Korean Journal of Remote Sensing, Vol.27, No.1, 2011, pp.33~41

위성영상정보 기반 코너 포인트 객체 추출 안드로이드 스마트폰 앱 개발

강상구 · 이기원†

한성대학교 정보시스템공학과

Development of Android Smartphone App for Corner Point Feature Extraction using Remote Sensing Image

Sanggoo Kang and Kiwon Lee†

Dept. of Information Systems Engineering, Hansung University

Abstract : In the information communication technology, it is world-widely apparent that trend movement from internet web to smartphone app by users demand and developers environment. So it needs kinds of appropriate technological responses from geo-spatial domain regarding this trend. However, most cases in the smartphone app are the map service and location recognition service, and uses of geo-spatial contents are somewhat on the limited level or on the prototype developing stage. In this study, app for extraction of corner point features using geo-spatial imagery and their linkage to database system are developed. Corner extraction is based on Harris algorithm, and all processing modules in database server, application server, and client interface composing app are designed and implemented based on open source. Extracted corner points are applied LOD(Level of Details) process to optimize on display panel. Additional useful function is provided that geo-spatial imagery can be superimposed with the digital map in the same area. It is expected that this app can be utilized to automatic establishment of POI (Point of Interests) or point-based land change detection purposes.

Key Words : Smartphone, Open Source, Remote Sensing Images, App, Harris Algorithm, Corner Point.

요약 : 최근 국내외적으로 인터넷 웹에서 스마트폰 앱으로 정보통신기술 사용자 요구와 개발 환경이 변화되고 있어 공간정보 분야에서도 이에 따른 기술적 대응이 요구되고 있다. 그러나 현재의 수준은 스마트폰 지도서비스와 위치 확인 서비스가 주가 되고 있어 공간정보 콘텐츠 서비스를 위한 스마트폰 앱의 개발은 전 세계 기술 개발 동향을 고려하더라도 시험 개발의 초기 단계로 볼 수 있다. 본 연구에서는 공간영상정보를 활용하여 코너 포인트 객체 (Corner Point Feature) 추출 및 DB 연동 처리 기능을 제공하는 앱을 개발하였다. 이때 코너 포인트 객체 추출은 Harris 알고리즘을 적용하였으며, 데이터베이스 서버와 어플리케이션 서버, 사용자 환경으로 구분한 기본적인 시스템 환경의 모든 처리 모듈은 오픈소스 기반으로 설계 및 구현하였다. 추출되는 코너 포인트는 사용자 요구사항에 따라 화면 확대, 축소에 따라 상세화(Level of Details) 과정을 거쳐 화면에 최적화하도록 설계하였다. 한편 공간영상정보와 동일한 대상 지역의 수치지도가 있는

접수일(2011년 2월 5일), 수정일(1차 : 2011년 2월 22일), 게재확정일(2011년 2월 22일).

† 교신저자: 이기원(kilee@hansung.ac.kr)

경우에는 앱 상에서 수치지도 레이어를 중첩 표현할 수 있는 추가 기능을 제공하도록 하였다. 본 연구에서 추출되는 자동 POI (Point of Interests) 설정이나 포인트 객체 기반 국토변화 탐지에 적용이 가능할 것으로 예상한다.

1. 서 론

스마트폰과 모바일 애플리케이션, 태블릿 PC 등과 같이 일상에서 누구나 접하게 되는 정보통신기술은 간단하게 모바일이라는 주요어로 요약할 수 있으며, 동시에는 이는 가장 핵심적인 기술 동향이라고 할 수 있다 (한국정보화진흥원, 2011).

이는 모바일 단말기의 보편적인 사용으로 인터넷 웹에서 스마트폰 앱으로 정보통신기술 사용자 요구사항이 변하고 있음을 의미하는 것이기도 하다. 관련 산업 분야로 스마트폰 지도서비스와 위치 확인 서비스 등과 직접 또는 간접적으로 연계되는 스마트폰 앱은 나름대로의 산업 시장도 형성하고 있다. 그러나 이러한 서비스는 기본적으로 공간정보 데이터 자체와 실시간 위치 정보에 기반하는 것으로 데이터 자원에 대한 접근과 관련된 매핑 서버의 운영도 구글 등과 같은 정보 제공자의 매핑 서버에 의존하는 것으로 어떤 기관이 자체적인 서버 운영을 운영하여 고유한 장보자원을 제공하거나 또는 특정한 활용 분야에 이용되는 목적에 부합되는 공간정보 콘텐츠 서비스를 위한 스마트폰 앱의 개발은 국제적으로도 시험 개발의 초기 단계로 볼 수 있다. 즉, 지도 정보 자원의 가시화 기능을 위주로 하는 공간정보 서비스 앱은 대중적인 관심을 끌고 지도 정보의 대중화에는 크게 기여하고 있지만 공간영상정보 등과 같은 잠재적 고부가 정보자원을 이용하여 분석기능을 제공하는 전문적인 모바일 앱은 개발 성과가 보고된 사례가 거의 없다.

