KIPS Tr. Software and Data Eng.
Vol.5, No.8 pp.369~376 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2016.5.8.369

Vehicle Detection Method Based on Object-Based Point Cloud Analysis Using Vertical Elevation Data

Junbeom Jeon† · Heezin Lee†† · Sangyoon Oh††† · Minsu Lee††††

ABSTRACT

Among various vehicle extraction techniques, OBPCA (Object-Based Point Cloud Analysis) calculates features quickly by coarse-grained rectangles from top-view of the vehicle candidates. However, it uses only a top-view rectangle to detect a vehicle. Thus, it is hard to extract rectangular objects with similar size. For this reason, accuracy issue has raised on the OBPCA method which influences on DEM generation and traffic monitoring tasks. In this paper, we propose a novel method which uses the most distinguishing vertical elevations to calculate additional features. Our proposed method uses same features with top-view, determines new thresholds, and decides whether the candidate is vehicle or not. We compared the accuracy and execution time between original OBPCA and the proposed one. The experiment result shows that our method produces 6.61% increase of precision and 13.96% decrease of false positive rate despite with marginal increase of execution time. We can see that the proposed method can reduce misclassification.

Keywords : LiDAR Data, Vehicle Detection, OBPCA, Vertical Elevation

OBPCA 기반의 수직단면 이용 차량 추출 기법

전 준 범† · 이 희 진†† · 오 상 윤††† · 이 민 수††††

요 약

점 클라우드로부터 차량을 추출하는 다양한 방식 중 OBPCA 방식은 세그먼트 단위의 평가-분류로 정확도가 높고, 단순한 직사각형 평면도에서 특성 값들을 추출하므로 분류가 빠르다. 그러나 이 OBPCA 방식은 차량과 크기가 비슷한 직육면체 모양의 물체를 차량과 구별하지 못하는 문제를 가지므로 이를 극복하고 차량 추출의 정확도를 높이는 방안에 대한 연구가 필요하다. 본 논문에서는 이 문제를 해결하기 위해 수평 단면과 함께 수직 단면을 이용하는 확장 OBPCA 방식을 제안한다. 제안 방법은 수평 단면을 통해 차량 후보를 1차로 선별하고, 각 차량 후보에서 가장 특징적인 수직 단면을 찾아서 그 단면의 특성 값들을 임계값들과 비교하여 차량 여부를 판단한다. 비교실험에서는 본 제안방식이 기존 OBPCA 방식에 비해 정밀도가 6.61% 향상되고 위양성률이 13.96% 감소됨을 확인했으며, 이를 통해 제안 방식이 기존 OBPCA 분류오류 문제에 대해 효과적인 해결방안임을 보였다.

키워드 : 라이다 데이터, 차량 추출, OBPCA, 수직 단면

1. 서 론

리모트 센싱과 관련된 여러 기술 중 LiDAR(Light Detection And Ranging)는 고해상도의 점 클라우드를 제공하며 지도 제작, 물체 탐지, 재난 조사 등에 활용되고 있다. LiDAR는 DTM (Digital Terrain Model)이나 건물, 나무, 도로와 같은 물체들을 추출할 수 있는 고가치의 데이터 소스로서, 최근에 이 LiDAR 데이터 분류에 패턴 인식과 컴퓨터 비전에 기반을 둔 새로운 방법론들이 제시되고 있다. 물체 추출 작업 면에서는, 지식 기반 방법 외에 AdaBoost 알고리즘[1]이나 MPP (Marked Point Process)같이 데이터 기반 학습 방법을 다루는 연구들의 비중이 증가하고 있는데, 이러한 트렌드는 데이터 기반 학습 방법이 더 유연성 있는 결과를 가져온다는 사실에 기반을 두고 있다[2].

LiDAR 점 클라우드로부터 물체 추출을 위해서는 우선 이미지 데이터를 특정 이미지 표현 단위로 나누어 특성 벡터로 표현한다. 이 특성 벡터를 입력 데이터로 사용하고 데

※ 본 논문은 2015년도 정부(미래창조과학부)재원으로 지원된 한국연구재단 신진연구자지원사업(2015R1C1A1A01054305), 정부(교육부)재원으로 지원된 한국연구재단 기초연구지원사업(NRF-2015R1D1A1A01059557), 그리고 미래창조과학부 및 정보통신기술진흥센터의 ICT/SW창의연구과정(SW중심대학) 지원사업의 연구결과로 수행되었음(R2215-15-1002).
† 비 회 원 : TmaxOS Office실 선임연구원
†† 비 회 원 : 미국 캘리포니아대학교 책임연구원
††† 종신회원 : 아주대학교 소프트웨어학과 부교수
†††† 종신회원 : 이화여자대학교 컴퓨터공학과 연구교수
Manuscript Received : February 3, 2016
First Revision : April 14, 2016
Accepted : April 15, 2016
* Corresponding Author : Minsu Lee(michelle.lee@ewha.ac.kr)

이터마이닝이나[3, 4] 지식 기반 전문가 시스템을 이용하여 이미지에 클래스 레이블을 할당해 주어 물체를 분류한다.

특별히 물체들 중에서도 차량을 찾아내는 기술은 DTM을 생성할 때 차량을 제거하기 위해서 빈번히 활용된다. DTM을 위한 필터링 단계에서는 차량 점들이 지면에 매우 가까우므로 차량과 지면을 구별하는 것이 어렵다. 게다가 차량은 종류도 다양하고 구조도 복잡하기 때문에, 차량 추출은 구체적인 필터링을 요구하여 현재에도 도전적인 과제가 되고 있다.

