KIPS Trans. Comp. and Comm. Sys.
Vol.12, No.3 pp.93~102
ISSN: 2287-5891 (Print), ISSN 2734-049X (Online)

https://doi.org/10.3745/KTCCS.2023.12.3.93

Performance Evaluation Using Neural Network Learning of Indoor Autonomous Vehicle Based on LiDAR

Yonghun Kwon† · Inbum Jung††

ABSTRACT

Data processing through the cloud causes many problems, such as latency and increased communication costs in the communication process. Therefore, many researchers study edge computing in the IoT, and autonomous driving is a representative application. In indoor self-driving, unlike outdoor, GPS and traffic information cannot be used, so the surrounding environment must be recognized using sensors. An efficient autonomous driving system is required because it is a mobile environment with resource constraints. This paper proposes a machine-learning method using neural networks for autonomous driving in an indoor environment. The neural network model predicts the most appropriate driving command for the current location based on the distance data measured by the LiDAR sensor. We designed six learning models to evaluate according to the number of input data of the proposed neural networks. In addition, we made an autonomous vehicle based on Raspberry Pi for driving and learning and an indoor driving track produced for collecting data and evaluation. Finally, we compared six neural network models in terms of accuracy, response time, and battery consumption, and the effect of the number of input data on performance was confirmed.

Keywords : Indoor Autonomous Driving, Neural Network, LiDAR, Raspberry Pi

라이다 기반 실내 자율주행 차량에서 신경망 학습을 사용한 성능평가

권 용 훈† · 정 인 범††

요 약

클라우드를 통한 데이터 처리는 통신 과정에서 지연시간과 통신비용 증가 등 같은 많은 문제가 발생한다. 사물인터넷 분야에서는 이러한 문제를 해결하기 위해 엣지 컴퓨팅 연구가 활발히 이루어지고 있고, 대표적인 응용 분야로 자율주행이 있다. 실내 자율주행에서는 실외와 달리 GPS와 교통정보를 이용할 수 없기 때문에 센서를 활용하여 주변 환경을 인식해야 한다. 그리고 자원이 제약된 모바일 환경이기 때문에 효율적인 자율주행 시스템이 필요하다. 본 논문에서는 실내 환경에서 자율주행을 위해 신경망을 사용하는 기계학습을 제안한다. 신경망 모델은 LiDAR 센서에서 측정된 거리 데이터를 바탕으로 현재 위치에 가장 적절한 주행 명령을 예측한다. 신경망의 입력 데이터의 수에 따른 성능 평가를 수행하기 위해 6가지의 학습 모델을 설계하였다. 주행과 학습을 위해 Raspberry Pi 기반의 자율주행 차량을 제작하였고, 학습 데이터 수집과 성능평가를 위한 실내 주행 트랙을 제작하였다. 6가지의 신경망 모델들은 정확도와 응답시간 그리고 배터리 소모에 대한 성능 비교를 하였고, 입력 데이터의 수가 성능에 미치는 영향을 확인하였다.

키워드 : 실내 자율주행, 신경망, 라이다, 라즈베리 파이

1. 서 론

자율주행은 사물인터넷이 발전하면서 활발히 연구되고 있는 분야로 농업과 같은 1차 산업에서부터 운송, 공공 복지서비스까지 산업 전반에 걸쳐 다양한 목적을 가지고 연구되고 있다[1]. 시간이 지남에 따라 자율주행 연구는 대역폭의 문제로 인해 지연시간이 발생하는 클라우드보단 데이터를 생성된 곳 근처에서 처리하여 지연시간 문제를 해결할 수 있는 엣지 컴퓨팅으로 확대되고 있다[2]. 엣지 컴퓨팅은 차량에서 엣지 서버로 오프로딩(offloading)하여 연산하거나, 차량이 모바일 엣지 기능을 수행하는 방식 등 연결성을 중점으로 발달했다. 모바일 엣지는 사용자가 운전하는 자동차를 비롯한 운송수단과 원격으로 조종하는 드론 등 다양한 플랫폼으로 구현되어 서비스를 제공할 수 있다[3].

† 준 회 원 : 강원대학교 컴퓨터정보통신공학과 박사과정
†† 종신회원 : 강원대학교 컴퓨터정보통신공학과 교수
Manuscript Received : September 20, 2022
First Revision : November 2, 2022
Accepted : November 17, 2022
* Corresponding Author : Inbum Jung(ibjung@kangwon.ac.kr)

실내 자율주행은 실외와 다르게 GPS (Global Positioning System) 이용한 위치인식과 주변 환경 파악이 어렵다는 차이점이 존재한다. 그러므로 PLAN(Positioning, Localization and Navigation) 기술을 바탕으로 하여 요구사항과 센서에 따라 다양한 기법들이 연구되고 있으며, 자율주행에서 사용되는 센서는 카메라, 라이다(LiDAR, Light Detection And Ranging), 관성센서, 초음파 등이 있다. 실내 자율주행에서는 다양한 센서를 이용하여 내부의 구조를 파악하고 지도 작성과 위치 인식을 동시에 수행하는 SLAM(Simultaneous Localization and Mapping)과 주행 경로를 계산하는 알고리즘 그리고 객체 인식과 같은 기계학습 알고리즘으로 차량을 제어하는 연구 또한 수행되고 있다[4]. 다양한 목적에 맞게 실내 자율주행은 다양한 플랫폼에서 연구가 수행되고 있다. 소형 보드인 Raspberry Pi는 값이 저렴하기 때문에 소형 자율주행 차량을 구현하기에 접근성이 좋지만, 많은 연구에서 활용되는 Nvidia의 Jetson TX2에 비하여 성능이 부족하다[5].

본 논문에서는 구조가 단순한 실내 환경에서 라이다 센서를 활용한 신경망 기반의 자율주행에 대한 시스템을 제안한다. 라이다 센서만 활용할 경우 처리해야 할 데이터의 수가 적고, 크기가 작은 신경망의 경우 연산량이 줄어 자원 효율적인 시스템을 구현할 수 있다. 라이다 센서는 빛을 발사하고 물체에 부딪혀서 돌아오는 시간을 바탕으로 거리를 계산한다. 빛이 물체에 부딪힐 때 물체와의 거리와 물체 표면의 재질에 따라 물체와의 거리가 정확하게 측정되지 않을 수 있다[6]. 이러한 문제점을 해결하기 위하여 라이다 센서에서 수집한 데이터에서 노이즈가 발생할 경우, 노이즈 주변의 데이터를 이용하여 노이즈를 제거한다. 그 후 가공된 데이터를 신경망에 입력하고 해당 위치에 알맞은 5가지의 주행 명령 중 하나를 예측한다. 차량은 예측한 명령에 따라 자율주행을 수행한다. 신경망에 입력되는 거리 데이터의 수가 많아지면 정확도가 상승할 수 있지만, 연산에 사용되는 자원 또한 증가한다. 본 논문에서는 사용되는 라이다 센서의 거리 데이터 수에 따른 신경망 모델의 성능평가를 진행한다. 실내에 구축된 자율주행 트랙에서 데이터를 수집하고 이를 입력 데이터의 수에 따라 6개의 신경망 모델을 생성한다. 마지막으로 실험을 통하여 각 학습 모델의 성능을 평가한다.