공간정보 콘텐츠를 처리하는 모바일 앱은 모바일 GIS와 유사하게 인식될 수도 있다. 모바일 GIS는 GNSS (Global Navigation Satellite System) 위치정보에 기반한 정밀도가 높은 공간정보의 수집, 무선 인터넷 연결, 매핑 지원 등을 주요 기능으로 하고 있다. 모바일 GIS 소프트웨어 제품도 이미 ESRI 사의 iPhone 용 모바일 SW, Astech 사의 MobileMapper, Leica Geosystem사의 Zeno 10 등과 같이 상업적 활용 단계에 있는 경우도 있다(Lemmens, 2010). 그러나 본 연구에서는 공간정보 서비스 스마트폰 앱 또는 모바일 앱의 개념은 모바일 GIS와는 다소 차이가 있다는 점을 전제로 하고자 한다. 첫 번째 차이점으로는 스마트폰 앱은 GIS의 고유 기능 보다는 특정한 활용 목적을 지향해야 한다는 점이다. 두 번째는 많은 복잡한 기능보다는 직관적인 사용자 인터페이스를 통하여 모바일 단말 환경에 최적화된 정보를 서비스해야 한다는 점이다.

따라서 본 연구에서는 '특정한 활용 목적' 측면에서는 위성영상정보 기반 코너 포인트 추출 기능을 위주로 하고, '최적화 서비스' 측면에서는 추출된 포인트 객체의 상세화 (LOD: Levels of Details) 기능을 설계 구현하고자 한다. 시스템 구성은 특정한 저작권이 있는 소프트웨어가 아닌 오픈 소스 기반의 개방형 구조로 설계하고, 실험에 필요한 매핑 서버는 특정한 공공 서버가 아닌 시험용 서버를 사용하였다. 스마트폰 개발 플랫폼 중에서는 안드로이드(Android)를 적용하였다. 이는 안드로이드 운영체계가 다양한 앱의 개발을 지원할 수 있도록 계속 발전되고 있으므로 향후 확장성(Chen, *et al.*, 2011)과 개방성을 중시하였기 때문이다. 한편 공간정보 처리 오픈 소스로는 강과 이(2010)의 연구와 같이 Orfeo Toolbox(OTB)와 gvSIG Mini를, 데이터베이스 시스템은 PostgreSQL/PostGIS을 이용하였다.

2. 개발 환경 및 시스템 구성

안드로이드 플랫폼은 기본적인 런타임 핵심 라이브러리와 함께 미디어 프레임워크, SQLite, OpenGL ES, SGL, SSL, Free Type, Web Kit 등과 같은 표준 라이브러리를 제공한다. 개발 도구로는 Android Debugging Bridge, Android Development Tools Plugin, Android Asset Packaging Tool, Android Interface Definition Language 등을 제공하고 다양한 에뮬레이터를 지원하고 있다. 본 연구에서는 안드로이드 버전 2.2 기반으로 설계 및 구현 작업을 수행하였다.

위성영상 정보처리 및 분석을 위한 기본 엔진으로는 OTB를 적용하였다. OTB(http://www.orfeo-

toolbox.org)는 C++을 이용해서 개발되었고, Orfeo Wrapping을 통하여 Python, Java, Ruby 등과 같은 개발 언어를 지원하므로 어플리케이션 확장성이 우수한 오픈 소스이다(Guzzonato *et al.* 2010; OTB Development Team, 2010). OTB는 Kang and Lee (2010(a), 2010(b))의 기존 개발 연구에서도 적용된 바 있으며, 제공하는 기능이나 개발 지원 환경이 공간위성 영상정보 처리나 분석에는 충분히 적합한 것으로 나타났으며 본 연구에서는 OTB 3.8 버전을 이용하였다.

gvSIG Mini (http://confluence.prodevelop.es/display/GVMN/Home)는 Prodevelop사에서 개발한 스마트폰에서 구동되는 앱 개발자용 공간정보 브라우저 기능을 수행하도록 설계된 오픈 소스이다(Carrasco and Romeu, 2009). 한편 본 연구에서 적용된 기타 오픈 소스 및 개발 및 운영 환경은 Table 1에 정리하였다. 처리 결과 영상의 저장과 관리를 위한 데이터베이스는 다양한 프로그래밍 언어를 위한 인터페이스를 제공하는 오픈 소스 DBMS인 PostgreSQL/PostGIS(http://www.postgresql.org)를 사용하였고, 본 개발 과정에서는 Python 인터페이스인 psycopg2를 이용하였다. 영상 데이터베이스 구축을 위한 PostgreSQL/PostGIS와 어플리케이션 서버로서 영상 처리 및 분석을 수행하는 OTB 영상 처리 오픈소스를 연계하는 방법은 Kang and Lee(2010(a))의 연구 결과를 사용하였다.