LiDAR 데이터로부터의 차량 추출에 대해서 많은 연구들이 진행되고 있는데, 그 중 OBPCA (Object-Based Point Cloud Analysis)[5]는 3차원 점들만을 사용하여 차량을 자동적으로 추출하므로, 도로 데이터베이스와 같은 외부 데이터 소스를 필요로 하지 않는다. 또한 세그먼트 단위로 평가하여 분류하기 때문에, 점 단위로 분류하는 것보다 정확도가 높으며[3], 단순한 직사각형 모양의 평면도에서 특성 값들을 추출하므로 분류가 빠르다. 그러나 차량과 크기가 비슷한 직육면체 모양의 물체를 차량과 구별하지 못하는 단점이 있으므로 차량 추출의 정확도를 높여야 할 필요가 있다.

본 논문에서는 차량 추출의 정확도를 높이기 위해 수평 단면과 함께 수직 단면을 이용하는 확장된 OBPCA를 제안한다. 제안 방법은 수평 단면을 통해 차량 후보를 1차로 선별하고, 각 차량 후보에서 가장 특징적인 수직 단면을 찾아 그 수직 단면의 특성 값들을 임계값들과 비교하여 차량 여부를 판단한다. 특성 값으로는 area, rectangularity, elongatedness를 사용하였으며, 1차로 선별된 차량 후보들의 수직 단면을 추출하기 위해 PCA (Principal Component Analysis)를 적용하였다. 수직 단면의 특성 값들도 임계값들과 비교하여 신속하게 차량을 판별하도록 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 LiDAR 점들로부터 차량을 추출하는 기존 방법들과, 수직 단면을 고려하여 물체를 분류한 연구를 소개한다. 3장에서는 OBPCA을 확장하여 수직 단면 특성까지 고려한 제안 방법을 설명한다. 4장에서는 기존 OBPCA와 제안 시스템과의 예측 성능 비교 평가 결과를 보이고 5장에서 본 연구의 결론을 정리하고 향후 연구 방향을 제시한다.

2. 관련 연구

2.1 차량 추출 기법

1) 그리드셀 기반 방법

LiDAR 데이터 처리를 위한 기존 제안 방법들 중 그리드셀 기반 방법은[6] LiDAR 점들에 2차원 격자를 적용한 뒤 격자 기반으로 점 클라우드를 처리하는 방법이다. 이 방법에서는, Fig. 1에서 보는 바와 같이, 지면, 건물, 초목과 같은 주변 물체들을 참고하여 차량을 추출한다.

Raw LiDAR 데이터를 이미지 처리 연산에 쓸 수 있는 그리드 데이터로 변형시키고, 점을 할당받지 못한 그리드 셀들은 곡면 적합(Surface fitting) 방법을 통해 채워나간다. 이후 그리드 데이터에 대하여 extended-maxima 변환을 통해 차량 지붕들을 찾고, marker-controlled 분수령 변환을 통해 최종적으로 차량을 추출한다.

그리드 셀 기반 방법은 정차해 있는 차량들을 잘 인식하지만, 외형과 이동 방향이 다양한 차량들 중에서 움직이는 차량을 탐지하고 속도를 추정하는 면에서는 3차원 점 클라우드 기반 방법보다 성능이 떨어진다. 또한 하나의 셀에 투사되는 여러 점들 중 하나의 점만 활용하기 때문에, 나머지 점들에 대한 정보 손실의 문제점을 가지고 있다.

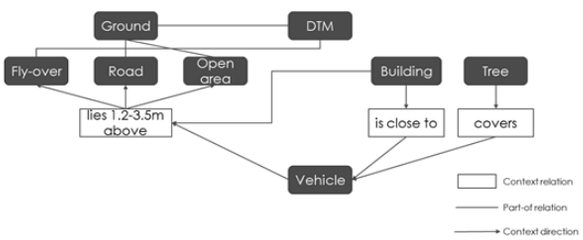


Fig. 1. 3D Vehicle Extraction Refering to Other Objects

2) OBPCA (Object-Based Point Cloud Analysis)

OBPCA는 LiDAR 데이터의 3차원 점들을 세그먼트 단위로 평가하여 세그먼트상의 특징 정보에 기반하여 차량을 분류한다. LiDAR 점 클라우드를 통해 시가지에서 나무를 추출한 연구에서는[7] 세그먼트 기반 분류가 점 기반 분류보다 정확도와 성능의 측면에서 더 좋은 결과를 가져온다는 실험 결과를 보여주었다. 점 기반 분류에 비해, 세그먼트를 이용하였을 때 이미지상의 다양한 특성들을 정의할 수 있고, 총 분류 횟수도 급격히 감소하기 때문이었다. 이후 차량을 추출하는 연구[8]에서는 점 클라우드를 다양한 물체들의 공간 확장을 근사하는 세그먼트들로 분할하는 AMS (Adaptive Mean Shift)에 기반을 둔 OBPCA 방법을 제안하였다. Fig. 2에서 OBPCA의 수행 절차를 확인할 수 있다.

