JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지 Vol. 25, No. 7: 977~985, Jul. 2021

멀티채널 LiDAR 센서 기반 차량 검출 플랫폼을 위한 효율적인 저전력 신호처리 기법

정태원[1] · 박대진[2*]

Efficiency Low-Power Signal Processing for Multi-Channel LiDAR Sensor-Based Vehicle Detection Platform

Taewon Chong[1] · Daejin Park[2*]

[1]Senior Engineer, CARNAVICOM Co., Ltd., Incheon, 41566 Korea
[2*]Assistant Professor, School of Electronics Engineering, Kyungpook National University, Daegu, 41566 Korea

요 약

자율주행 차량이 주목받게 되면서 LiDAR 센서가 대두되었다. LiDAR 센서는 LASER를 이용하여 범위 내에서 특정 지점까지 측정된 거리 값을 3차원 정보로 제공한다. 3차원 거리 값인 만큼 방대한 데이터를 전송하게 되고, 차량의 메인 프로세서 등에서 다른 데이터와 같이 이를 실시간으로 처리하기에는 무리가 있다. 이러한 이슈를 해결하기 위해 통합처리 시스템을 개발하고자 한다. 시스템은 센서로부터 데이터를 받아 처리하는 client와 각 client로부터 데이터를 취합하여 이를 외부로 전송하는 server 프로세스로 구성된다. 각 프로세스의 데이터 수신 및 처리 방법, 프로세스 구동 방법을 변화시켜가며 시스템의 실시간성 확보를 위한 테스트를 진행하였다. 실험 결과, 4대의 LiDAR 센서로 데이터를 수신 받도록 하였으며, background 나 multi-core processing을 적용하여 프로세스를 동작시켰을 때, 각 client는 약 13.2 ms, server는 약 12.6 ms의 응답시간을 확인할 수 있었다.

ABSTRACT

The LiDAR sensor is attracting attention as a key sensor for autonomous driving vehicle. LiDAR sensor provides measured three-dimensional lengths within range using LASER. However, as much data is provided to the external system, it is difficult to process such data in an external system or processor of the vehicle. To resolve these issues, we develop integrated processing system for LiDAR sensor. The system is configured that client receives data from LiDAR sensor and processes data, server gathers data from clients and transmits integrated data in real-time. The test was carried out to ensure real-time processing of the system by changing the data acquisition, processing method and process driving method of process. As a result of the experiment, when receiving data from four LiDAR sensors, client and server process was operated using background or multi-core processing, the system response time of each client was about 13.2 ms and the server was about 12.6 ms.

키워드 : LiDAR 센서, 임베디드 시스템, 병렬 처리, 저전력 신호 처리
Keywords : LiDAR sensor, Embedded system, Parallel processing, Low-power signal processing

Received 8 June 2021, **Revised** 9 June 2021, **Accepted** 15 June 2021

*** Corresponding Author** Daejin Park (E-mail:boltanut@knu.ac.kr, Tel:+82-53-950-5548)
Assistant Professor, School of Electronics Engieering, Kyungpook National University, Daegu, 41566 Korea

Open Access http://doi.org/10.6109/jkiice.2021.25.7.977 print ISSN: 2234-4772 online ISSN: 2288-4165

ⓒThis is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License(http://creativecommons.org/li-censes/ by-nc/3.0/) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.
Copyright ⓒ The Korea Institute of Information and Communication Engineering.

Ⅰ. 서 론

자율주행 차량이 시장에 대두되면서 차량 주변의 물체 및 장애물 등을 감지하기 위한 센서가 주목 받기 시작했다. 카메라, 초음파 센서, RADAR 센서는 현재 차량에 활발히 탑재되는 센서들이지만 해당 센서들의 장단점에 의해 자율주행 차량에 적용하기엔 한계점을 맞이하였다 [1].

카메라는 영상을 기반으로 하기 때문에 이를 이용한 객체 인식 시, 다른 센서보다 정확한 물체의 인식 및 검출이 가능하다는 장점을 가지지만, 야간과 같이 시야나 조도가 제한되는 상황에서 인식 능력이 현저히 저하된다는 점, 거리를 정확히 측정하는 것이 불가능하다는 단점을 가진다 [2]. RADAR 센서는 다른 센서들에 비해 장거리까지 거리 측정이 가능하다는 장점이 있지만, 고정된 물체에 대한 검출 능력이 다소 떨어지며, 물체의 종류를 분간하기 어렵다는 단점을 가진다 [3]. 초음파 센서는 다른 센서에 비해 가격적인 면에서 강점을 가지는 센서이지만, 측정 가능 거리가 보다 짧다는 단점을 가진다 [4].

