 설치할 수 있다.

셋째, WeOCR의 아카이브를 다운로드한 뒤 빌드하여 설치한다. 설치 후에는 사용할 OCR 엔진을 설정하고 Apache와의 연동을 위해 몇 가지 설정을 해주어야 한다.

WeOCR에서 사용할 수 있는 엔진의 종류로는 GOCR, Ocrad, Tesseract 등이 있으며, Orthros에서는 Tesseract를 사용했다. 따라서 WeOCR의 bin 디렉토리에 위치한 Tesseract 설정 파일의 이름을 변경하여 활성화 시키고 파일 내용 중 $OCR_CMDSTR에 설정된 값을 Tesseract 실행 파일의 경로로 변경하여 OCR 엔진 명령을 수행할 때 Tesseract가 실행되도록 한다.

다음 과정은 Apache와 연동하기 위한 설정으로 WeOCR의 html 디렉토리와 cgi-bin 디렉토리 내 모든 파일을 각각 Apache의 웹 루트 디렉토리와 CGI-BIN 디렉토리 아래에 복사하여 위치시키고, 복사한 파일 중 CGI-BIN 내 weocr 디렉토리의 submit.cgi 내용 중 WeOCR의 BIN 디렉토리를 설정해주는 $WOBINDIR에 설정된 값을 WeOCR의 bin 디렉토리 경로로 수정하여 저장하면 된다.

넷째, WeOCR의 설치와 Apache와의 연동까지 완료되면 http://[웹서버 IP주소]/weocr/ 페이지에 접속하여 테스트 해 볼 수 있다. 테스트 이미지를 업로드하여 알맞은 결과 텍스트가 출력된다면 모든 설치를 성공적으로 끝마친 것이다. 만약 오류가 발생한다면 tmp 디렉토리 내에 위치한 WeOCR의 로그 파일인 tesseract.out의 내용을 확인하여 원인을 해결하도록 한다.

OCR 서버는 오픈소스인 WeOCR과 Tesseract를 사용하여 구현되었다. WeOCR은 OCR 엔진의 설치와 설정 파일의 변경을 통해 원하는 OCR 엔진을 선택하여 사용할 수 있다. 따라서, Orthros는 추후에 텍스트 인식률이 더 높은 OCR 엔진으로의 변경이 가능하다는 장점을 지닌다.

또한, 현재 사용 중인 Tesseract는 다양한 언어의 추출을 지원하는 OCR 엔진으로 새로운 언어가 계속해서 추가되고 있다. Tesseract는 언어 데이터를 다운로드하여 추출할 언어를 유동적으로 변경, 추가할 수 있는 장점을 지니므로, 이를 이용하여 추후에 다양한 언어의 텍스트를 인식, 추출하는 등 지속적인 확장, 발전을 할 수 있다.

5.2 모바일 클라이언트 구현

모바일 클라이언트는 윈도우 PC에서 안드로이드 2.2 플랫폼을 기반으로 Eclipse Ganymede를 사용하여 구현하였으며, 테스트 단말기로 HTC 사의 넥서스원(GGL-NX1) 모델을 사용하였다.

본 절에서는 모바일 클라이언트의 인터페이스와 인식률 개선 기능 구현에 대해 설명한다.

5.2.1 인터페이스 구현

모바일 클라이언트의 인터페이스 구현은 이미지를 이용한 텍스트 검색 기능과 재검색을 위한 검색내역 조회 기능으로 구분하여 설명한다.

검색내역 조회 기능의 화면은 그림 9와 같다.

그림 9는 모바일 클라이언트의 내장 메모리를 이용한 텍스트 검색 과정순차적으로 나타낸다. 그림 9(a)는 메뉴를 나타내고, '기존 이미지로 검색' 버튼을 클릭하면 그림 9(b)가 나타난다. 하나의 이미지를 선택하고 이에 텍스트 크롭 기능을 적용하면 각각 그림 9(c)와 그림 9(d)를 볼 수 있다. 그림 9(e)는 OCR 서버는 텍스트 추출 결과와 검색하고자 하는 상용 검색 엔진을 선택하는 화면이며, 그림 9(f)는 원하는 검색 엔진으로 검색한 결과이다.