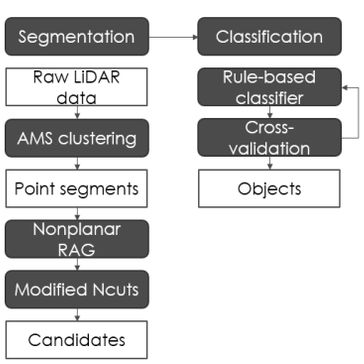


Fig. 2. AMS Based OBPCA

OBPCA는 필터링, 세그멘테이션, 추출 과정을 거쳐 차량을 추출한다. OBPCA의 차량 추출은 Fig. 3의 예를 통해 설명할 수 있다. Fig. 3의 (a)에 나와 있는 점 클라우드가 차량 후보 세그먼트 한 개의 평면도이고, 이 평면도에 나와

있는 점들의 Convex hull을 그린 것이 (b)이다. (c)는 세그먼트를 포괄하는 MOBB(Minimal Oriented Bounding Box)를 나타낸다. 이러한 평면도와 MOBB로부터 area, rectangularity, elongatedness라는 특성들의 값들을 경험적 임계값들과 비교하여 차량 여부를 판단한다. OBPCA는 Convex hull의 넓이만 사용할 뿐 형태적인 특성은 사용하지 않는다. 따라서 LiDAR 데이터만 사용하여 직사각형 형태인 MOBB의 특성값들을 계산하고 임계값과의 단순한 비교만 수행하기 때문에 빠르게 차량을 탐지할 수 있다는 장점을 가지고 있다. 그러나 형태적 특성을 단순화하기 때문에 차량과 크기가 비슷한 직육면체 물체와 차량을 구별할 수 없으며 나무와 같은 장애물에 가려진 차량도 탐지할 수 없다는 한계점이 있다.

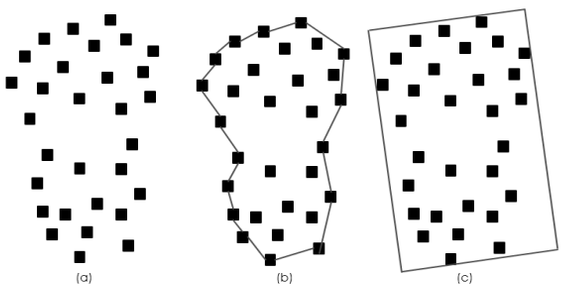


Fig. 3. Knowledge Based OBPCA

2.2 수직 단면을 이용한 물체 분류 기법

JointBoost[9]는 수직 단면, 정사영, 밝기(intensity)로부터 다양한 특성 값들을 계산하여 초목, 송전선, 송전탑, 건물, 지면 등의 물체들을 분류하는 알고리즘이다. 이러한 알고리즘에서 수직 단면을 통해 물체를 분석하는 부분이 Fig. 4에 나와 있다. 수직 단면에서는 점들의 높이 차이를 이용하여 물체의 특성을 파악하는데, 건물의 경우 커다란 높이 차이가 2번 나타난다는 사실을 통해 건물을 추출할 수 있다.

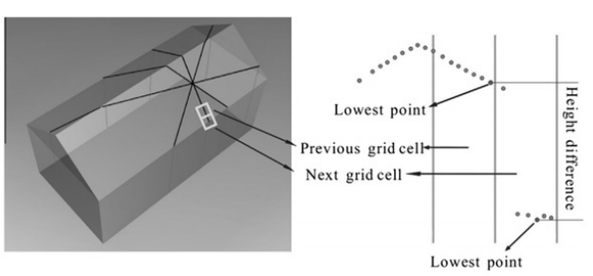


Fig. 4. Object Analysis Using Vertical Elevation[9]

JointBoost 알고리즘은 고품질의 분류 결과들을 생산하지만, 초기 분류 과정에서는 과분할과 과소분할로 인해 여전히 오류가 발생한다. 때문에 물체를 배경에 약하게 연결시키는 graph-cut 세그멘테이션 방법을 통해 JointBoost 분류 결과를 개선시켰다. 그러나 여전히 분류를 위해 사용되는 특성들이 물체를 정확히 표현한다는 보장이 없다.

3. OBPCA 기반의 수직 단면 이용 물체 분류 기법

3.1 필터링

필터링 과정에서는 전체 점 클라우드로부터 지면 점들을 제거함으로써 처리해야 하는 데이터 량을 줄인다. 필터링 기법들 중에서는 단순 계산을 통해 지형의 경사도를 알 수 있는 TIN(Triangulated Irregular Network) 기법을 사용한다. TIN은 적은 데이터 량으로 복잡한 지형을 나타낼 수 있고, 복잡한 지형은 더 많은 삼각형들을 통해 표현할 수 있다. LiDAR 점들로부터 가장 낮은 점을 찾아낸 뒤, 그 점으로부터 가장 가까운 점 두 개를 연결하여 삼각형을 형성한다. 이후 이 두 점에 각각 가까운 두 점들 두 개씩을 연결하여 또 다른 삼각형들을 형성하고, 이와 같은 과정을 반복하여 Fig. 5와 같은 TIN을 형성한다.

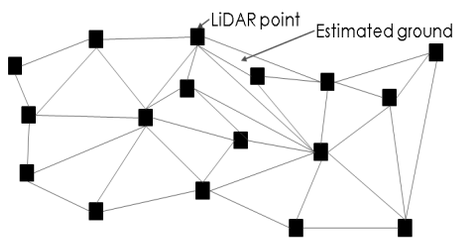


Fig. 5. Triangulated Irregular Network

3.2 세그멘테이션

필터링 후에는, 근접한 점들끼리 클러스터링하는 세그멘테이션 과정이 수행된다. 이 과정에서는 이웃을 찾는 기법들 중 하나인 FDN(Fixed Distance Neighbors)을 사용하여 서로 특정 거리 이하로 떨어져 있는 점들을 하나의 세그먼트로 그룹화한다. 이러한 과정의 의사코드가 Fig. 6에 나와 있다.