본 논문의 구성은 다음과 같다. 2장에서는 실내 자율주행에 대한 기존의 연구내용을 설명한다. 3장에서는 실내 자율주행을 위한 신경망 모델을 서술한다. 4장에서는 구현된 자율주행 차량과 실험환경을 설명한다. 5장에서는 실내 주행 트랙에서 실행된 실험 결과를 바탕으로 분류 성능과 응답속도 및 자원 효율성을 평가하며, 6장에서는 본 논문에서 제안하는 내용을 결론으로 정리한다.

2. 관련 연구

실내 자율주행은 다양한 방식으로 연구되고 있다. 대표적으로, SLAM은 지도 작성과 위치인식을 동시에 수행하는 기법으로, 방식에 따라 카메라와 같은 시각 센서를 활용하는 방식과 라이다 센서를 활용하는 방식으로 구분할 수 있다[7]. 지하 광산과 같이 카메라 센서를 활용할 수 없는 경우에 라이다 기반의 SLAM은 효과적으로 활용될 수 있다[8]. SLAM 이외에도 실내 자율주행을 수행하기 위해서 다양한 기계학습이 이용되기도 한다. 대표적으로 카메라로 수집한 이미지로 객체인식을 수행하는 YOLO(You Only Look Once)가 있다. [9]에서는 드론이 비행하는 과정에서 충돌을 방지하기 위해 계단을 인식하였고, [10]에서는 물류센터에서 자율주행 차량이 정해진 상자를 인식하기 위해서 YOLO가 사용되었다. 기계학습은 객체 인식 이외에도 활용될 수 있다. 자율주행을 수행하는 과정에서 바퀴의 미끄러짐이나 이동속도에 따라 오차가 발생할 수 있는데, [11]에서는 역전파 신경망과 CNN을 활용하여 오차를 보정하였다.

Raspberry Pi는 비교적 값이 저렴하기 때문에 자율주행 플랫폼으로 많이 활용되고 있다. 하지만 자율주행을 온전히 수행하기에는 성능이 부족한 측면이 있다. [12]에서는 Hector SLAM을 이용하여 지도 작성과 위치인식을 구현하였으나, 결과적으로 그 면적이 상당히 협소하다. [13]에서는 Gmapping을 이용한 SLAM을 구현하였지만, 차량 제어를 이용한 자율주행이 수행되진 않았다. [14]에서는 카메라에서 수집한 이미지를 바탕으로 차량의 움직임을 제어하였고, [15]에서는 라이다 센서를 기반으로 하여 객체 인식을 수행하여 충돌 방지 시스템을 구현하였다. 결과적으로 차량의 움직임을 제어하는 연구에서는 반대로 지도 제작과 위치인식은 수행되지 않았다.

3. 실내 자율주행을 위한 신경망 학습

Fig. 1는 본 논문에서 제안하는 실내 자율주행을 위한 자율주행 차량 시스템의 구조도이다. 자율주행 시스템은 라이

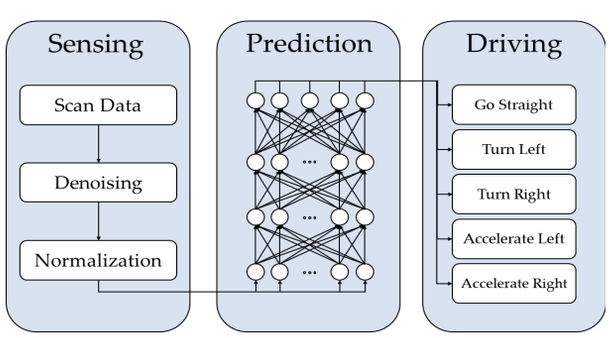

Fig. 1. Indoor Autonomous Driving System

다 센서에서 수집한 거리 데이터를 이용하여 현재 상태에 맞는 주행 명령을 예측하여 스스로 움직인다. 제안된 시스템은 Sensing, Prediction 그리고 Driving 기능을 수행하는 3가지 요소로 구성된다. Sensing은 라이다 데이터를 수집하여 가공하고, Prediction에 전달하는 역할을 수행한다. 라이다의 데이터는 다양한 센서들의 데이터를 이용할 수 있게 해주는 미들웨어인 ROS(Robot Operating System)를 통해 측정된다. ROS는 다양한 센서를 활용할 수 있도록 지원하는 미들웨어다. Prediction에서는 신경망을 이용하여 현재 자율주행 차량이 수행해야 하는 주행 명령을 예측하는 기능을 한다. 신경망의 학습은 자율주행 차량이 이동하는 경로에서 미리 수집된 데이터를 바탕으로 수행된다. 추정된 주행 명령은 Driving으로 전달된다. Driving은 예측된 주행 명령을 바탕으로 차량을 제어한다. 주행 명령은 모터 드라이버에 전달되어 자율주행 차량의 바퀴에 장착된 모터를 구동시킨다.

3.1 학습 데이터 구성

본 논문에서는 실내 주행환경을 구축하고 라이다 센서를 이용하여 직접 데이터를 수집하였다. 라이다 센서는 360°를 회전하면서 1회 측정을 통해 360개의 거리 데이터를 생성한다. 본 논문에서는 신경망에 입력되는 데이터의 수에 따른 성능평가를 위해 6가지의 학습 모델을 생성한다. 각 모델의 학습을 위해 데이터의 측정 폭을 조정하여 6가지의 거리 데이터의 세트로 분류하였다. Equation (1)~(6)은 본 논문에서 사용하는 라이다 데이터 세트를 표현하고 있다. 각 데이터 세트는 $Data_{\mathrm{n}}$으로 표현되며 n은 하나의 데이터 세트를 구성하는 거리 데이터의 수를 의미한다. 각 데이터는 360°에서 일정한 간격으로 분포되어 있다.

$$\begin{cases} Data_{8} = [range_0, range_{45}, \ldots, range_{270}, range_{315}] & (1) \\ Data_{18} = [range_0, range_{20}, \ldots, range_{320}, range_{340}] & (2) \\ Data_{45} = [range_0, range_{8}, \ldots, range_{344}, range_{352}] & (3) \\ Data_{90} = [range_0, range_{4}, \ldots, range_{352}, range_{356}] & (4) \\ Data_{180} = [range_0, range_{2}, \ldots, range_{356}, range_{358}] & (5) \\ Data_{360} = [range_0, range_{1}, \ldots, range_{358}, range_{359}] & (6) \end{cases}$$

Equation (1)은 8개 방향의 거리 데이터를 의미한다. 각 데이터는 45도 간격으로 분포되어 있고, 데이터의 수는 8개이다. Equation (2)는 20도 간격으로 수집되었고, 데이터의 수는 18개이다. Equation (3)은 8도 간격으로 측정된 45개의 거리 데이터, Equation (4)는 4도 간격으로 측정된 90개의 거리 데이터를 나타낸다. 그리고 Equation (5)는 2도 간격으로 측정된 180개의 거리 데이터 세트이다. 마지막으로 Equation (6)은 간격을 증가시키지 않은 360개의 온전한 거리 데이터이다.