Table 1. Operation and development environment based on open sources

	Environment	Version
Server-side	Operating System	Fedora 12
	Orfeo Toolbox	3.8
	GDAL	1.7.2
	PostgreSQL	8.4.4
	PostGIS	1.4.2
	psycopg2	2.2.2
	Apache	2.2.13
	Python	2.6.2
Client-side	Operating System	Android 2.2
	gvSIG mini	0.2.0
	Java JDK	1.6.0_18

Fig. 1은 전체적인 시스템 구성도이고 단위 모듈로 사용한 오픈 소스를 같이 전개한 것이다. 기본적인 시스템 구성은 Kang and Lee(2010(b))와 같은 기존의 연구에 기반을 두고 있으나, 본 연구에서는 공간영상정보를 이용하여 코너 포인트 객체 추출을 수행하는 Harris Detection 모듈을 추가하였고 처리 결과를 데이터베이스에 객체로 저장 및 관리한다. 또한 구글에서 개발하고 OGC에서 공간정보 웹 표준으로 인정한 KML 유형(Wernecke, 2009)으로 선이나 포인트 객체를 벡터 레

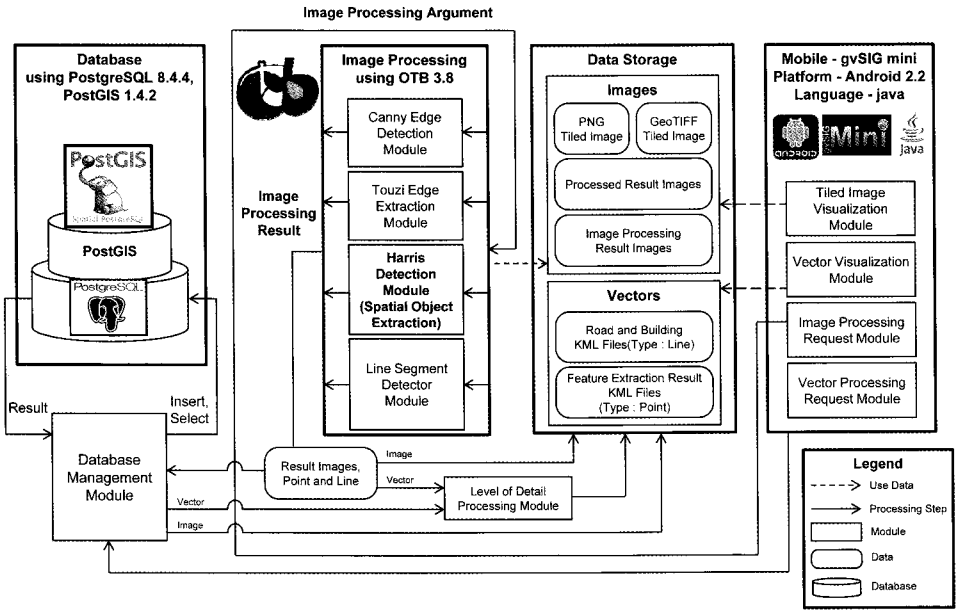


Fig. 1. System architecture and its components related to open sources.

이어를 가공하도록 한 점에서 차이가 있다.

스마트폰 단말에서 어플리케이션 서버에 요청하는 영상 처리 기능은 호출시에만 구동하는 On-the-fly 방식이므로 Canny Edge 필터나 Touzi Edge 필터 처리 기능 등과 같은 다른 기능은 해당 기능을 요청하는 경우에만 호출되어 단위 기능 수행시에는 영향을 주지 않는다.

3. 코너 포인트 추출

영상 기반 코너 포인트 추출은 영상 처리, 패턴 인식, 컴퓨터 비전 등 다양한 분야에서 영상 정합(Matching)과 접합(Mosaicing), 모션 감지 및 추적, 파노라마 영상 제작, 3차원 모델링과 객체 인식 등과 같은 응용 분야에서 활용된다. 따라서 이러한 코너 포인트의 자동 추출을 위한 알고리즘 개발도 중요한 연구 주제중의 하나이다 (Mehrotra and Nichani, 1990; Tomasi and Kanade, 2004; Trujillo and Olague, 2008; Zhang *et al.*, 2010).