OBPCA 방법에 포함되어 있는 세그멘테이션 과정에서는 먼저 점 클라우드를 스택과 KD-tree에 insert 한다. KD-tree를 사용하는 것은 특정한 점의 빠른 검색 및 제거를 위함이고, 이 구조에서의 점 데이터들은 모두 unlabeled로 초기화한다. 그리하여 KD-tree에서 점 하나를 시드로 설정한 뒤, 시드 리스트를 생성하여 그 리스트에 시드 점을 추가한다. 이후 Check_PTR function에서는 KD-tree 내의 모든 점들을 확인하면서 시드 점과의 유클리드 거리가 임계값보다 짧게 나오는 것들은 이웃으로 간주하고 시드 리스트에 추가한다. 시드 점에 대해서 이웃들을 찾는 작업이 끝나면, Check_SEED function에서 시드 리스트의 다음 점을 시드로 설정하고 똑같이 이웃들을 찾는 작업을 반복한다. 그리하여 찾은 이웃들을 labeled로 지정하여 하나의 세그먼트로 정하고, 똑같은 방법으로 다른 세그먼트들을 Fig. 7과 같이 찾는다.

여기에서는 checked_all_seeds와 is_checked라는 두 가지 플래그들을 사용하였는데, checked_all_seeds는 시드 리스트 내의 모든 점들의 이웃을 찾았는지를 확인시켜 주고,

is_checked는 해당하는 점이 시드로서 설정된 적이 있는지를 확인시켜 준다. 이 두 플래그 덕분에 한 번 확인하였던 시드 점들을 다시 확인하는 중복 작업을 피할 수 있다.

Algorithm 1. Segmentation on C++ Program

INPUT : LiDAR data, OUTPUT : Candidates
PTR: Pointer in KD_Tree, KD_Tree: Entire point cloud

```
Function Check_PTR //Find neighbors of SEED
1     IF KD_Tree is not empty
2       IF PTR is neighbor with SEED
3             insert PTR into SEED_List
4             delete PTR from KD_Tree
5             set 1 to checked_all_seeds
6             Check_PTR(PTR, SEED)
7       ENDIF
8     ELSE
9       Check_PTR(PTR->left, SEED)
10      Check_PTR(PTR->right, SEED)
11    ENDIF
12  Function Check_SEED //Decide SEED
13    IF Seed_List is not empty
14      IF is_checked is 0
15            set List pointer to SEED
16            Check_PTR(PTR, SEED)
17            set 1 to is_checked
18      ENDIF
19      Check_SEED(SEED->next)
20    ENDIF
21  Function Main
22    Build KD_Tree(PTR)
23    Build Seed_List(SEED)
24     WHILE LiDAR_Data[]
25            insert point into KD_Tree
26     ENDWHILE
27     WHILE KD_Tree is not empty
28            set Tree Head pointer to PTR
29            insert PTR to Seed_List
30            set List Head pointer to SEED
31            delete PTR from KD_Tree
32            set 0 to checked_all_seeds
33            WHILE checked_all_seeds is 0
34                  set 1 to checked_all_seeds
35                  Check_SEED(SEED)
36            ENDWHILE
37     ENDWHILE
38
```

Fig. 6. Pseudocode of Proposed Segmentation Algorithm

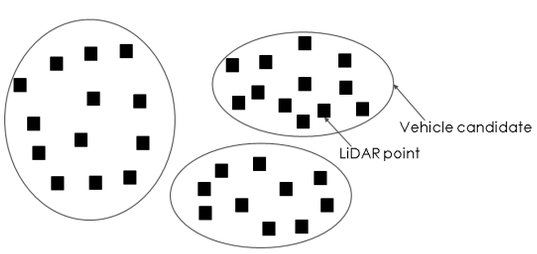


Fig. 7. Segmentation

3.3 추출

세그멘테이션이 끝난 후 만들어진 차량 후보 세그먼트들에 대하여 해당하는 세그먼트가 차량인지 아닌지를 최종적으로 판단하는 추출 과정이 진행된다. 이 과정에서는 차량 후보 세그먼트의 여러 2차원 기하학적 특성 값들을 측정한 뒤, 이 값들을 특성마다 다른 임계값들과 비교함으로써 각 세그먼트의 차량 여부를 결정한다. 기존의 OBPCA 방법에서는 차량 후보 세그먼트의 평면도에서만 2차원 기하학적 특성 값들을 측정하여 차량을 추출하지만, 제안하는 OBPCA에서는 수직 단면의 특성 값들도 추가적으로 측정하여 차량 추출에 활용한다. 우선은 평면도로부터 특성 값들을 측정하는 과정을 설명한 뒤에, 제안하는 방법에 해당하는 수직 단면 특성 값 측정 과정을 설명하도록 한다.

1) 평면도의 특성 추출

차량 후보의 특성 값들은 아래와 같다.

- Area: 세그먼트 convex hull의 면적
- Rectangularity: Area와 MOBB 면적의 비율
- Elongatedness: MOBB의 긴 변과 짧은 변의 길이 비율

LiDAR 데이터에서 나타난 차량들에 대한 지상 실측 정보를 통계적으로 분석한 결과, 위의 세 가지 특성 값들의 범위는 다음과 같은 임계값들을 이용하여 표현될 수 있다[10].

