33D LIDAR 를 기반으로 하는 저비용 자율 주행 시뮬레이터 프레임워크에 대한 연구

오은택, 조민우, 구본우*
경민대학교 게임콘텐츠과

griogrio@kyungmin.ac.kr

Study about Low-Cost Autonomous Driving Simulator Framework Based on 3D LIDAR

Eun Taek O, Min Woo Cho, Bon Woo Gu
Dept. of Game Contents, Kyung-Min University

요 약

자율주행 시뮬레이터를 위한 대체재로 게임 엔진을 통한 가상 환경 모의 연구가 수행되고 있다. 하지만 게임 엔진에서는 자율 주행에 필요한 센서를 기기에 맞게 사용자가 직접 모델링을 해줘야 하기 때문에 개발 비용이 크게 작용된다. 특히, Ray 를 활용한 3D LIDAR 는 GPU(Graphics Processing Unit) 사용량이 많은 작업이기 때문에 저비용 시뮬레이터를 위해서는 저비용 3D LIDAR 모의가 필요하다. 본 논문에서는 낮은 컴퓨터 연산을 사용하는 C++ 기반 3D LIDAR 모의 프레임 워크를 제안한다. 제안된 3D LIDAR 는 다수의 언덕으로 이루어진 비포장 Map 에서 성능을 검증 하였으며, 성능 검증을 의해 본 논문에서 생성된 3D LIDAR 로 간단한 LPP(Local Path Planning) 생성 방법도 소개한다. 제안된 3D LIDAR 프레임 워크는 저비용 실시간 모의가 필요한 자율 주행 분야에 적극 활용되길 바란다.

1. 개요

높은 비용의 시뮬레이터를 대체하기 위하여 게임 엔진을 통한 가상 환경 모의 연구가 수행되고 있다[1]. 하지만 게임 엔진에서는 자율 주행에 필요한 센서를 기기에 맞게 사용자가 모델링을 해줘야 하기 때문에 개발 비용이 크게 작용한다. 특히 Ray 를 활용한 3D LIDAR 는 GPU(Graphics Processing Unit) 사용량이 많은 작업이기 때문에, 저비용 시뮬레이터를 위해서는 저비용 3D LIDAR 모의가 필요하다.

본 논문에서는 낮은 컴퓨터 연산을 사용하는 C++ 기반 3D LIDAR 모의 프레임 워크를 제안한다.

2. 관련 연구

황종락 외는 3D LIDAR 데이터 처리를 통해 생성된 도로 특징 지도를 구축하고 Particle Filter 기반 실시간 위치인식기법을 개발하였다[2].

박재웅 외는 글로벌 맵(global map)과 LIDAR 를 통해 얻은 로컬 맵(local map)을 매칭하여 자율 주행하는 알고리즘을 개발하였다[3]. 송현우 외는 LIDAR 센서를 구현해 장애물과의 거리와 방향을 측정하고 목적지의 좌표와 해당 센서의 위치를 비교해 장애물을 퍼지 제어 기반 알고리즘을 이용해 구별하고 회피하는 방법을 제안하였다[4].

조든솔 외는 3D 자동차 시뮬레이터 기반 상호작용형 ADAS 개발 및 검증 프레임워크를 개발했다. ADAS 는 시뮬레이터로부터 필요로 하는 데이터를 직접적으로 제공하지 않아 검증에 한계가 있어 이를 보완하고자 하였다[5].

본 논문에서는 높이가 불규칙한 비포장도로에 제안한 3D LIDAR 모의 프레임 워크를 적용하여 실시간 자율 주행을 실험했다. 3D LIDAR 는 LPP 를 생성하며, GPP 에 최대한 가까운 주행 방향을 생성한다. 본 논문에서 제안된 3D LIDAR 프레임 워크와 시뮬레이터가 저비용 실시간 모의가 필요한 자율 주행 분야에 적극 활용되길 바란다.

3. 3D LIDAR 를 이용한 방향 결정 모듈

본 논문에서 제안된 3D LIDAR 모의는 OUSTER 사의 OS1 을 모델로 모의하며[6] DirectX11 기반 동적 라이브러리(Dynamic Link Library)로 개발한다

3.1 3D LIDAR 를 이용한 방향 결정 모듈

Table. 2.은 3D LIDAR 의 포인트 클라우드를 통해 구한 LPP 와 3D LIDAR 의 방향을 통해 얻은 GPP 를 통해서 차량의 방향을 결정하는 수도 코드를 보여준다. 본 수도 코드는 3D LIDAR 를 통해 얻은 각

Direction 포인트 클라우드의 높이 값을 평균 내어 평균값이 낮을수록 높은 LPP 확률 값을 Direction 에 부여하였다. Direction 의 방향 벡터와 차량부터 목표까지의 방향 벡터의 길이를 비교하여 길이가 짧을수록 높은 GPP 확률 값을 부여하였다. 차량의 최종적인 방향은 LPP 에 가중치 W_1 을 곱한 값과 GPP 에 가중치 W_2 을 곱한 값을 합하여 가장 값이 높은 Direction 으로 결정한다. 본 논문에서는 W_1 와 W_2 을 각각 0.4, 0.6 의 수치를 부여하여 LPP 와 GPP 의 비율을 조절하였다.