그림 10(a)는 검색내역을 나타내는 화면이고, 그림 10(b)는 검색 세부 정보를 나타낸다. 세부 정보의 삭제 버튼을 선택하면, 삭제 의사를 확인하기 위한 화면이 그림 10(c)와 같이 나타난다.

(a) 메인 화면 (b) 이미지 선택 (c) 텍스트 크롭

(d) 텍스트 크롭 후 (e) 추출 결과 및 검색엔진 선택 (f) 검색 결과

(그림 9) 텍스트 검색 기능 화면

(a) 단어장 (b) 단어 선택 (c) 단어 삭제

(그림 10) 검색내역 조회 기능 화면

5.2.2 인식률 개선 기능 구현

인식률 개선 기능은 OCR 서버의 텍스트 인식률을 향상시키기 위한 기능으로, 오토 포커스 기능과 텍스트 크롭 기능이 있다.

오토 포커스 기능은 모바일 클라이언트가 스마트폰의 내장 카메라를 사용할 때 화면에 보이는 Focus 버튼을 선택하여 사용할 수 있다.

오토 포커스 기능은 안드로이드 플랫폼에서 제공하는 API를 이용하여 구현한다. 그림 11은 오토 포커스의 구현 소스이다. XML 파일에서 정의한 Focus 버튼에 대한 이벤트 리스너에 오토 포커스 API를 추가한다.

오토 포커스를 스마트폰에서 사용하기 위해서는 그림 12와 같이 안드로이드 프로젝트의 AndroidManifest.xml 파일에 uses-feature 태그를 사용하여 오토 포커스 기능의 사용을 명시해야 한다.

```
// 오토 포커스 시작
findViewById(R.id.focus).setOnClickListener(
    new Button.OnClickListener() {
        public void onClick(View v) {
            mSurface.mCamera.autoFocus(mAutoFocus);
            btn_shutter.setEnabled(true);
        }
    });
```

(그림 11) 오토 포커스의 구현 소스

```
<uses-feature
    android:name="android.hardware.camera.autofocus"/>
```

(그림 12) 오토 포커스 활성화 소스

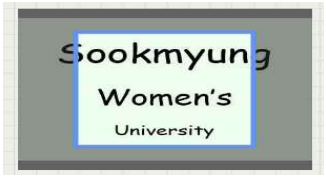
텍스트 크롭 기능은 이미지에서 텍스트가 존재하는 영역만을 잘라내어 텍스트만이 존재하는 이미지를 OCR 서버에 전송함으로써 정확한 추출이 가능하도록 하며, 그림 13과 같이 나타난다.

텍스트 크롭 기능은 크게 3개의 클래스로 나뉜다. 첫째, 이미지를 자르는 CropImage 클래스와 둘째, 그림 13의 가변 프레임을 나타내는 HighlightView 클래스, 마지막으로 가변 프레임을 조절하는 CropImageView 클래스로 나눌 수 있다.

CropImage 클래스는 이미지의 URI를 통해 이미지를 가져오고 사용자의 액션을 감지하여 사용자가 선택한 텍스트 영역을 즉시 자른다.

HighlightView 클래스는 가변 프레임에 대한 속성을 포함하며, 화면에는 그림 13과 같이 파란색 사각형으로 나타난다. 사각형 테두리를 선택했을 때 보이는 화살표 이미지나 화면의 크기를 통해 사각형을 조정할 수 있는 공간을 계산하는 등의 기능을 한다.

CropImageView 클래스는 CropImage형 객체에서 가공하고 있는 이미지를 보여준다. HighlightView를 그리고, 사용자의 touch action에 따라 가변 프레임을 조작한다. 각종 함수를 이용하여 가변 프레임을 이동, 확대, 축소 시키며, 자르고 싶은 만큼의 이미지를 쉽게 선택할 수 있도록 영역을 표시해준다.



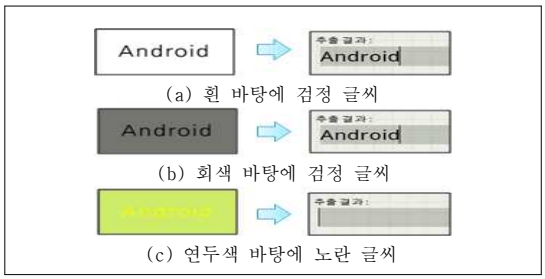
(그림 13) 텍스트 크롭 화면

6. 실험 및 분석

본 장에서는 이미지의 색과 텍스트 크롭에 대해 실험하여 OCR 서버가 추출, 인식할 수 있는 이미지의 특성이 무엇인지 분석한다.