- $2 < \text{area} < 15$
- $0.6 < \text{rectangularity} < 1$
- $0.25 < \text{elongatedness} < 0.65$

Algorithm 2. OBPCA with top-view on C++ Program

INPUT : Candidates, OUTPUT: First extracted vehicles
PTR: Pointer in Candidates, LABEL: Vehicle number

```
1   Function Main
2     set List Head pointer to LABEL
3     WHILE Candidates[]
4     set List Head pointer to PTR
5       WHILE Segment[] //Find MOBB
6       set 0 to Xmax, Xmin, Ymax, Ymin
7             IF PTR label is same with LABEL
8                   update Xmax, Xmin, Ymax, Ymin
9                   insert point into Segment
10            ELSE
11                  break
12            ENDIF
13      ENDWHILE
14      Calculate lengths of edges of MOBB
15      Calculate area of MOBB
16      Calculate elongatedness
17      Build_Convex_Hull(each X, Y min, PTR)
18      Calculate area of Convex_Hull
19      Calculate rectangularity
20      IF area, elongatedness, and rectangularity are valid
21            Mark Segment as a vehicle
22      ENDIF
23      Set PTR to LABEL
24    ENDWHILE
```

Fig. 8. Algorithm Pseudocode for Calculating 2D Geometry Feature of Top-view

각 차량 후보 세그먼트의 평면도 특성 값들을 계산하는 의사코드가 Fig. 8에 나와 있다. Area를 구하기 위해서는 convex hull을 사용해야 하는데 이 convex hull을 찾는 과정은 N이 점의 개수일 때, O(*NlogN*)의 시간 복잡도를 가지는 Fig. 9의 Graham 스캔 알고리즘[10]을 활용한다. Graham 스캔 알고리즘은 이전에 찾아 놓은 Xmax, Ymin 값을 통해 가장 아래에 있으면서 가장 오른쪽에 있는 점을 시작점으로 사용한다. 이 점에 대한 나머지 점들의 각도를 구한 다음, 각도가 작은 순서대로 나머지 점들을 정렬시킨다. 시작점과 두 번째 점을 스택에 push한 후, 이 두 점들과 세 번째 점 간의 외적을 구한다. 이 외적이 양수인지 음수인지에 따라 세 점으로 만들어진 두 선분들이 볼록한 방향을 이루는지 오목한 방향을 이루는지를 파악할 수 있다. 만약 두 선분들이 볼록한 방향을 만들어내면 세 번째 점도 스택에 push하고, 그렇지 않으면 두 번째 점을 pop한 뒤에 세 번째 점을 push한다. 이렇게 세 기준점들을 각각 다음 점들로 바꿔가면서 같은 연산을 진행하면, 최종적으로는 바깥쪽 점들만을 이은 convex hull이 완성된다.

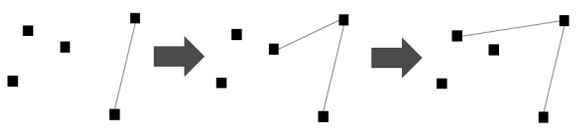


Fig. 9. Graham Scan Algorithm

세그먼트의 area는 이렇게 찾은 convex hull의 MOBB로 결정된다. convex hull에서의 가장 북쪽에 있는 점, 가장 남쪽에 있는 점, 가장 서쪽에 있는 점, 가장 동쪽에 있는 점들을 각각 찾아낸다. 이 점들이 MOBB의 꼭지점들이 되며, 유클리드 거리를 각각 계산하여 MOBB의 면적과 rectangularity, elongatedness를 계산하고 이를 임계값 범위 안에 있는지 여부를 평가하면 해당 세그먼트의 차량 여부를 확인할 수 있다. 이것이 평면도의 특성 값들을 통한 차량 추출 기법이다.

2) 수직 단면의 특성 추출

앞의 과정까지가 LiDAR 점들로부터 차량을 추출하는 기존의 OBPCA 방법이다. 본 논문에서는 Fig. 10에서 보는 것과 같이, 기존의 OBPCA가 선정한 차량 후보들에 대해 추가적인 특성 값들을 계산하여 더 정확하게 차량을 추출하는 방법을 제시한다. 이를 위해서는 차량 후보에서 가장 특징적인 수직 단면을 찾고, 그 단면의 특성 값들을 통해 각 후보의 차량 여부를 판단하도록 한다.

가장 특징적인 수직 단면을 찾기 위해서는 점 데이터 분포의 주성분을 찾는 PCA 기법을[11] 사용한다. 여기서 주성분이란 데이터들의 분산이 가장 큰 방향벡터를 말하며, Fig. 11에서 점들의 분산이 가장 큰 방향 벡터들은 e1과 e2라는 것을 알 수 있다. 이러한 주성분들의 방향 및 크기를 통해 차량 후보 점 분포의 특성을 간단히 표현할 수 있다.

PCA는 점 데이터들의 평균점을 지나는 직선들 중에서, 데이터들을 투사하였을 때 가장 분산이 큰 직선을 선정하여 주성분으로 활용한다. 본 논문에서는 3차원 점 데이터들을 투사하였을 때 가장 분산이 큰 수직 단면을 찾아내고 수직 단면에서의 특성을 평가하여 차량 추출에 활용한다. 평면도의 특성 값들을 이용하여 추출한 차량 후보들의 점의 개수가 M개일 때, 수직 단면 발견을 위한 PCA 주성분 분석의 시간 복잡도는 O(M)이다. 평면도에서와 마찬가지로 수직 단면에서도 아래와 같은 area, rectangularity, elongatedness라는 특성들을 활용한다. 본 논문에서는 시각화 툴을 통해 실험 데이터에 대한 차량 수직 단면의 특성 임계값들을 통계적으로 구하여 사용하였다. 보다 정확하며 신뢰성이 높은 차량 수직 단면의 특성 임계값을 구하기 위해서는 보다 큰 대량의 LiDAR 데이터로부터 수직 단면 특성의 임계값을 구하는 과정이 필요하다.

