멀티 채널 라이다 오브젝트 탐지 시스템 개발

(Deveopment of multi-channel LiDAR object detection system)

김 용 훈[†*], 송 진 선[‡]
[†]경북IT융합산업기술원
(Yong-Hoon Kim, Jin-Seon Songk)
([†]Gyeongbuk Institute of IT Convergence Industry Technology.)

Abstract : LiDAR sensors are an essential sensor in autonomous vehicles, and as the price of LiDAR sensors is falling, technologies using LiDAR sensors are being actively developed. In this paper, we describe an system for identifying and distinguishing objects in the environment around autonomous cars using multi-layer LiDAR sensors.

Keywords : Autonomous vehicle, LiDAR, Velodyne LiDAR, Multi-Layer LiDAR, Object

Ⅰ. 서론

오늘날 자동차들은 여러 ADAS 시스템들이 개발되면서 지능화, 고도화 되어 운전자에게 보다 안전하고 편리한 편의성을 제공해주고 있다.[1] 최근에는 자율주행이라는 혁신적인 기술개발로 인하여 고속도로와 같은 제한적인 공간에서 부분적으로 자율주행이 상용화 되어 가고 있는 추세이다. 자율주행 자동차에서 라이다 센서는 영상 센서와 함께 필수 센서 중 하나로 구분되어 있고 최근 라이다 센서의 가격의 인하로 다양한 어플리케이션의 활용이 가능해진 상황이다.[2]

본 논문에서는 자율주행 자동차의 필수 센서인 라이다 센서를 활용하여 주변 오브젝트를 탐지(Detection)하는 알고리즘을 설명한다.

Ⅱ. 본 론

1. 센서부 구성

라이다 센서를 활용한 오브젝트 탐지 알고리즘 개발을 위해 사용된 센서는 Velodyne 사의 VE16E 센서를 사용하였다. 본 실험에서는 strongest로 설정하여 600 RPM으로 센서를 동작 시켰다.

그림 1. Velodyne 16E
Fig. 1. Velodyne 16E

2. 처리부 구성

VLP-16 라이다 센서는 한 패킷 당 24개의 angle 정보를 가진다. 0.2도를 기준으로 했을 때 1800개의 horizontal angle 수가 나온다. 여기서 각 horizontal angle 당 16개의 vertical angle을 가진다. 한번 스캐닝을 할 때마다 28800개의 거리정보 (2 byte) 와 intensity (1 byte)를 가진다. 이를 바이트로 환산했을 때 86400 byte이다. 최대 20Hz의 스캐닝 속도를 가질 경우는 고속의 패킷 처리 기술이 필요하게 된다.

초당 8Mbps의 데이터량을 가지기 때문에 센싱 포인트 정보를 point 단위가 아닌 오브젝트 단위로 관리하여 처리량을 줄여주고 이로 인해 처리 속도 향상을 도모하였다. 그림 2는 VLP-16 데이터 시트에 나오는 패킷 파싱 알고리즘이다. 그림 2와 같은 알고리즘으로 UDP 패킷을 파싱하여 한 프레임에 대한 distance 정보와

* 교신저자(Corresponding Author)
김용훈 : (재)경북IT융합산업기술원
※ 본 연구는 산업통산자원부와 한국산업기술진흥원이 지원하는 광역협력권산업육성사업으로 수행된 연구결과입니다.(과제번호 : P0006199/P0008604)

intensity 정보를 정리하여 데이터 구조로 정리해야 한다.

```
//  First, adjust for a rollover from
359.99° to 0°
If (Azimuth[3] < Azimuth[1])
   Then Azimuth[3]:= Azimuth[3]+360;
Endif;
// Perform the interpolation
Azimuth[2]:=Azimuth[1]+((Azimuth[3]-Azimuth[1])/2);
// Correct for any rollover over from 359.99° to 0°
If (Azimuth[2]>360)
   Then Azimuth[2]:= Azimuth[2]-360;
Endif
```