6.1 색에 따른 실험 및 분석

그림 14는 이미지의 배경과 텍스트의 색상 차이에 따른 텍스트 추출 결과를 나타낸다. 그림 14(a)과 그림 14(b), 그림 14(c)은 각각 흰 바탕에 검정 글씨, 회색 바탕에 검정 글씨, 연두색 바탕에 노란색 글씨로 표현된 입력 이미지와 추출 결과를 나타낸다. 그림 14(a)과 그림 14(b)은 이미지 내의 텍스트와 추출 결과가 일치하는 반면, 그림 14(c)은 일치하지 않는다. 결과적으로, OCR 서버는 입력 이미지의 배경과 텍스트의 명도 차이에는 예민하지만 색상 차이에는 둔감하다는 것을 알 수 있다.



(그림 14) 색에 따른 텍스트 인식 결과

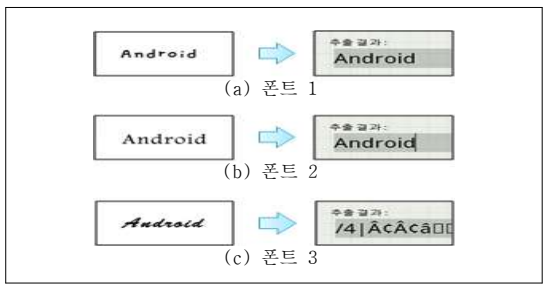
6.2 텍스트 크롭에 따른 실험 및 분석

그림 15는 텍스트 크롭 여부에 따른 텍스트 인식 결과를 나타낸다. 그림 15(a)과 그림 15(b)은 각각 텍스트 크롭을 실시하지 않을 때와 실시할 때의 결과 화면이다. OCR 서버는 이미지와 텍스트를 구별하지 못하므로 텍스트 크롭 기능을 통해 영역을 축소시키는 것은 텍스트 인식률 향상에 많은 영향을 미친다.

(그림 15) 텍스트 크롭에 따른 인식 결과

6.3 폰트에 따른 실험 및 분석

본 절에서는 여러 가지 폰트에 따른 텍스트 추출 결과를 비교하였다. 그림 16은 여러 가지 폰트에 따른 텍스트 추출 결과를 보여준다. 그림 16(a)와 그림 16(b)는 비교적 바른 글씨체이고, 자간이 충분히 넓다. 하지만, 그림 16(c)는 글씨가 기울어져 있으며 필기체로 쓰여 있으며 글자 사이에 빈 공간이 없어 부정확한 인식 결과가 나타난다. 따라서, 그림 16의 (c)와 같이 기울어져 있거나 흘려서 쓴 글씨는 잘 인식되지 않는다. 결과적으로 글씨체가 간단하고 자간이 뚜렷할수록 인식률이 높다.



(그림 16) 폰트에 따른 텍스트 인식 결과

7. 결론 및 향후 연구

본 논문은 사용자의 정보 접근성 증대를 위하여 이미지를 이용한 모바일 검색을 위한 콘텐츠인 Orthros를 제안하였다.

Orthros는 서버-클라이언트 구조를 갖는다. OCR 서버는 이미지에서 텍스트를 추출하기 위해 오픈 소스 OCR 엔진인 Tesseract를 사용하고, 모바일 클라이언트는 안드로이드 플랫폼 기

반의 스마트폰을 사용하는 모바일 어플리케이션이다.

Orthros의 기능은 이미지를 이용한 텍스트 검색과 검색내역 조회로 나뉜다. 이미지를 이용한 텍스트 검색 기능은 OCR 서버와 모바일 클라이언트가 연동하여 이미지에서 텍스트를 추출한 뒤, 그 텍스트를 상용 검색 엔진의 키워드로 입력하여 모바일 검색을 실시한다.

검색내역 조회 기능은 검색 내역과 세부 정보를 모바일 클라이언트의 데이터베이스에 저장하여 조회, 재검색이 가능하도록 하였다.

실험을 통해 OCR 서버가 인식할 수 있는 이미지의 특성을 분석하였다. 본 Orthros 시스템은 향후 다양한 상황에서의 정보 검색이 쉽게 이루어질 수 있도록 하여 모바일 환경에서의 정보 접근성을 극대화시킬 것이라 사료된다.