- $1 < \text{area} < 21$
- $0.3 < \text{rectangularity} < 0.9$
- $0.1 < \text{elongatedness} < 0.7$

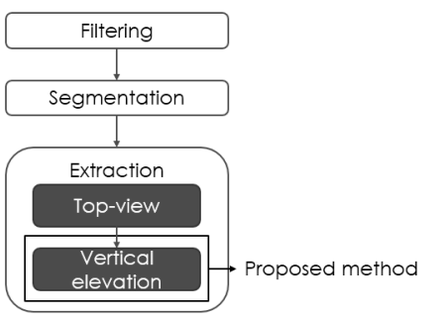


Fig. 10. Proposed OBPCA

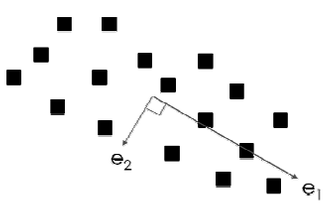


Fig. 11. Principal Component Analysis

이러한 수직 단면의 특성 값들을 계산하는 의사코드가 Fig. 12에 나와 있다. 수직 단면을 2차원 xy 평면으로 전환하여, 결과적으로는 가장 끝 점으로부터 각 점까지의 거리가 x 값으로, z값이 y값으로 전환된다. 이후 area, rectangularity, elongatedness를 계산하는 방법은 평면도에서와 동일하다.

이와 같이, 기존의 OBPCA로 찾은 차량 후보들에 대하여 수직 단면의 OBPCA를 적용함으로써 직육면체 형태의 비차량 물체들을 구별해 낼 수 있다. 본 논문에서는 수동 조사 결과와 항공 정밀 사진이 없는 상황을 고려하여 시각화 툴을 통해 수동적으로 조사를 하는 방법을 이용하여 지상 실측 정보를 생성하였다. LiDAR 시각화 툴로 MARS (Merrick Advanced Remote Sensing)를 사용하여 실제 차량들의 위

치를 확인할 수 있었다(Fig. 13). 이렇게 얻은 지상 실측 정보를 통해 LiDAR 점들로부터의 차량 추출 기법의 품질을 검증하고 비교 분석한다.

Algorithm 3. OBPCA with vertical elevation on C++ Program

```
INPUT : First extracted vehicles
OUTPUT: Final extracted vehicles
PTR: Pointer in Candidates, LABEL: Vehicle number
Function Main
    set List Head pointer to LABEL
    WHILE Candidates[]
      set List Head pointer to PTR
      WHILE Segment[] // Find PCA
            Find average point
            Find two points farthest from each other
            Calculate vector using those two points
            Find plane including three points
            Project points into the plane
            Transform the plane into xy plane
            Calculate height and length of the plane
            Calculate area and elongatedness
            Build_Convex_Hull
            Calculate area of Convex_Hull
            Calculate rectangularity
            IF the features are valid
                  Mark Segment as a vehicle
            ENDIF
            set PTR to LABEL
      ENDWHILE
      ENDFOR
  ENDWHILE
```

Fig. 12. Algorithm Pseudocode for Calculating 2D Geometry Feature of Vertical Elevation

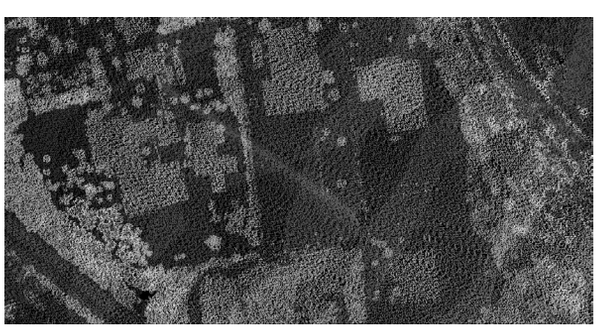


Fig. 13. Visualization of LiDAR Poing Cloud(MARS)

4. 성능 평가

4.1 성능평가 환경

제안 방법의 예측 성능을 측정하기 위해 기존의 OBPCA 방법과 함께 C++로 구현하였으며, OBPCA에 제안 방법을 적용하였을 경우와 적용하지 않았을 경우를 비교해 보았다. 예측 성능 평가에 사용된 실험 환경은 Table 1과 같다.

Table 1. Experimental Environment

CPU	i7-5930K
	3.5 Ghz
	Hexa-core
RAM	64GB
OS	Ubuntu 14.04 LTS
Programming Language	C++
Database	MongoDB

Table 2에서 보는 것과 같이, 원시 점 데이터는 5천여 만 개의 점들을 가진 1.8 GB 데이터이다. 이 데이터로부터 TIN 알고리즘을 통해 지면 점들을 분리하고, 이후 임계값을 통해 비차량 점들을 제거하면, 최종적인 차량 후보 점들은 천만 여개인 295MB만 남게 된다. 천만 여개의 점들만을 처리하여 차량들을 추출하면 되기 때문에 처리 효율이 훨씬 증가한다.