Fig. 2는 실내 환경에서 주행 시 위치에 맞게 수행해야 하는 주행 명령의 예시를 보여준다. 주행 명령은 ①-좌회전(Turn Left), ②-좌측 바퀴 가속(Accelerate Left), ③-직진(Go Forward), ④-우측 바퀴 가속(Accelerate Right) 그리고 ⑤-우회전(Turn Right)으로 5가지로 구분된다. 명령은 직진해야 하는 상황과 회전해야 하는 상황으로 나뉜다. 직진해야 하는 상황은 3가지의 명령으로 구분된다. 직진 환경에서는 좌우의 벽이나 장애물과 충돌하지 않고 안정적인 주행을 위해 차체의 중심을 주행 경로의 중심으로 이동하도록 수행된다. 자체가 중심에 위치하는 경우 ③-직진 명령이 수행된다. 차체가 중심보다 좌측으로 치우쳐진 경우 ②-좌측 바퀴 가속 명령이 수행된다. 반대로 차체가 중심보다 우측으로 치우쳐진 경우는 ④-우측 바퀴 가속 명령이 수행된다. 분기점이나 모서리 구간에서는 ①-좌회전과 ⑤-우회전 명령이 수행된다. 모서리 구간에서는 주행할 수 있는 경로를 따라서 수행되지만, 분기점에서는 같은 분기점이라도 경로에 따라 수행되는 명령이 다르다.

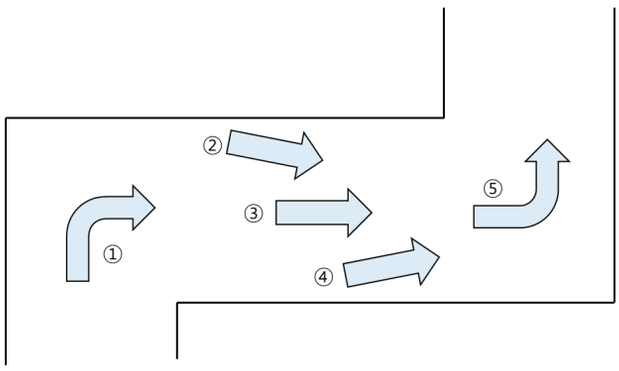

Fig. 2. Autonomous Driving Commands

3.2 데이터 가공

본 논문에서 사용된 라이다 센서는 거리가 측정되지 않으면 정상적인 값을 반환하지 않아 노이즈를 발생시킨다. 노이즈가 섞일 경우 동일한 지점이라도 데이터를 샘플링하는 간격에 따라 학습의 결과가 달라질 수 있고, 학습이 완료된 신경망 모델의 정확도를 떨어뜨릴 수 있다. 본 논문에서는 노이즈를 제거하여 학습의 신뢰성을 개선하였다.

$$f(x_n) = \begin{cases} x_n & (x_n \neq 0) \\ \dfrac{f(x_{n-1}) + f(x_{n+1})}{2} & (x_n = 0) \end{cases} \quad (7)$$

$$(n = 1, 2, \ldots 360)$$

Equation (7)은 노이즈를 필터링하는 기능을 수행한다. 정상적으로 거리 데이터를 수집한 경우 그 값을 저장하고, 0인 경우 인접한 데이터를 이용하여 그 값을 예측한다. Fig. 3은 특정 지점에서 측정된 거리 데이터의 분포를 나타낸다. 그

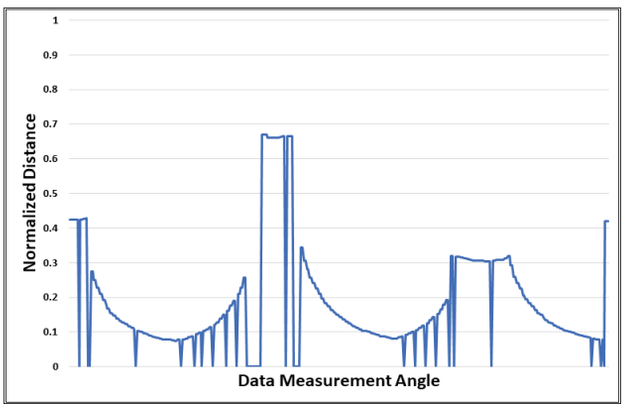

Fig. 3. Normalized Range Data

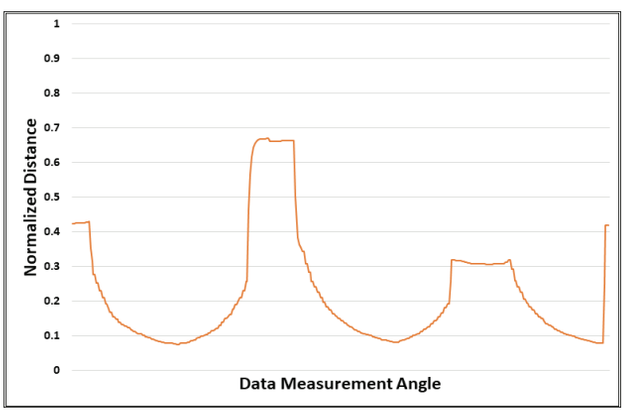

Fig. 4. Normalized Range Data with Denoising

래프의 x축은 데이터가 수집되는 시점인 0°에서 359°까지 360개의 데이터의 순서를 나타낸다. 그래프의 y축은 정규화된 거리 데이터의 크기를 나타낸다. Fig. 4는 이러한 노이즈를 제거한 데이터의 분포를 보여준다. 데이터에서 노이즈가 발생한 경우, 주변의 데이터를 기반으로 하여 새로운 데이터를 생성하여 노이즈를 제거한다.