수도 코드 1:
차량 방향 결정

INPUT: *PointSum, PointCount, DirCount, Angle, GoalDir, CurDir*
OUTPUT: *CarDir*

```
for i ← 0 to DirCount do :
  if PointCount[i] = 0 then
    PointAvg[i] ← ∞
  else
    PointAvg[i] ← PointSum[i] / PointCount[i]
  end if

  CurDir ← [Sin(Angle[i]), Cos(Angle[i])]
  DirLen[i] ← Length(GoalDir, CurDir[i])
end for

Pmin ← min(PointAvg)
Pmax ← max(PointAvg)
Plen ← Pmax - Pmin

Dmin ← min(DirLen)
Dmax ← max(DirLen)
Dlen ← Dmax - Dmin

for I ← 0 to DirCount do:
  LPPValue[i] ← 1.0 - ((PointAvg[i] - Pmin) / Plen)
  GPPValue[i] ← 1.0 - ((DirLen[i] - Dmin) / Dlen)
  DirRate[i] ← (PointValue[i] · W1) + (DirValue[i] · W2)
end for

CarDir ← argmax(DirRate)
return CarDir
```

<표 1> Determining the direction of the Vehicle

Pmin 은 *Point* 높이의 최솟값을 의미한다. *Pmax* 는 Point 높이의 최댓값을 의미한다. *Plen* 은 *Pmax* 와 *Pmin* 의 길이를 의미한다. *DirCount* 는 3D LIDAR Direction 의 총 개수를 의미한다. 본 논문에서 Direction 은 차량 전방의 9 방향을 사용하였다. 각 Direction 은 각 10° 의 조향각을 가진다. *PointCount* 는 Direction 에 존재하는 Point 의 개수를 의미한다. *PointAvg* 는 Direction 에 존재하는 Point 의 높이 평균 값을 의미한다. *PointAvg* 가 높을수록 주행이 불가능한 벽을 의미한다. Angle 은 Direction 의 라디안 각도를 의미한다. *Dmin* 은 *DirLen* 의 최솟값을 의미한다. *Dmax* 는 *DirLen* 의 최댓값을 의미한다. *Dlen* 은 *Dmin* 과 *Dmax* 의 길이를 의미한다. *PointSum* 은 Direction 에 존재하는 Point 의 높이 총합 값을 의미한다. *CurDir* 은 Direction y 축 각도의 2 차원의 단위 벡터를 나타낸다. *DirLen* 은 *CurDir* 와 *GoalDir* 의 길이로 두 단위 벡터 사이의 유사도를 의미한다. *GoalDir* 은 차량과 GPP 목표 사이의 2 차원 단위 방향 벡터를 의미한다. *LPPValue* 는 각 Direction 의 LPP 확률을 의미한다. *GPPValue* 는 각 Direction 의 GPP 확률을 의미한다. *DirRate* 는 LPP 와 GPP 확률 수치의 합을 의미한다. 본 프레임워크에서는 *DirRate* 가 높을수록 차량이 안정적으로 이동할 수 있는 길이라고 판단한다. *argmax* 는 *DirRate* 중 가장 높은 확률의 Index 를 반환한다. 본 수도코드에서 Index 는 차량이 회전할 수 있는 9 방향을 의미한다. *CarDir* 은 차량의 최종 방향을 의미한다.

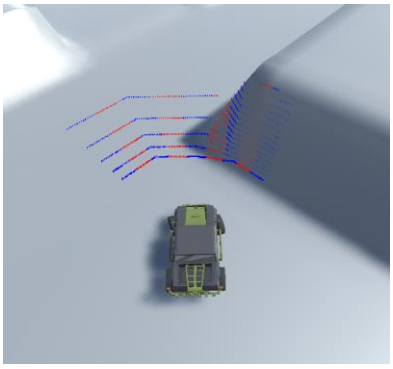
4. 실험 결과

이 섹션에서는 제안된 3D LIDAR 모의 결과와 주행 방향 결정에 대한 실험을 진행한다. 4.1. 에서는 Unity 3D Engine 에 구현된 결과를 설명한다. 4.2. 에서는 30 개의 크기와 높이가 서로 다른 불규칙한 미로 Map 에서 수행한 주행 결과를 설명한다.

실험에서 사용된 프로그램은 Unity 3D Engine version 2019.4.34f1 을 사용하였으며, 사용 언어로는 DirectX11, C/C++, C# Script 를 사용하였다. 실험 환경은 Intel-i5 CPU, 8GB RAM 을 사용하였다.

4.1 실험 결과 화면

Unity Engine 의 WheelCollider 라이브러리를 사용하여 4 륜구동 차량을 모의하였고, 21x21, 31x31, 41x41 사이즈의 미로 Map 을 각각 10 가지 씩 총 30 가지의 지형을 바탕으로 실험을 진행하였다.