OTB에서도 몇 가지 포인트 객체 추출 알고리즘을 제공하고 있는 데, Lahlou *et al.*(2009)은 OTB에서 제공하는 Harris, SIFT (Scale Invariant Feature Transformation), SURF (Speed Up Robust Features)등과 같은 세 가지의 포인트 추출 알고리즘을 비교한 바 있다.

본 연구에서는 이 중에서 모바일 단말 성능 등을 고려하여 Harris 알고리즘을 적용하였으며, Harris 알고리즘은 다음과 같이 정리할 수 있다(Harris and Stephens, 1988).

영상 정보를 구성하는 화소 전개 방향을 기준으로 하여 X, Y 방향에 따른 변이량의 세기 함수 또는 구배 강도 함수를 $E(u,v)$라고 할 때, 윈도우 함수 $w(x,y)$ 를 적용하여 표현하면 식(1)과 같다. 한편 $E(u,v)$는 영상 화소의 변량에 대한 식 (3)과 같은 2×2 행렬 M을 이용하여 식 (2)와 같은 쌍 선형 근사치(Bilinear approximation) 함수로 표현할 수 있다. Harris 알고리즘에서는 윈도우 함수를 가우스 함수로 이용하게 된다.

여기서 M에 대한 λ_1, λ_2와 같은 고유치(Eigen value)를 식 (4)와 같이 구하면, 이는 식 (5)와 같은 코너에 대응하는 측정치 R에 대한 함수로 표현할 수 있다. 코너 포인트는 λ_1과 λ_2의 국지적 최대치가 되는 경우에 해당한다.

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2 \quad (1)$$

$$E(u,v) \cong [u,v]\ M \begin{bmatrix} u \\ v \end{bmatrix} \quad (2)$$

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (3)$$

$$\begin{aligned} \det M &= \lambda_1 \lambda_2 \\ \text{trace } M &= \lambda_1 \lambda_2 \end{aligned} \quad (4)$$

$$R = \det M - k(\text{trace } M)^2 \quad (5)$$

4. 스마트폰 사용자 인터페이스와 구현 결과

본 연구에서는 Harris 알고리즘에 기반한 OTB 오픈 소스를 스마트폰 앱으로 제공하는 것이 우선적인 목적이므로 스마트폰의 주요 특징중 하나인 직관적인 사용자 인터페이스 설계와 시험 적용 결과를 제시하고자 한다.

Fig. 2는 본 연구에서 설계된 스마트폰 사용자 인터페이스를 보여주는 것으로, 처리 기능의 복잡도를 고려하여 주메뉴와 부메뉴로 구분하였다. 주메뉴에서 벡터 시각화, 데이터베이스, 영상처리를 선택하면 각 유형의 부메뉴를 사용할 수 있는 데, 이는 데이터베이스에 구축된 벡터 레이어를 요청하여 위성영상에 중첩 표현할 수 있도록 하는 메뉴, 데이터베이스로부터 처리된 결과 정보를 불러 올 수 있도록 하는 메뉴와 코너 포인트 추출이나 기타의 영상처리를 어플리케이션에 요청 처리할 수 있도록 하는 메뉴 등이다.

Fig. 3은 이 중에서 벡터 레이어의 호출에 따른 데이터 흐름도를 도시한 것으로 데이터베이스 서버에 구축된 수치지도 기반 벡터 레이어는 웹 표준 중 하나인 KML 유형으로 변환되어 앱 상에 표현된다.

Fig. 4는 코너 포인트 객체 추출을 위한 사용자 인터페이스와 입력 변수, 중간 처리 결과 등을 보여주는 것이다. 사용자 인터페이스에서는 추출된 포인트를 영상화하여 보여주는 버튼 기능과 핀 포인트 형식으로 포인트 객체 만을 배경화면위에 중첩하여 가시화하는 버튼 기능을 제공한다. 입력 변수에서는 sigmaD와 sigamI를 설정하