3.3 신경망 모델

Fig. 5는 본 논문에서 사용하는 신경망 모델을 나타낸다. 해당 모델은 입력된 라이다 거리 데이터를 바탕으로 현재 상황에 맞는 최적의 주행 명령을 예측한다. 신경망 모델은 완전연결 계층(Fully connected layer) 형태로 입력층(Input Layer), 중간층(Hidden Layer) 그리고 5가지의 주행 명령으로 구분하기 위한 출력층(Output Layer)으로 구성되어 있다. 중간층의 활성화 함수는 ReLU를 사용하였고, 출력층은 주행 명령 분류를 위하여 Softmax를 사용하였다. 모델의 손실함수는 Cross Entropy를 사용하였다. Fig. 4의 n은 입력의 크기를 나타낸다. 입력의 크기는 3.1에서 분류한 6가지 학습 데이터 세트에 대응된다. Equation (1) 데이터를 사용하는 신경망 학습모델은 'Model-8'이다. 동일한 의미로

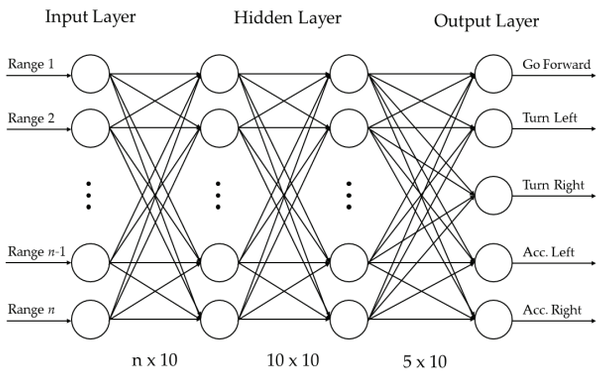

Fig. 5. Neural Network for Autonomous Driving

Equation (2)의 학습 데이터를 사용하는 모델은 'Model-18'이다. 'Model-45', 'Model-90', 'Model-180' 그리고 'Model-360'은 각각 Equation(3)~(6)에 대응된다. 제한된 환경에서의 성능평가를 위해서 epoch는 10,000으로 설정하였다. 모델의 학습률은 학습 과정을 반복하면서 도출하였다. 학습률을 0.3으로 설정하였을 때, 제한된 epoch에서 가장 정확하게 학습을 완료하였다.

4. 구현 및 실험

4.1 자율주행 차량

Fig. 6은 구현된 자율주행 차량을 보여준다. 자율주행 차량은 Raspberry Pi 4 Model B를 기반으로 구성되어 있다. Raspberry Pi는 교육용으로 개발된 작고 저렴한 보드이다. 가격 대비 우수한 성능을 가지며, 리눅스 기반의 OS를 이용하여 자율주행을 위한 시스템을 개발하기에 용이하다. 데이터를 수집하는 라이다 센서는 SLAMTEC RPLIDAR A1M8 모델을 사용하였다. 해당 라이다 센서는 360° 회전하며 최소

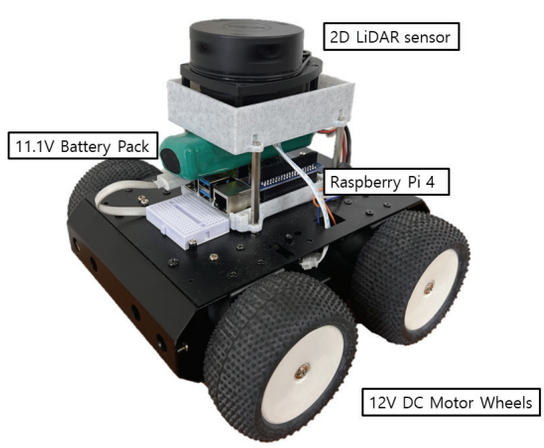

Fig. 6. Autonomous Vehicle with Neural Network

Table 1. Specifications for Raspberry Pi and LiDAR

	Raspberry Pi 4 Model B
CPU	Broadcom BCM2711, Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
RAM	2GB LPDDR4-3200 SDRAM
OS	Ubuntu MATE 20.04 LTS
	RPLIDAR A1M8
Distance Range	0.15-12.0 meter
Angular Range	0-360 degree
Scan Rate	5.5 Hz

0.15m에서 최대 12m까지 거리 측정이 가능하다. 구동을 위해 12V DC 모터 4개에 바퀴를 연결하였고, 메인 보드와 모터를 제어하기 위한 모터 드라이버에 전원을 공급하기 위한 11.1V의 배터리팩이 탑재되었다. Table 1은 Raspberry Pi와 라이다 센서의 사양을 나타낸다[16, 17]

4.2 자율주행 환경과 데이터 수집

학습데이터를 수집하기 위해 4.1에서 구현한 자율주행 차량을 이용하였다. Fig. 7은 자율주행을 위해 실내에 구축한 실내 주행 트랙이다. 라이다를 탑재한 자율주행 차량을 자율주행 트랙에 배치하고, 트랙을 따라 천천히 이동하여 이동한 곳의 거리 데이터를 수집하였다.

Fig. 8은 해당 트랙에서 데이터를 수집한 경로를 나타낸다. 자율주행 차량은 'Start'에서 출발하여 ①~⑦ 순서로 주행하여 'Finish'에서 정지한다. 주행 과정에서 수집되는 라이다 데이터를 저장하였고, 각 위치에서 요구되는 주행 명령을 해당 위치에서 수집한 데이터에 레이블링(Labeling)하였다. 수집 과정에서 360개의 거리 데이터로 구성된 1,100개의 데이터 세트가 수집되었다. 수집된 데이터에서 770개는 학습에 사용되었고, 330개는 평가에 사용되었다.

4.3 학습 데이터 생성

실험 단계에서는 신경망 입력의 크기에 따라서 성능을 비교하는 것을 목표로 한다. 라이다 데이터는 360°를 1° 간격으로 수집하여 총 360개의 데이터로 표현된다. 입력 데이터의 수를 조절하기 위해 데이터 수집 간격을 조절하였고, 입력

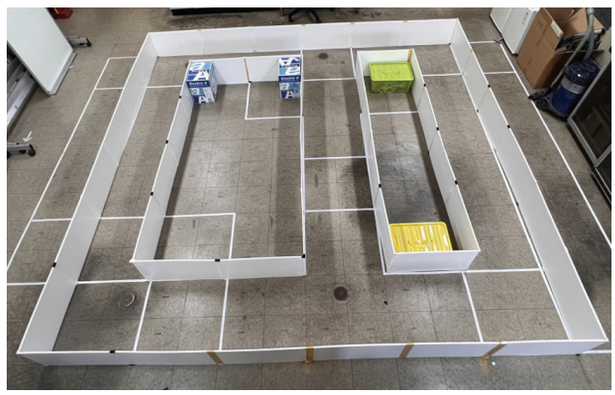

Fig. 7. Indoor Autonomous Driving Track

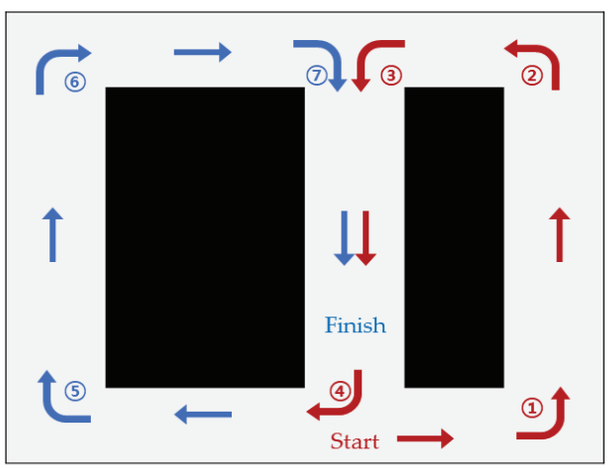

Fig. 8. Path in a Driving Track

데이터의 크기에 따라 6가지의 데이터 세트를 생성하였다.