(그림 1) Hill Drive

그림 1 은 주행 방향 결정에 대한 결과를 보여준다. 차량은 획득한 Point Cloud 의 높이 평균 값을 이용하여 언덕을 구분한다. 낮은 수준의 언덕을 마주할 경우, 주위 벽들에 비해 높이가 상대적으로 낮은 언덕을 벽으로 인식하지 않고 주행을 계속하는 모습을 보여주며, 3D LIDAR 해상도의 높낮이에 따라 다른 경로를 탐색하게 되는 상황이 있다.

4.2 주행 결과

표 2 는 실제 시뮬레이터 실행 후 시작 지점부터 도착 지점까지 진행된 시간을 기록한 표이다. 1 열은 맵 해상도를 구분한다. 1 행은 1 회차 내에 변경되어 적용한 3D LIDAR 의 해상도를 구분한다.

3D LIDAR 해상도 / 맵 해상도	512*16	1024*16	512*64	1024*64
21 * 21	29.793	26.5837	24.494	23.392
31 * 31	44.157	41.187	39.367	35.46
41 * 41	58.845	59.707	54.681	51.808

<표 2> Determining the direction of the Vehicle

각 맵의 해상도와 3D LIDAR 의 해상도 별로 5 번씩 실행하였으며 각 수행마다 도착 지점과 출발 지점을 무작위로 선정하였다.

그 후 3D LIDAR 해상도 별로 평균을 적용하였다. 대다수의 기록은 3D LIDAR 의 해상도가 높아질수록 평균 진행 시간이 줄어드는 결과를 보여주지만 간혹 41*41 맵 해상도의 1024 * 16 과 같이 더 높은 3D LIDAR 해상도에서 더 높은 평균 진행 시간이 기록되는 경우도 존재했다. 이는 원칙 상 진행할 수 없는 구간이지만 해상도가 낮은 탓에 제대로 인식하지 못하고 진행할 수 있다고 인식하고 직진한 경우인데 지속적인 직진에 의해 억지로 그 구간을 넘어가면서 기록이 단축된 케이스이다. 또한 맵의 해상도가 커질수록 연속적으로 이어진 언덕이 다수 발생하여 모든 방향의 LPP 확률이 낮아지는 현상도 발생하였다.

본 실험을 통하여, 본 논문에서 제안한 3D LIDAR 모의 프레임 워크는 GPU 하드웨어의 제한 없이 저비용으로 자율 주행 방향 선택을 시뮬레이션 할 수 있음을 확인하였다.

5. 결론

본 논문에서는 저비용 3D LIDAR 모의 시뮬레이터를 제안하였으며, 3D LIDAR 로부터 획득한 포인트 클라우드 데이터를 이용하여 목표까지 이동하는 방법을 실험했다. 본 실험에서는 무작위로 생성된 미로 환경에서 3D LIDAR 를 이용하여 획득한 클라우드 포인트의 높이 값으로 지형의 주행 가능 확률을 계산하였으며, 계산된 주행 가능 확률을 기반으로 차량의 주행 방향을 선택하였다. 제안한 방법은 Unity 3D 엔진을 이용하여 제작한 환경에서 검증을 진행하였다. 하지만 시뮬레이션 환경이 실제 환경에 비해 상대적으로 단순하여 나무 등과 같은 장애물 같은 요소는 포함되지 않았고 목표 지점까지 이동하는 것에는 성공하였으나 효율적인 주행이라고는 할 수 없다. 향후 연구에서는 시뮬레이션 환경에 나무, 동물 등과 같은 정적, 동적 장애물을 추가하고 이를 3D 라이더를 통해 장애물을 판단, 방향 선택을 진행하여 보다 현실적인 시뮬레이터를 연구할 예정이다. 본 논문은 센서를 이용한 자율주행 연구 분야에 기여하길 바란다.

참고문헌

[1] 정홍렬; 박영제; 문형필. 언리얼 엔진을 이용한 다개체 주행 로봇용 시뮬레이터 개발. 대한기계학회 춘추학술대회, 제주국제컨벤션센터(ICC 제주), 2019, 1256-1259.

[2] 황종락; 안경재; 강연식. 3D 라이다 센서와 도로 환경 지도의 비교를 통한 자율주행 자동차 위치인식 기법 검증. 제어로봇시스템학회 논문지, 25.6: 557-564, 2019.

[3] 박재웅; 김재환; 김정하. 차량 모델 및 LIDAR 를 이용한 맵 매칭 기반의 야지환경에 강인한 무인 자율주행 기술 연구. 제어로봇시스템학회 논문지, 17.5: 451-459, 2011.

[4] 송현우; 이광국; 김동헌. 무인선박의 자율운항을 위한 저가형 LiDAR 센서 기반의 장애물 회피 시스템 구현. 전기학회논문지, 68.3: 480-488, 2019.

[5] CHO, Deun-Sol, et al. Interactive ADAS development and verification framework based on 3D car simulator. Journal of IKEEE, 22.4: 970-977, 2018.

[6] 권순웅; 강동완; 김정하. LiDAR 기반 V2I 를 이용한 자율 주행 자동차의 객체 검출 알고리즘 개선 연구. 한국자동차공학회 추계학술대회 및 전시회, 제주 신화월드, 2020, 674-678.