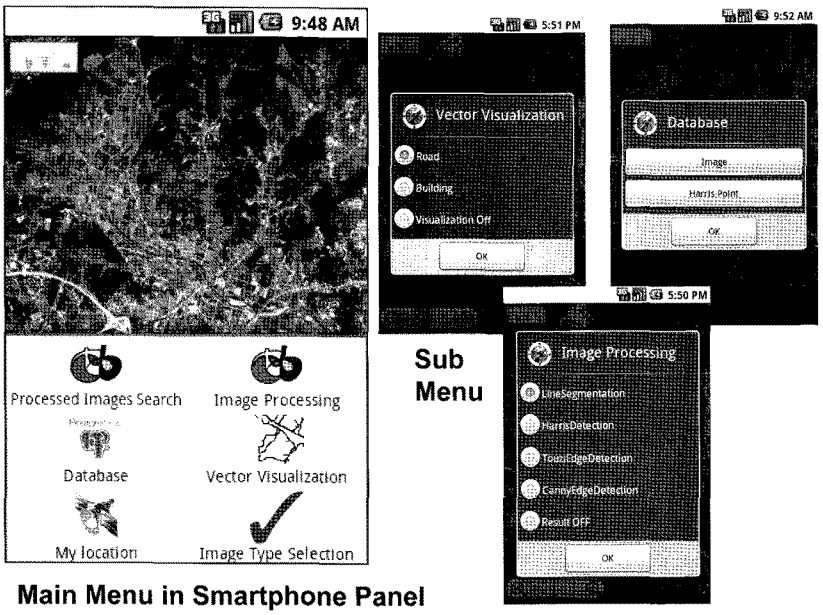


Fig. 2. Main menu and sub menu in app developed in this study.

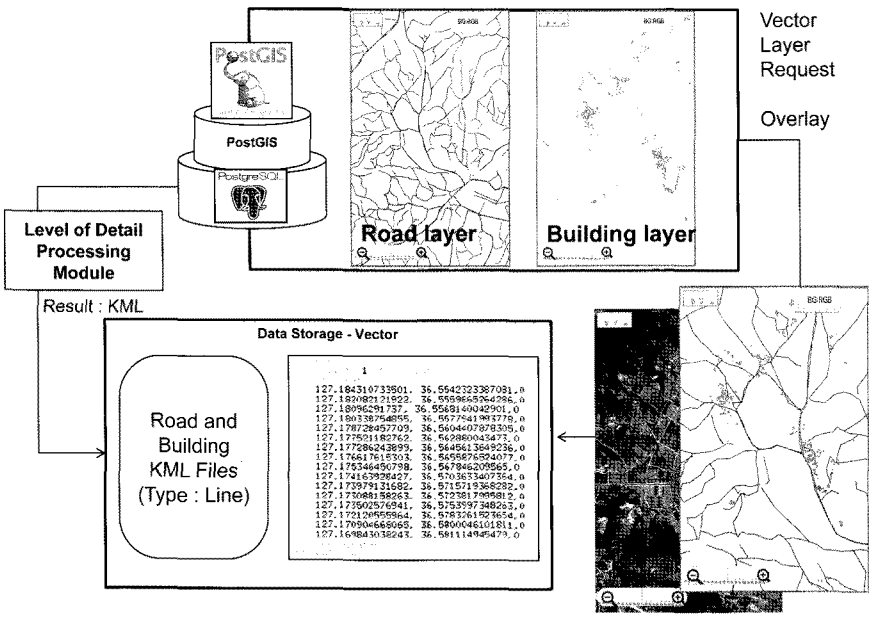


Fig. 3. Data flow in KML related to vector layer request and overlay processing.

도록 하였는 데, 이는 Harris 알고리즘에서 요구하는 변수 값으로 각각 Derivation scale과 Integration scale을 나타내며 가우시안 커널(Gaussian Kernel)함수 결정에 이용된다. 또한 하위 임계치 (Lower threshold)와 상위 임계치 (Upper threshold)라는 두 개의 임의 설정 임계치를 사용자가 입력하도록 하였는 데, 이는 식(5)에서 얻어지는 코너 포인트 객체의 개수를 사용자가 설정하도록 하는 값이다. Fig. 5는 Harris 알고리즘과 임계치를 연계하여 어플리케이션 서버에 처리 과정을 요구하는 오픈 소스 기반 Python 코드의 예시로, 사용자가 설정한 값을 입력하여 Harris 알고리즘을 실행하면, 결과로 각 TIFF 포맷 타일링 영상의 POI를 텍스트 파일

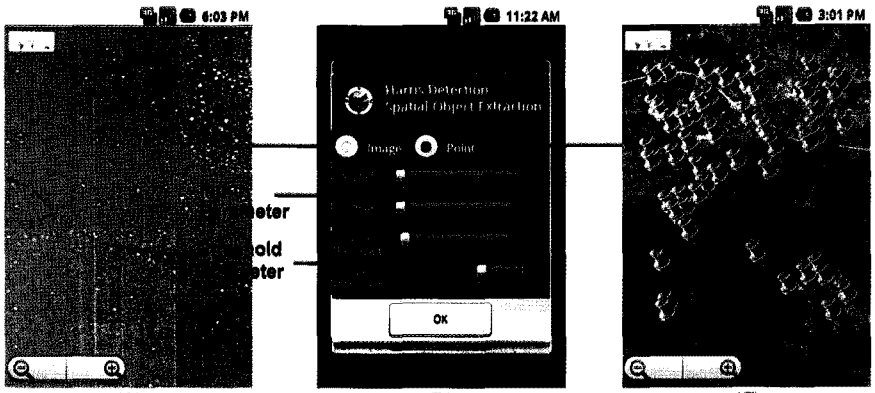


Fig. 4. Corner point extraction in app, and controlling interfaces for input parameters of Harris algorithm.