Table 2는 입력의 크기가 360인 모델을 위해서 생성한 학습 데이터의 일부를 보여준다. R_0는 라이다 센서의 0°에서 수집된 거리 데이터를 나타낸다. 거리 데이터는 수집된 데이터 중에서 가장 큰 값을 기준으로 하여 0~1 사이의 값으로 정규화하였다. 본 데이터는 분류 모델을 학습하기 위한 데이터 세트이다. 그러므로 거리 데이터 세트에는 해당 데이터 세트가 수집된 지점에 알맞은 주행 명령이 정답으로 레이블링 되었다. 주행 명령은 C_0부터 C_4까지 각각 '직진', '좌회전', '우회전', '좌측 바퀴 가속' 그리고 '우측 바퀴 가속'에 대응된다.

Table 2. Part of Training Data Set

R_0	R_1	...	R_{358}	R_{359}	C_0	C_1	C_2	C_3	C_4
0.648	0.652	...	0.074	0.068	1	0	0	0	0
0.153	0.155	...	0.115	0.111	0	0	0	1	0
0.189	0.189	...	0.040	0.038	0	0	0	0	1
0.052	0.052	...	0.074	0.072	0	1	0	0	0
0.078	0.078	...	0.080	0.074	0	0	1	0	0

5. 성능 평가

본 실험은 자원이 제한적인 모바일 엣지 환경에서 예측 모델을 학습하는 것을 중점을 둔다. 수집한 모든 데이터를 학습하는 대신 데이터의 일부를 추출하여 사용하는 모델과 모든 데이터를 사용하는 모델을 비교하여 줄어든 연산량에서 발생하는 정확도 변화와 자원 효율성을 평가하고자 한다. 자율주행 차량을 위해 신경망 모델은 입력의 크기에 따라 6개가 생성되었다. 각 모델은 모바일 엣지에서의 운용을 위해 분류 성능, 학습 속도, 응답시간 그리고 배터리 소모량을 비교하였다.

5.1 분류 성능평가

Table 3은 6개의 모델의 성능 평가를 비교한 결과를 나타낸다. 성능평가의 신뢰성을 위하여 6개 모델 각각 10회씩 학습하였고, 성능평가 지표의 평균을 바탕으로 계산되었다. 각 지표는 Equation (8)~(11)을 이용하여 계산하였다.

$$\begin{cases} Accuracy = \dfrac{TP+TN}{TP+FP+TN+FN} & (8) \\ Precision = \dfrac{TP}{TP+FP} & (9) \\ Recall = \dfrac{TP}{TP+FN} & (10) \\ F1\,score = 2 \times \dfrac{Precision \times Recall}{Precision + Recall} & (11) \end{cases}$$

모든 테스트 데이터에 대하여 정답을 정답으로 예측한 경우(TP, True Positive), 오답을 오답으로 예측한 경우(TN, True Negative), 오답을 정답으로 예측한 경우(FP, False Positive) 그리고 정답을 오답으로 예측한 경우(FN, False Negative)로 구분한다. 각각의 수치는 5가지의 주행 명령의 평균으로 계산된다. 정확도(Accuracy)는 주어진 모든 결과에 대하여 정답과 오답으로 예측한 비율을 나타내며, 모델의 입력 크기에 비례하여 증가하는 것을 확인할 수 있다. 입력의 크기가 증가할 때 평균적으로 0.0141만큼 증가하였다. 입력의 크기 차이가 가장 큰 model-180와 model-360 사이에서 가장 큰 상승 폭이 있었고, 0.0219만큼 정확도가 상승하였다. 정밀도(Precision)는 정답으로 예측한 경우에 대해서 실제로 정답인 비율을 나타낸다. 재현율(Recall)은 실제 정답인 경우에 대해서 정답으로 예측한 비율을 나타낸다. 본 실험에서는 모든 모델에 대하여 정밀도보다 재현율이 높게 나타낸다. F1 Score는 정밀도와 재현율의 조화평균으로, 불균형한 데이터 분포에서 좀 더 정확하게 성능을 비교할 수 있는 지표이다. 위 실험에서는 회전 구간이 많아 직진 명령을 예측해야 하는 데이터와 회전 명령을 예측하는 데이터가 비슷한 비율로 분포되어있다. 하지만 주행환경에 따라 직진 명령을 수행하는 구간이 많은 경우가 있고, 극단적으로 대부분 회전 명령만 수행하는 미로 같은 환경이 있을 수 있다. F1 Score를 비교할 경우, Model-8은 0.5 이하의 낮은 성능을 보인다. 그리고 중간 단계의 모델들은 0.5에서 0.8 사이의 값을 가지며, 입력의 크기가 가장 큰 Model-360에서 0.8275의 값을 나타낸다. Model-8과 Model-360을 비교할 경우 정확도에서 7.9%만큼 성능 차이가 나타나고, F1 Score에서 68.05%의 성능 차이가 발생한다.

Table 3. Evaluation Comparison Between Models

	Accuracy	Precision	Recall	F1 Score
model-8	0.8953	0.4480	0.5504	0.4924
model-18	0.9051	0.4705	0.5778	0.5157
model-45	0.9101	0.4776	0.587	0.5237
model-90	0.9269	0.5735	0.6676	0.6143
model-180	0.9442	0.6784	0.7493	0.7086
model-360	0.9661	0.8107	0.85	0.8275

5.2 학습 속도

모바일 환경에서 학습의 속도를 높여 학습 시간 동안의 전력 소모량을 줄여야 한다. 하지만 신경망에 입력되는 데이터의 수를 줄이면 모든 파라미터를 1회 학습하는데 필요한 연산의 수는 줄어들지만, 그만큼 학습하는 파라미터의 수도 줄어 정확도가 낮아질 수 있다. 또한 입력 데이터를 늘리면 정확도는 상승할 수 있지만 학습 시간이 길어질 수 있다. 이를 평가하기 위하여, 6개의 모델에 대하여 정확도가 0.95에 도달할 때까지 각 모델에 대하여 10회 학습을 진행하였고 이때 소요되는 시간의 평균을 비교하였다. 학습 과정에서 epoch는 최대 500,000으로 설정하였다.