향후 연구로는 텍스트 크롭을 이용한 텍스트 인식의 구체적인 성능 연구가 진행될 예정이다.

참 고 문 헌

[1] 김기영, 강동호, “개방형 모바일 환경에서 스마트폰 보안기술,” *한국정보보호학회지*, Vol.19, No.5, pp. 21-28, 2009

[2] 장상규, 배건성, “음성 인식을 이용한 정보검색 시스템용 사용자 인터페이스 개발,” *대한전자공학회 학술대회 논문집*, Vol.9, No.1, pp. 607-610, 1996

[3] "doopedia," http://www.doopedia.co.kr/encyber/master/master.do?_method=view&MAS_IDX=101013000790664

[4] G. Chang, C. Tan, G. Li, and C. Zhu, "Developing Mobile Applications on the Android Platform," *Mobile Multimedia Processing: Fundamentals, Methods, and Applications*, Springer, 2010

[5] "Independent JPEG Group", http://www.ijg.org/

[6] J. He, R. Zhou, and Z. Hong, "Modified fast climbing search auto-focus algorithm with adaptive step size searching technique for digital camera," *IEEE Trans. On Consumer Electronics*, Vol.49, No.2, pp. 257-262, 2003

[7] LAB Asprise, "The Java Developer's Guide to Asprise OCR SDK 4.0," Asprise, 2007

[8] R. Smith, "An Overview of the Tesseract OCR Engine," in Proc. of *Intl' Conf. on Document Analysis and Recognition(ICDAR)*, 2007

[9] "tesseract-ocr," http://code.google.com/p/tesseract-ocr/

[10] R.Smith, "The Tesseract OCR Engine," http://conferences.oreillynet.com/presentations/os2007/os_raysmith.pdf, 2007

[11] “OCR - Optical Character Recognition," https://help.ubuntu.com/community/OCR

[12] "WeOCR Project," http://weocr.ocrgrid.org/

[13] "The Apache HTTP Server Project," http://httpd.apace.org/

[14] "PHP: Hypertext Preprocessor," http://www.php.net

[15] "Netpbm home page", http://netpbm.sourceforge.net/

[16] "project-O2 home page", http://www.imglab.org/p/O2/

고 은 비

2007년~현재 : 숙명여자대학교 멀티미디어과학과(학사과정)

관심분야 : 데이터베이스 시스템, 멀티미디어데이터베이스, 데이터 마이닝, 영상 미디어, 디지털 영상 처리

하 유 진

2007년~현재 : 숙명여자대학교 멀티미디어과학과(학사과정)

관심분야 : 데이터베이스 시스템, 멀티미디어데이터베이스, 모바일 소프트웨어, 컴퓨터 그래픽스

최 수 렴

2007년~현재 : 숙명여자대학교 멀티미디어과학과(학사과정)

관심분야 : 데이터베이스 시스템, 웹 서비스, 정보보호, 모바일 웹 서비스

이기훈

1996년 3월~2000년 2월 KAIST 전산학과 (학사).
2000년 3월~2002년 8월 KAIST 전산학과 (석사).
2002년 9월~2009년 2월 KAIST 전산학과 (박사).
2009년 3월~2010년 1월 KAIST 정보전자연구소 박사후 연구원.
2010년 2월~현재 : KT 종합기술원 중앙연구소 선임연구원.
관심분야 : XML 데이터베이스, 정보 검색, 질의 최적화

박 영 호

1992년 : 동국대학교 공과대학 컴퓨터공학과(석사)
2005년 : 한국과학기술원 전산학과 (공학박사)
1993년～1999년 : 한국전자통신연구원(ETRI) 교환전송연구단 선임연구원
2001년～2006년 : 한국산업기술대학교(KPU) 컴퓨터공학과 겸임교수
2005년～2006년 : 한국과학기술원 첨단정보기술연구센터 연구원
2005년～2006년 : 동국대학교 컴퓨터멀티미디어학과 겸임교수
2006년～현재 : 숙명여자대학교 이과대학 멀티미디어과학과 조교수
관심분야 : 데이터베이스, XML, IR(정보검색), 멀티미디어데이터베이스, Bio정보공학, 영상미디어, 예술&공학인터페이스