실험은 데이터 크기에 따라 실행 시간이 어떻게 변화하는지를 확인하기 위해 Table 2 데이터셋을 필터링과 세그멘테이션을 통해 탐색 공간을 대폭 줄인 세 가지 데이터셋에서 이루어졌으며 뒤에 비교 분석을 수행하였다(Table 3).

Table 2. Entire LiDAR Point Data Information

	Default	Ground	Vehicle Candidate
Data size	1.8GB	417MB	295MB
Number of points	55,223,871	11,873,089	10,791,115
Range	2km*2km	2km*2km	2km*2km

Table 3. Processed LiDAR Point Data Information

	Data1	Data2	Data3
Data size	240MB	660MB	900MB
Number of points	7,363,183	20,248,753	27,611,935
Range	2km*300m	2km*700m	2km*1km

4.2 성능평가 결과

기존 OBPCA와 보완된 OBPCA의 실행 시간을 비교한 Table 4에 나와 있으며, 단위는 초이다. Table 4에서 전체 실행 시간은 필터링 과정부터 세그멘테이션 후 추출 과정까지 걸린 시간의 총합이며, 이 중 차량 추출 과정에서 소요된 시간을 별도로 명시하였다.

Table 4에서 보듯이, 추출을 하는 과정 보다 주로 필터링과 세그멘테이션 과정에서 시간이 많이 소요되었다. 이것은 필터링이 전체 점 데이터를 모두 처리해야 하고, 세그멘테이션은 점들 간의 거리를 모두 계산해야 하는 시간적 비용을 가지고 있기 때문이다. 기존 OBPCA만 사용할 경우의 전체 실행 시간에 비해 수직 단면을 추가적으로 고려한 본 제안 방법은 전체 실행 시간이 13% 정도 더 오래 걸린 것을 확인할 수 있다. 수직 단면의 특성들을 구하는 과정에서는 점들의 분포 방향을 알아내고, 그 방향에 맞는 단면에 점들을 투사하는 작업으로 인하여 추가적인 시간이 발생한다.

Table 4. Comparison on Execution Times between Original OBPCA and Proposed Method

	OBPCA		Proposed method	
	Total execution time	Execution time for vehicle extraction	Total execution time	Execution time for vehicle extraction
Data1	2,433	465	2,748	780
Data2	6,851	1,438	7,713	2,300
Data3	9,382	2,001	10,583	3,202

기존 OBPCA에 제안 방법을 적용하였을 때와 적용하지 않았을 때의 분류 성능을 비교하기 위해 Recall, Precision, Quality, Accuracy, FP(False Positive) rate, TN(True Negative) rate, False discovery rate과 같은 평가 기준들을 사용하였다. 이를 위해 먼저 아래 Table 5와 같이 TP(True Positive), FP(False Positive), TN(True Negative), FN(False Negative)들의 수를 측정하였다. 각 metric들의 계산 식들은 표 아래에 나와 있다.

실험을 통해 얻은 분류 결과는 Table 6과 같다. 실험 데이터에서의 세그먼트의 총 개수는 413개였으며, TP, FP, TN, FN의 단위는 개수이다. Table 6은 기존 OBPCA에 제안 방법을 추가적으로 적용하였을 때의 분별력의 변화를 잘 나타내고 있다.

Table 5. Kinds of Measure for Calculating Accuracy Metrics

Total		Actual class	
		Positive	Negative
Predicted class	Positive	TP	FP
	Negative	FN	TN

Table 6. Comparison on Vehicle Extraction Accuracy between Original OBPCA and Proposed Method

Measures	OBPCA	Proposed method
Density of points (points/m2)	20	
TP	202	202
FP	75	52
TN	90	113
FN	46	46
Recall(%) = TP/(TP+FN)	81.45	81.45
Precision(%) = TP/(TP+FP)	72.92	79.53
Quality(%) = TP/(TP+FP+FN)	62.54	67.33
Accuracy(%) = TP+TN/(TP+FP+FN+TN)	70.7	76.27
FP rate(%) = FP/(TN+FP)	45.45	31.52
TN rate(%) = TN/(TN+FP)	54.55	68.48
False discovery rate(%) = FP/(TP+FP)	27.08	20.47

본 제안 방법은 기존 OBPCA를 수행한 후 추가적인 수직 단면에 대한 평가를 수행함으로써 기존 OBPCA에서 차량으로 인식한 것들 중 틀린 것들 (FP)를 줄여준다. 따라서 기존 OBPCA에서 이미 차량이 아닌 것으로 판별한 세그먼트는 수직 단면 평가 대상에서 제외되므로 Recall 값은 기존 OBPCA와 변동이 없다. 그러나 기존의 OBPCA가 차량이라고 찾아낸 것들 중에서 실제로 차량이 아닌 것들을 골라냄으로써 FP를 감소시키고 TN을 증가시켰다. 그리하여 Recall은 그대로이지만, Precision과 Quality, Accuracy가 각각 6.61%, 4.79%, 5.57% 향상되고 False positive rate와 false discovery rate는 각각 13.93%와 6.61% 감소된 것을 확인할 수 있었다. 이것은 차량과 크기가 비슷한 직육면체 형태의 물체로부터 차량을 구별하지 못하는 기존 OBPCA의 단점을 해결하고자 하는 본래의 목적에 부합하며, 차량이 아닌 것을 차량이라고 판단하는 잘못된 인식의 비율을 줄이는 방향으로의 정확도 향상에 기여하였다고 볼 수 있다.