python harrisPointLODprocessing.py - pseudocode

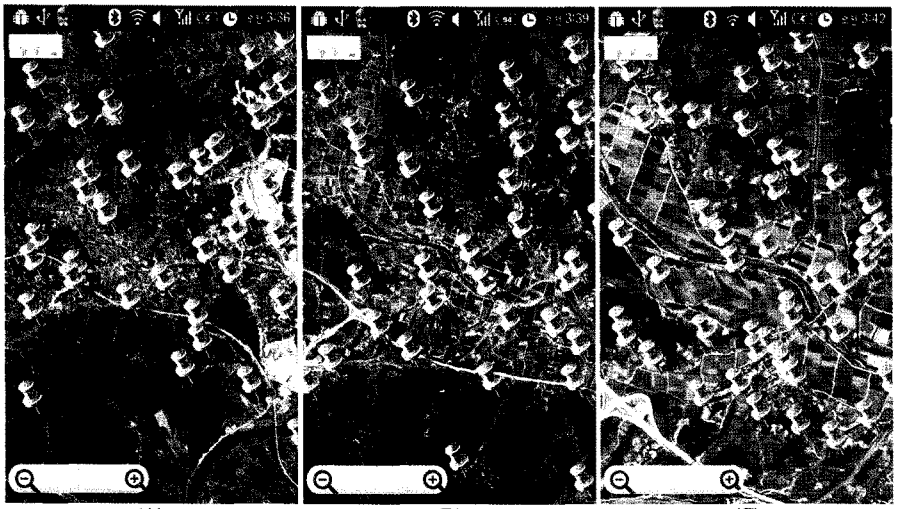
```
01 def harrisLOD(int ixp, int iyp, int depth):
02   p = Proj(proj='utm',zone=52,ellps='WGS84')
03   geoString_List_big = []
04   f = open("/var/www/html/spatialObject/harrisPoint"+depth+"_"+str(ixp_i)+"_"+str(iyp_i)+".txt")
05   rl = f.readlines(); sizeRl = len(rl)
06   pointString_x_List=[ ]; pointString_y_List=[ ]; geoString_List=[ ]; pointString="; pointString_x="; pointString_y=""
07   k = 0
08   for i in range(0, sizeRl):
09     pointString = rl[i]; xN = pointString.find(','); yN = xN+2
10     if int(depth) == 12:
11       if k%200 == 0: pointString_x_List.append(pointString[1:xN]); pointString_y_List.append(pointString[yN:-2])
12     if int(depth) == 13:
13       if k%150==0 or k%200==0: pointString_x_List.append(pointString[1:xN]); pointString_y_List.append(pointString[yN:-2])
14     if int(depth) == 14:
15       if k%100 == 0: pointString_x_List.append(pointString[1:xN]); pointString_y_List.append(pointString[yN:-2])
16     if int(depth) == 15:
17       if k%50 == 0: pointString_x_List.append(pointString[1:xN]); pointString_y_List.append(pointString[yN:-2])
18     k = k+1
19     f.close()
20     for i in range(0, len(pointString_x_List)):
21       os.system("gdal2xyz.py -srcwin "+pointString_x_List[i]+" "+pointString_y_List[i]+" "+" 1 1 "+"/var/www/html/tif/"
                +depth+"/"+str(ixp)+"/"+str(iyp)+".tif /var/www/html/spatialObject/geoPoint/geoPoint"+str(i)+".txt")
22       openString = "/var/www/html/spatialObject/geoPoint/geoPoint"+str(i)+".txt"
23       f1 = open(openString); geoString_temp = f1.readline()
24       bN1 = geoString_temp.find(' '); bN2 = geoString_temp.find(' ', 20, -1)
25       geoString_x = geoString_temp[:bN1]; geoString_y    = geoString_temp[bN1:bN2]
26       ix, iy = p(geoString_x, geoString_y,inverse=True)
27       geoString = str(ix)+","+str(iy)+", 0"; geoString_List.append(geoString); geoString_List_big.append(geoString)
28       f1.close()
29       f2 = open("/var/www/html/kml/harrisPoint.kml", "w")
30       f2.write("""<?xml version="1.0" encoding="utf-8"?><kml xmlns="http://www.opengis.net/kml/2.2"><Document>
                <Placemark><MultiGeometry>₩n""")
31       for i in range(0, len(geoString_List_big)):
32         f2.write("    <Point>₩n"); f2.write("      <coordinates>₩n"); f2.write("      "+geoString_List_big[i]+"₩n")
33         f2.write("    </coordinates>₩n"); f2.write("    </Point>₩n")
```

Fig. 5. Point LOD processing pseudo-code in Python.