Fig. 9는 각 모델이 정확도가 0.95에 도달할 때까지 소요된 시간을 보여준다. 6개의 학습 모델 중 Model-8은 10회

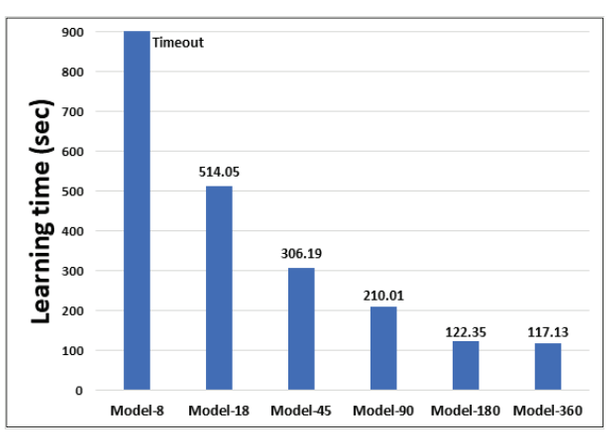

Fig. 9. Learning Time of Prediction Models

학습을 진행하는 과정에서 최대 epoch에 도달할 때까지 정확도가 0.95에 도달하지 못하였다. Model-18은 514.05초, Model-45는 306.19초가 소요되었고, 두 모델은 각각 평균 epoch가 49,856 그리고 29,696이다. 두 모델을 비교하였을 때, 입력 데이터의 수가 150% 증가하였고, 학습 시간은 67.9% 감소하였다. Model-90은 평균 18,601 epoch에서 정확도가 0.95에 도달하였다. 이때 210.01초가 소요되었다. model-180에서는 학습 시간이 71.6% 감소하였고, 입력 데이터의 수가 증가함에 따라 학습 시간이 가장 큰 비율로 감소하였다. 이때 epoch는 9305로 5.1에서 수행한 평가 환경과 가장 유사하다. 라이다에서 생성된 모든 거리 데이터를 사용한 Model-360은 epoch의 값이 7,339일 때 정확도가 0.95에 도달하였고, 학습 시간은 117.13초이다. 학습하는 거리 데이터의 수가 증가할 때마다 학습 시간은 평균적으로 29.5% 만큼 감소하였다.

5.3 응답시간

자율주행과 같이 차량이 이동하여 다른 객체들과 실제 공간에서 상호작용을 하는 경우 응답시간이 길어지면 물리적인 충돌 및 오작동이 발생할 수 있다. 그러므로 지연시간이 발생하지 않게 주행 명령을 예측할 필요가 있다.

Fig. 10은 6개의 예측 모델들의 응답시간을 보여주는 그래프이다. 각 모델별로 1,000,000회의 데이터를 입력하였고, 이 중 모델별로 상위 5%와 하위 5%를 제외한 데이터의 평균을 비교하였다. 응답시간은 모델의 입력 데이터의 수가 증가할수록 증가한다. Model-8의 응답시간은 378.903마이크로초(μs)초 가장 빠르며 Model-360의 응답시간은 417.985마이크로초가 소요되어 가장 느리다. 입력의 크기가 커질수록 평균적으로 2%만큼 응답시간이 증가하였다. 입력의 크기가 작은 모델에서 입력의 크기가 커질수록 신경망 모델의 입력층에서 처리하는 연산량이 2배 이상 증가하여 응답시간의 증가 폭은 더 커지게 된다.

5.4 배터리 소비량

모바일 환경에서 영속성은 모바일 객체의 활동 반경 및 서비스 시간에 큰 영향을 주는 요인이다. 자원 효율적인 모델을 적용하여 제한된 배터리 수명을 최대한 연장하는 것이 필요하다. Fig. 11은 6개의 예측모델에 주기적으로 해당 모델에 맞는 라이다 데이터를 입력하여 10분 동안 주행 명령을 예측하면서 소요되는 배터리 소비량을 측정하였고, 각 모델 별로 10회 측정하여 평균을 비교하였다. 대기상태(Idle)에서는 평균 80.6 mAh만큼 전력을 소비하였다. 처리해야하는 입력 데이터가 가장 적은 Model-8은 평균적으로 93.6 mAh만큼 소비하였고, 입력 데이터의 수가 가장 많은 Model-360은 평균적으로 95.8 mAh만큼 전력량이 측정되었다. 배터리 소비량

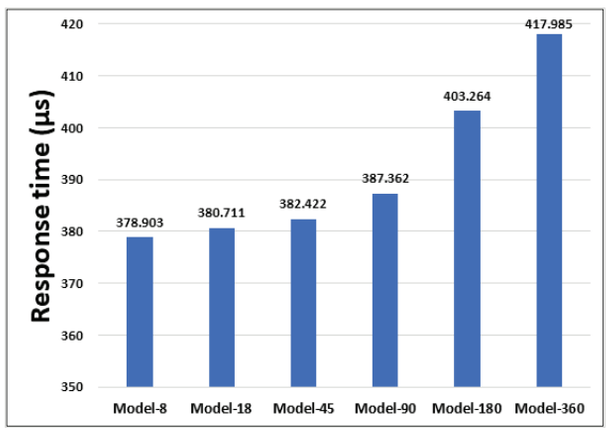

Fig. 10. Response Time of Prediction Models

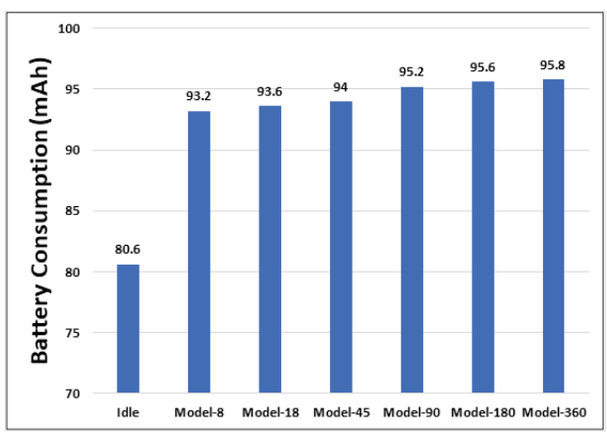

Fig. 11. Battery Consumption of Prediction Models

은 입력 데이터의 수가 증가할수록 증가하지만, 데이터의 수에 비례하여 크게 증가하지 않는다. 대기 상태와 Model-8를 비교하면 배터리 소비량은 12.6mAh 만큼 증가였지만, Model-8에서 Model-360까지 입력의 크기가 증가하는 동안 배터리 소비량은 평균 0.52 mAh 증가하였다. 이는 주행 명령을 예측하여 반환하는 프로그램의 전체 연산에서 주행 명령을 예측하여 반환하는 신경망의 연산의 비중이 그리 크지 않은 걸로 판단된다.