5. 결 론

점 클라우드로부터 차량을 추출하는 다양한 방법들 중에서 OBPCA 방식은 비교적 우수한 정확도와 빠른 추출 성능을 보여 여러 응용 프로그램에서 사용되어 왔다. 하지만, 수평단면만에서 추출한 특성만을 사용하여 추출을 진행하므로 비슷한 크기의 직사각형 물체에 대해 분류오류가 많이 발생하는 단점이 있었다. 본 논문에서는 기존 OBPCA 절차의 수행이후에 추가적으로 수직 단면을 이용한 평가를 수행함으로써 기존 OBPCA에서 차량으로 인식된 세그먼트에 대한 오류를 줄이는 새로운 방식을 제안했다.

향후 연구로 본 제안 방법의 추가적인 수직 단면 평가 작업으로 인한 수행시간 증가 문제를 보완하기 위해 차량 추출 단계에서 세그먼트 단위의 병렬 처리를 통해 수행 시간을 가속화할 계획이다. 또한 다양한 종류의 관련 데이터들을 활용하고 차량 추출 과정을 병렬화한다면 차량 추출의 분류 정확도와 처리 성능을 동시에 향상시킬 수 있을 것으로 기대한다.

References

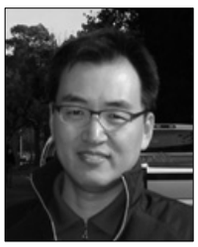
[1] B. Yang, P. Sharma, and R. Nevatia, “Vehicle detection from low quality aerial LIDAR data,” *IEEE Workshop on Applications of Computer Vision (WACV)*, pp.541-548, 2011.

[2] F. Rottensteiner, “Advanced Methods for Automated Object Extraction from LiDAR in Urban Areas,” *Geoscience and Remote Sensing Symposium (IGARSS), IEEE International*, pp.5402-5405, 2012.

[3] X. Feng, “Artificial neural networks forecasting of PM2.5 pollution using air mass trajectory based geographic model and wavelet transformation,” *Atmospheric Environment*, Vol.107, pp.118-128, 2015.

[4] J. Han, M. Kamber, and J. Pei, “Data mining concepts and techniques,” *Morgan Kaufmann Pub.*, pp.1-5, 2012.
[5] J. Zhang, M. Duan, Q. Yan, and X. Lin, “Automatic vehicle extraction from airborne LiDAR data using an object-based point cloud analysis method,” *Remote Sensing*, Vol.6, No.9, pp.8405-8423, 2014.
[6] W. Yao, “Comparison of Two Methods for Vehicle Extraction From Airborne LiDAR Data Toward Motion Analysis,” *IEEE Geoscience and Remote Sensing Society*, Vol.8, Issue 4, pp. 607-611, 2011.
[7] J. Secord and A. Zakhor, “Tree Detection in Urban Regions Using Aerial Lidar and Image Data,” *Geoscience and Remote Sensing Letters, IEEE*, Vol.4, Issue 2, pp.196-200, 2007.
[8] W. Yao, S. Hinz, and U. Stilla, “3D object-based classification for vehicle extraction from airborne LiDAR data by combining point shape information with spatial edge,” *6th IAPR Workshop Pattern Recognition in Remote Ssensing*, pp.1 - 4, 2010.
[9] B. Guo, X. Huang, F. Zhang, and G. Sohn, “Classification of airborne laser scanning data using JointBoost,” *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol.100, pp.71-83, 2015.
[10] M. Atwah, and J. Baker, “An Associative Implementation Of Graham's Convex Hull Algorithm,” *Parallel and Distributed Computing*, 1995.
[11] I. Jolliffe, “Principal Component Analysis,” in *Wiley StatsRef: Statistics Reference Online*, John Wiley & Sons, Ltd, 2002.

전 준 범
e-mail : stema@ajou.ac.kr
2014년 아주대학교 정보컴퓨터공학부 (학사)
2016년 아주대학교 소프트웨어특성화학과 (석사)
2016년～현 재 TmaxOS Office실 선임연구원
관심분야 : Large scale data processing, Computer vision, Machine learning



이 희 진
e-mail : heezin.lee@berkeley.edu
2008년 미국 플로리다대학교 전기컴퓨터공학 (박사)
2008년～2011년 미국 플로리다대학교 박사후연구원
2011년～현 재 미국 캘리포니아대학교 책임연구원
관심분야 : 신호처리, Lidar Remote Sensing, 영상처리, 패턴인식

오 상 윤
e-mail : syoh@ajou.ac.kr
2006년 미국 인디애나대학교 컴퓨터공학과 (박사)
2006년～2007년 SK텔레콤
2007년～현 재 아주대학교 소프트웨어학과 부교수
관심분야 : 분산/병렬 시스템, 고성능컴퓨팅, Large Scale Software System, Semantic Web

이 민 수
e-mail : michelle.lee@ewha.ac.kr
2001년 이화여자대학교 수학과(학사)
2003년 이화여자대학교 컴퓨터공학과(석사)
2007년 이화여자대학교 컴퓨터공학과(박사)
2007년～2011년 서울대학교 컴퓨터공학과 박사후연구원
2013년～2014년 미국 인디애나대학교 정보컴퓨팅학부 방문연구원
2014년～2015년 이화여자대학교 컴퓨터공학부 박사후연구원
2015년～현 재 이화여자대학교 컴퓨터공학과 연구교수
관심분야 : Active learning, Adaptive stream data mining, Bioinformatics, Machine learning