로 저장되고, 사전에 TIFF 포맷 타일링 영상과 동일하게 타일링된 GeoTIFF 포맷 영상으로부터 실좌표를 추출하게 된다. 예시 코드는 이중에서 텍스트 파일에서부터 실좌표를 얻을 때 LOD를 적용시켜 추출하는 것을 나타내고 있다. 실좌표를 추출할 때 사용자가 현재 보고 있는 줌 레벨에 따라서 추출하는 포인트의 개수를 임의로 지정하여 KML 파일로 저장시키는데 그 이유는 줌 레벨에 따라서 적절한 개수의 포인트를 표현하고, 스마트폰 상에서 벡터 포인트를 표현할 때 속도를 향상시키기 위함이다.

Fig. 6은 스마트폰 화면에서 사용자의 확대, 축소 단계에서 따라 추출된 코너 포인트 객체를 상세화 단계에 따라 표현하는 예시이다.

Fig. 7은 추출된 포인트 객체를 데이터베이스와 연동하여 처리하는 과정을 제시한 것으로 (A) 특정 포인트 객체의 선택, (B) 객체 이동, (C) 특정 포인트 객체의 삭제, (D) 포인트 객체 속성의 입력과 속성 보기, (E) 데이터베이스 POI 테이블의 생성과 (F) 데이터베이스 POI 테이블의 삭제 등을 보여주는 것이다.



(A) (B) (C)

Fig. 6. Point feature optimization according to zoom levels.

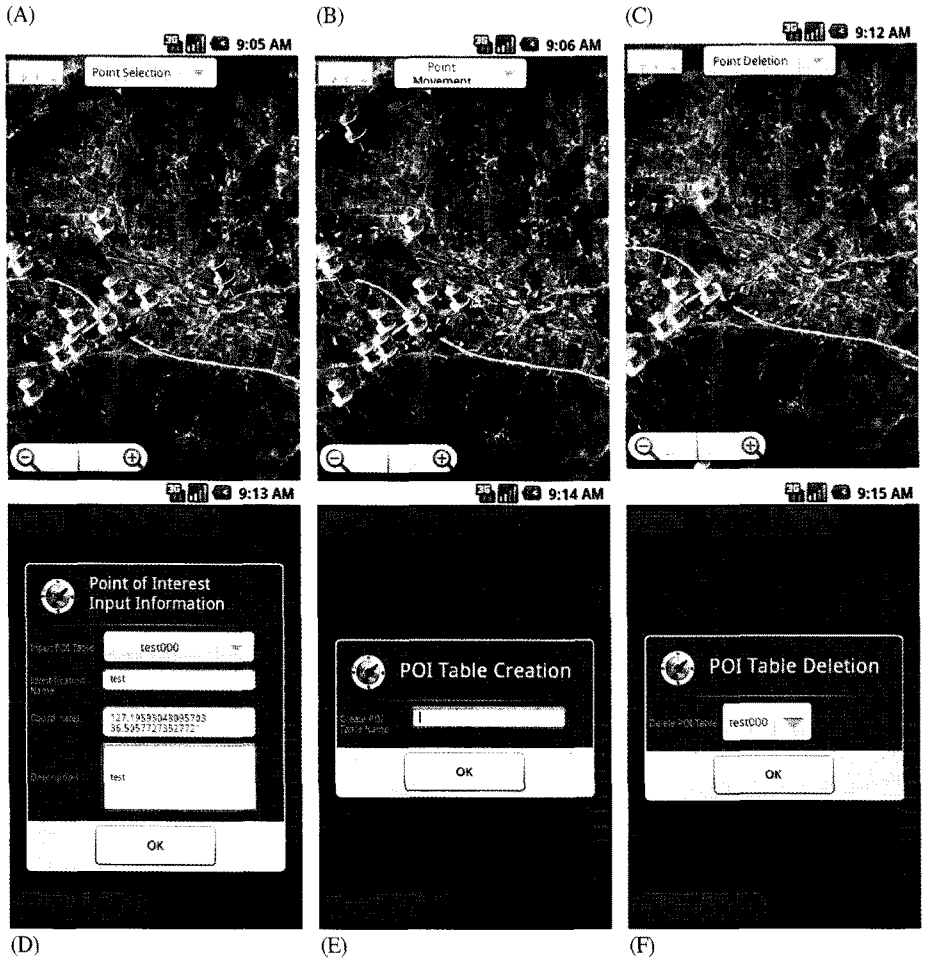


Fig. 7. DB connecting interfaces: (A) Point selection of red pin point, (B) Point movement of red pin point, (C) Point deletion, (D) Input and view of Point attribute, (E) Creation DB POI table, and (F) Deletion DB POI table.