5.5 주행 성능 평가

본 논문에서는 신경망을 이용하여 라이다 센서 데이터를 주행 명령으로 예측하는 모델을 설계하였다. 평가 데이터를 이용하여 신경망의 분류 성능을 비교하는 것과 달리 실제로 주행 중 라이다 센서를 이용하여 측정한 거리 데이터를 이용하였다. 각 모델에 거리 데이터를 입력하였을 때 나타나는 예측 결과를 실제 수행되어야 하는 주행 명령과 비교하였다.

Fig. 12는 6개의 모델에 대한 실험 결과를 보여준다. 주행 경로는 Fig. 8에 대응한다. 주행 중 총 153번의 주행 명령을

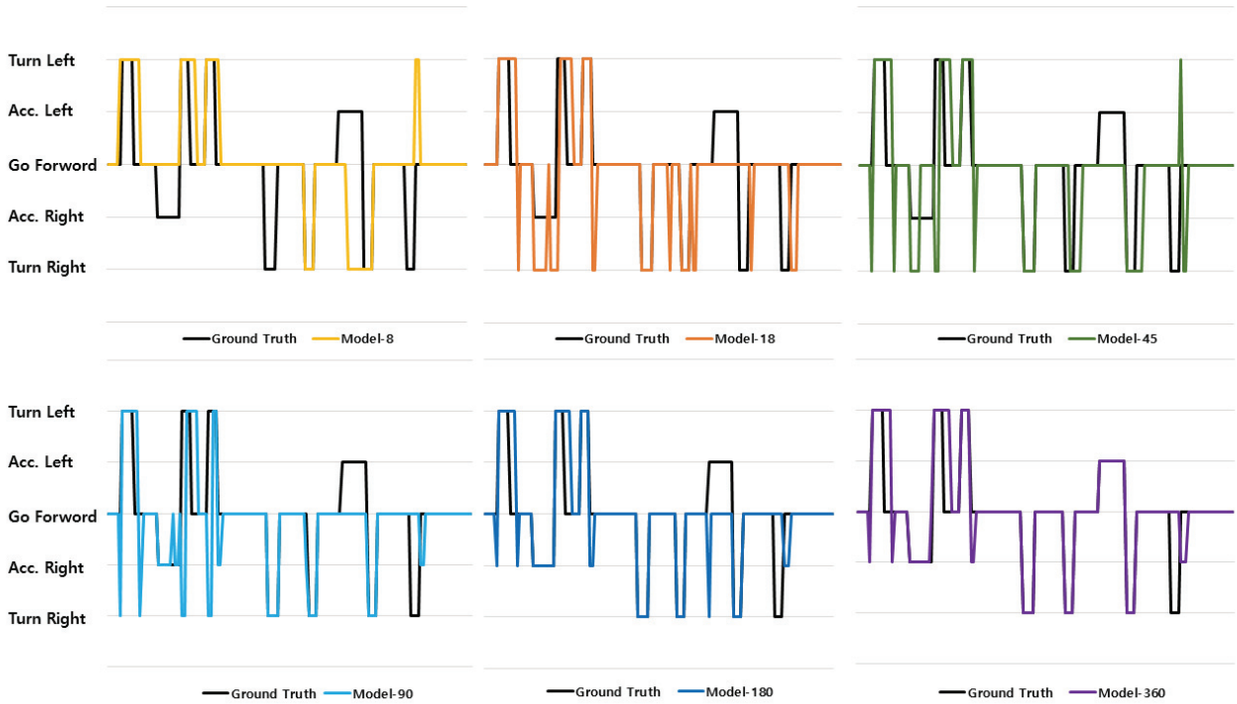

Fig. 12. Prediction Results of each Model while Driving

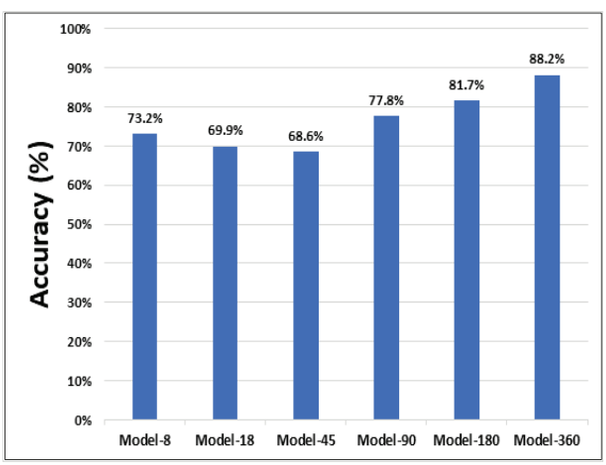

Fig. 13. Prediction Accuracy while Driving

예측하였다. Fig. 12에서 그래프의 중점은 그래프의 위에서부터 좌회전, 좌측 바퀴 가속, 직진, 우측 바퀴 가속, 우회전에 대응한다. 그래프는 실제 주행 중 수행되어야 하는 명령과 신경망에서 예측한 주행 명령을 비교한다. 그래프의 중심은 직진에 해당하고 회전 명령을 수행하면 위쪽과 아래쪽으로 값이 이동한다. Fig. 13은 각 모델에 대하여 주행 중 수행된 153번의 예측에서 정확하게 주행 명령을 예측한 비율을 보여준다. Model-360에서 가장 정확하게 측정하였다. Model-180 모델에서는 81.7%의 정확도를 보이고, 그 외의 모델에서는 80% 이하의 정확도를 보인다. 정확도가 낮은 모델에서는 직진 구간과 회전 구간에서 주행 명령 예측이 정교하게 수행되지 않았다. Model-360에서도 직진 구간과 회전 구간의 경계에서의 예측의 정확도는 낮지만, 직진 구간에서 차체가 치우쳤을 때, 좌측 바퀴 가속 명령과 우측 바퀴 가속 명령에 대해서 다른 모델에 비해 높은 정확도를 보인다. Model-18과 Model-45에서는 입력 데이터의 수가 가장 적은 Model-8보다 낮은 정확도를 보인다. 5.1의 분류성능평가 Model-8과 Model-18 그리고 Model-45 모델은 F1 Score에서 큰 차이를 보이지 않는다. 이는 입력 데이터의 개수가 일정 수준 이상이 되어야 유의미한 성능의 변화가 있다는 것을 보여준다.