그림 2. 패킷 파싱 알고리즘
Fig.2. Packet parsing algorithm

그림 3은 포인트 정보에서 오브젝트 정보로 변환하기 위한 오브젝트의 정보 구조체이다.

```
typedef struct {
        int VobjID;
        int polyObjID;
        int MaxUpperIndex;
        int MaxDownIndex;
        int ObjectPointCount;
        int pointNANCount;
        float posX[16];
        float posY[16];
        float posZ[16];
        float posI[16];
        float pointNAN[16];
} V_OBJList;
```

그림 3. 오브젝트 구조체
Fig.3. Object Structure

그림 4은 point 데이터의 전체적인 흐름이다.

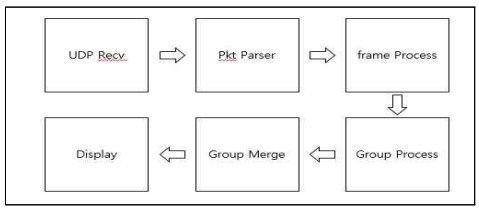

그림 4. 오브젝트 탐지 절차
Fig.4. Sequence of object detection

먼저 패킷을 seamless하게 처리할 수 있어야 한다. recv task와 parser task 사이에는 링버퍼를 두어 packet loss를 줄여 준다. parser에서 해체된 정보들은 frame task에서 하나의 프레임으로 완성시켜준다. 완성된 프레임에서 각 포인트의 Grouping을 수행시킨다. 이때 지면 정보는 제거해 준다. 지면이 제거된 point cloud에서 인접한 포인트끼리 그룹을 만들어 준다. 이후 생성된 그룹 간 거리 값을 기준으로 다시 그룹화를 해준다. 두단계로 나눠진 그룹화이지만 두 번째의 경우 이미 필터링 된 데이터를 그룹화하기 때문에 적은 리소스르 사용한다.

Ⅲ. 결 론

그림 5는 앞서 설명한 절차에 따라 오브젝트를 탐지하는 프로그램의 결과 화면이다. 정중앙 오른쪽 점들은 차량의 지붕이 검출된 것으로 처리 영역에서 제외시켰다. 중앙의 흰 선은 도로 노면으로 추정되는 영역을 흰색으로 표시하여 검출에 문제가 없는지 확인하였다. 그 외 파란색 선은 도로 노면은 아니지만 바닥으로 추정되는 영역이다. 장애물로 인식하는 개체들은 그림과 같이 폴리곤 박스를 처서 위치와 크기를 확인할 수 있다. 현재 실시간 동작 시 디버그 모드에서 10fps의 처리 속도를 보여 주고 있다. 현재 연구에서는 0°과 359° 사이 단락이 생기고 차량 속도가 증가할 경우 유격이 발생한다. 전방 장애물의 경우 두 개의 오브젝트로 오인식되는 경우가 있어 이를 수정 보완해야한다.

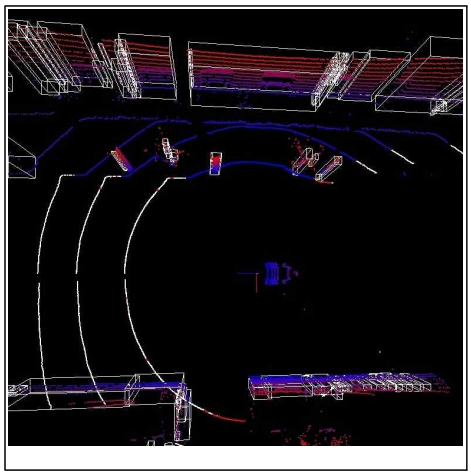

그림 5. 오브젝트 탐지 결과
Fig. 5. Object detection result

참 고 문 헌

[1] 이한승, 김동욱, 이경수 "자율 주차 시스템의 직각 주차를 위한 조향 제어 알고리즘", 한국자동차공학회 학술대회, 1280-1288쪽, 2011.

[2] Q. Zhu et al., "3D LIDAR point cloud based intersection recognition for autonomous driving", Proc. IV, 2012.