5. 결 론

본 연구에서는 공간영상정보를 활용하여 코너 포인트 객체 추출 및 DB 연동 처리 기능을 제공하는 안드로이드 스마트폰에서 구동되는 앱을 설계, 개발하였다. 코너 포인트 객체 추출은 Harris 알고리즘을 적용하였으며, 데이터베이스 서버와 어플리케이션 서버, 사용자 환경으로 구분한 기본적인 시스템 환경의 모든 처리 모듈은 오픈소스 기반으로 설계 및 구현하였다. 추출되는 코너 포인트는 직관적인 사용자인터페이스를 통하여 화면 이동, 확대, 축소 등의 요청에 따라 데이터베이스에 저장된 추출된 코너 포인트 객체를 화면에 최적화하여 도시할 수 있도록 하는 상세화 기능을 제공한다. 추가적으로는 선형이나 면형 객체 등과 같은 벡터 레이어를 포인트 객체와 같이 KML 유형으로 가공하여 공간영상정보에 중첩하여 표현할 수 있도록 하였다. 이러한 연구 결과는 수치지도 정보와 공간영상 정보를 데이터 서버에 구축하고, 사용자에게 이러한 공간정보 자원에 접근하여 필요한 가공하나 분석을 수행하도록 하는 스마트폰 앱 서비스를 제공하고자 하는 경우에 적용될 수 있는 모델이다. 특히 본 연구에서 주안점을 둔 공간영상정보 기반 코너 포인트 객체는 사용자가 특징 지역을 관심지점(POI)으로 설정하고자 하는 경우 사전에 공간영상에 나타나는 특징 지점을 자동으로 추출해 주고 사용자가 원하는 POI 만을 저장, 관리할 수 있도록 하는 기능을 제공하기 때문에 스마트폰 상에서 구동되는 포인트 객체 기반 국토변화 탐지 응용 시스템 개발시에도 적용이 가능할 것으로 예상한다.

사 사

본 연구는 국토해양부 첨단도시기술개발사업 - 지능형국토정보기술혁신 사업과제의 연구비지원(과제번호: 07국토정보C03)에 의해 수행되었습니다.

참고문헌

강상구, 이기원, 2010. 위성영상정보 분석을 위한 안드로이드 스마트폰 앱 개발, 대한원격탐사학회지, 26: 561-570.

한국정보화진흥원, 2011. 2011년 IT 트렌드 전망 및 정책방향, IT 정책연구시리즈.

Carrasco, J. and A. Romeu, 2010. gvGIS Mobile and Mini, FOSS4G 2010, Tutorial T-01.

Chen, M.-C., J.-L. Chen, and T.-W. Chang, 2011. Android/OSGi-based vehicular network management system, *Computer Communications*, 34: 169-183.

Guzzonato, E., C. Valladeau and J. Inglada, 2009. Orfeo ToolBox: An Open Source Tool for High Resolution Image Processing, *GeoCap Meeting*.

Harris, C. and M. Stephens, 1988. A combined corner and edge detector, *Proceedings of the 4th Alvey Vision Conference*, 147-151.

Kang, S. and K. Lee, 2010(a). Open Source Remote Sensing of ORFEO Toolbox and Its Connection to Database of PostGIS with NIX File Importing, *Korean Journal of Remote Sensing*, 26: 361-371.

Kang, S. and K. Lee, 2010(b). Smart Phone Application Development for Thematic Geo-spatial Image Handling, *Proceedings on 2010 International Symposium on Remote Sensing*.

Lemmens, M., 2010. Mobile GIS Systems, *GIM International*, 12: 21-27.

Mehrotra, R. and S. Nichani, 1990. Corner detection, *Pattern Recognition*, 23: 1223-1233.

OTB Development Team, 2010. *The ORFEO Tool Box Software Guide, Updated for OTB-3.8*, 670p.

Tomasi C. and T. Kanade, 2004. Detection and Tracking of Point Features, *Pattern Recognition*, 37: 165-168.

Trujillo L. and G. Olague, 2008. Automated design of image operators that detect interest points, *Evolutionary Computation*, 16(4): 483-507.

Wernecke, J., 2009. *The KML Handbook*, Google.

Zhang, X., A. W. B. Smith, X. Ling, B. C. Lovell, and

D. Yang, 2010. Corner detection based gradient correlation matrices of palnar curves, *Pattern Recognition*, 43: 1207-1223.

Lahlou, O., J. Michel, D. Pichard, and J. Inglada, 2009. Assessment of interest points detection algorithms in OTB, *IGARSS* 2009.