6. 결 론

실내 자율주행은 다양한 방법으로 구현되고 있으나 성능이 부족한 라즈베리 파이에서는 SLAM과 차량 제어 알고리즘을 동시에 구현하기 힘들다. 본 논문에서는 신경망 학습만을 사용한 자율주행 기법에 대한 방식을 제안하였다. 제안된 신경망 학습은 라이다 센서에서 측정한 거리 데이터를 입력으로 받아 현재 상황에 맞는 주행 명령을 수행하였다. 신경망은 라이다 데이터를 입력하면 5가지의 주행 명령 중 하나의 명령을 예측하고 수행한다. 또한 입력 데이터의 수가 성능에 미치

는 영향을 측정하기 위해 입력 데이터의 개수에 따라 6가지의 신경망 모델을 생성하여 성능평가를 진행하였다. 생성된 모델은 분류 성능과 응답속도 및 배터리 소비량을 중점으로 평가되었다. 실험을 위해 라즈베리 파이를 이용한 자율주행 차량을 구현하였고, 데이터 수집을 위해 실내에 주행 트랙을 설치하였다. 그리고 구현된 자율주행 차량을 이용하여 주행 트랙의 경로를 따라 데이터를 수집하였다. 신경망 학습과 성능평가는 모두 자율주행 차량에서 수행되었다. 성능평가에서 6개의 신경망 모델의 분류성능을 비교하였다. 정확도, 정밀도, 재현율 그리고 F1 Score에 대하여 입력 데이터의 수가 증가할수록 분류 성능이 향상되었다. 다음으로 정확도가 0.95에 도달할 때까지 소요되는 학습 시간을 비교하였다. 입력 데이터가 적으면 모델의 정확도가 상승하지 않거나 시간이 많이 소요되었고, 입력 데이터의 수가 가장 많은 모델에서 가장 빠르게 학습되었다. 그리고 응답시간과 배터리 소모량은 입력 데이터의 수에 따라 영향을 받았으며 입력 데이터의 수가 증가할수록 응답시간은 평균 2%만큼 증가하였고, 배터리 소모량은 평균 0.55%만큼 증가하였다. 마지막으로 주행 중 수집되는 데이터를 바탕으로 주행 명령을 예측하였을 때 Model-360에서가장 좋은 성능이 나타났고, 88.2%만큼의 정확도를 보였다. 향후 연구에서 다중 센서를 이용한 환경에서 센싱 주기 조절과 입력 데이터의 변화에 따른 신경망 모델 확장을 통하여 확장된 성능 평가를 수행할 예정이다.

References

[1] W. Kang, Y. Jung, and I. Hwang, "Major technology of autonomous driving," *KICS Information & Communication Magazine Open Lecture Series*, Vol.35, No.1, pp.28-35, 2018.

[2] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet of Things Journal*, Vol.5, No.1, pp.450-465, 2018.

[3] S. Jeong, "Edge computing and vehicles: Opportunities and challenges for the future," *The Journal of Korean Institute of Communications and Information Sciences*, Vol.46, No.5, 2021.

[4] N. El-Sheimy and Y. Li, "Indoor navigation: State of the art and future trends," *Satellite Navigation*, Vo1.2, No.1, pp.1-23, 2021.

[5] A. A. Süzen, B. Duman, and B. Şen, "Benchmark analysis of jetson TX2, jetson nano and raspberry PI using Deep-CNN," *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pp.1-5, 2020.

[6] J. Noh, K.-M. Yang, M.-R. Park, J.-W. Lee, M.-G. Kim, and K.-H. Seo, "LiDAR point cloud augmentation for mobile robot safe navigation in indoor environment," *Journal of Institute of Control, Robotics and Systems*, Vol.28, No.1, pp.52-58, 2022.

[7] Q. Zou, Q. Sun, L. Chen, B. Nie, and Q. Li, "A comparative analysis of LiDAR SLAM-based indoor navigation for autonomous vehicles," *IEEE Transactions on Intelligent Transportation Systems*, Vol.23, No.7, pp.6907-6921, 2022.

[8] H. Kim and Y. S. Choi, "Development of a ROS-Based autonomous driving robot for underground mines and its waypoint navigation experiments," *Tunnel & Underground Space*, Vol.32, No.3, pp.231-242, 2022.

[9] Y. J. Choi, T. Rahim, I. N. A. Ramatryana, and S. Y. Shin, "Improved CNN-based path planning for stairs climbing in autonomous UAV with LiDAR sensor," *2021 International Conference on Electronics, Information, and Communication (ICEIC)*, pp.1-7, 2021.

[10] D. G. Park, K. R. Lee, J. W. Jang, and D. H. Kim "ROS-based control system for localization and object identification of indoor self-driving mobile robot," *Transactions of the KSME A*, Vol.45, No.12, pp.1149-1160, 2021.

[11] Y. Quan, L. Huang, L. Ma, Y. He, and R. Wang, "Neural Network-Based Indoor Autonomously-Navigated AGV Motion Trajectory Data Fusion," *Automatic Control and Computer Sciences*, Vol.55, No.4, pp.334-345, 2021.

[12] D. T. Son, M. T. Anh, D. D. Tu, L. V. Chuong, T. H. Cuong, and H. S. Phuong, "The practice of mapping-based navigation system for indoor robot with RPLIDAR and raspberry Pi," *2021 International Conference on System Science and Engineering (ICSSE)*, IEEE, 2021.

[13] M. Liao, D. Wang, and H. Yang. "Deploy indoor 2D laser SLAM on a raspberry pi-based mobile robot," *2019 11th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, Vol.2. IEEE, 2019.

[14] H. León Araujo et al., "Autonomous mobile robot implemented in LEGO EV3 integrated with raspberry Pi to use android-based vision control algorithms for human-machine interaction," *Machines*, Vol.10, No.3, pp.193, 2022.

[15] M. Calcroft and A. Khan, "LiDAR-based obstacle detection and avoidance for autonomous vehicles using raspberry Pi 3B," *2022 UKACC 13th International Conference on Control (CONTROL)*, pp.24-29, 2022.

[16] Raspberry Pi [Internet], https://www.raspberrypi.org

[17] RPLIDAR A1M8 [Internet], https://www.slamtec.com

권 용 훈

https://orcid.org/0000-0003-3570-0843
e-mail : russell1113@kangwon.ac.kr
2019년 강원대학교 원예학과(학사)
2021년 강원대학교 컴퓨터정보통신공학과 (석사)
2021년 ~ 현 재 강원대학교 컴퓨터정보통신공학과 박사과정
관심분야 : Internet of Things, Edge Computing, Autonomous Driving

정 인 범

https://orcid.org/0000-0003-4451-6917
e-mail : ibjung@kangwon.ac.kr
1985년 고려대학교 전자공학과(학사)
1985년 ~ 1995년 삼성전자 컴퓨터시스템사업부 선임연구원
1994년 한국과학기술원 정보통신공학과 (석사)
2000년 한국과학기술원 전산학과(박사)
2001년 ~ 현 재 강원대학교 컴퓨터정보통신공학과 교수
관심분야 : Internet of Things, Edge Computing, On-Device AI