기존 센서들의 단점을 보완하기 위해 light detection and range (LiDAR) 센서가 대두되었다. LiDAR 센서는 이름과 같이 LASER를 이용하여 특정 지점(point)까지의 거리를 time of flight(ToF)나 phase shift(PS) 방식을 통해 측정하며, 회전 거울 등을 통해 측정 범위를 제어하게 된다 [5]. 센서를 통해 측정된 거리 값들이 모이면 이는 군집을 이루게 되고 이를 포인트 클라우드라 한다. LiDAR 센서를 이용하여 생성한 포인트 클라우드는 일반적으로 3차원 공간의 값(x,y,z)를 가지게 되며, LiDAR 센서의 각분해능, field of view (FoV), 채널 수에 의해 생성된 방대한 포인트 데이터를 외부로 송신하게 된다. LiDAR 센서로부터 출력되는 방대한 데이터 및 타 센서들의 데이터를 동시에 하나의 메인 프로세서에서 처리하는 것은 쉬운 일이 아니다. 본 논문에서는 이러한 메인 프로세서의 부담을 줄이고자 다수의 LiDAR 센서의 데이터를 동시에 처리하기 위한 실시간성을 가지는 통합처리 시스템을 개발하고자 한다 [6,7].

Ⅱ. 관련 연구

LiDAR 센서는 LASER를 송신하는 발광부와 레이저를 특정 각도로 쏘기 위해 회전 거울을 회전시키기 위한 구동부 및 송신된 LASER가 물체에 맞고 반사되어 돌아올 때 이를 수신하고 검출하기 위한 수신부로 구성되어 있다. 이러한 구성의 LiDAR 센서를 scanning LiDAR 센서라 한다. 구동부를 통해 송신되는 레이저의 방향을 바꿈으로서 센서가 측정할 수 있는 범위, FoV를 가지게 되며, 수신부에서 단일 수광부를 가진 검출기가 아닌 다수의 수광부를 가지는 다채널 검출기를 사용하면 수직 축에 대한 거리 값 또한 얻을 수 있다. LiDAR 센서는 특정 포인트의 거리를 측정하기 때문에 이를 모으게 되면 그림 1과 같은 포인트 클라우드라는 군집 데이터를 이루게 된다. 이를 통해 물체까지의 거리뿐 아니라 수직 및 수평 FoV 내에서 물체의 폭과 높이를 같이 알 수 있다는 장점을 가진다.

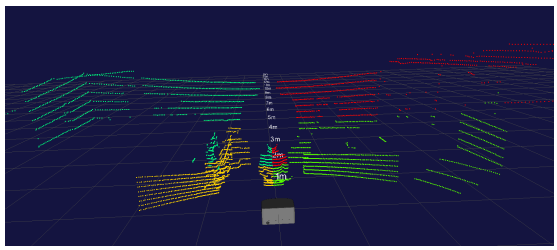


Fig. 1 Point cloud based on LiDAR sensor

본 논문에서 사용하는 LiDAR 센서는 ㈜카네비컴에서 개발한 VL-AS16 scanning LiDAR 센서를 사용하였다. VL-AS16은 16채널과 145°의 FoV를 가지며, 그에 따라 1 프레임에 18,560 포인트에 대한 정보를 송신하며, 30Hz의 구동 속도를 가지는 LiDAR 센서이다.

Ⅲ. 제안하는 시스템 구조

3.1. 통합 처리 시스템

본 논문에서 제안하는 통합처리 시스템은 최대 4개의 LiDAR 센서와 이더넷 기반의 유선 통신을 통해 데이터를 받게 되며, 이를 시스템 상에서 취합 및 처리하여 외부 서버나 시스템에 제공하게 된다. 단일 채널을 가지

거나 채널 수가 적은 LiDAR 센서의 데이터를 실시간으로 처리하는데 비해 다채널의 LiDAR 센서 여러 대의 데이터를 동시에 처리하기 위해서는 일정 스펙 이상의 프로세서가 요구된다. 이를 위해 NXP社 LS1028A를 사용한다 [8]. LS1028A는 2개의 멀티코어와 1개의 GPU 코어를 가지는 고성능 프로세서이다.

최대 4개의 LiDAR 센서와의 통신을 위해 통합처리 시스템은 4개 이상의 이더넷 포트와 하나 이상의 무선 통신 모듈을 갖도록 구성하였다. 그림 2는 시스템의 하드웨어 구성도이다. 시스템은 총 5개의 이더넷 포트와 1개의 무선 통신 모듈인 Wi-Fi 모듈을 탑재하도록 구성하였다. 5개의 이더넷 포트 중 4개의 포트는 LiDAR 센서와 통신을 수행하게 되며, 1개의 이더넷 포트와 무선 통신 모듈은 외부 서버나 시스템에 데이터를 전송해주기 위해 사용된다.

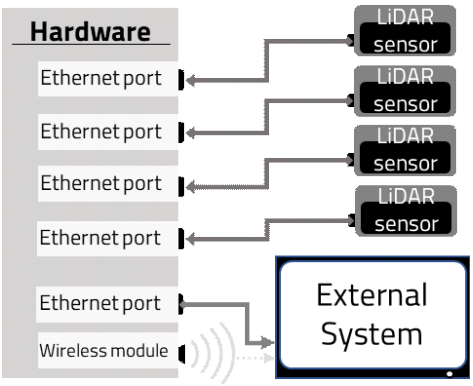


Fig. 2 System hardware configuration

본 논문에서 사용한 LiDAR 센서인 VL-AS16은 16개의 채널과 145˚의 FoV를 가지는 다채널 LiDAR 센서이다. 수평 각 분해능 0.125˚와 FoV 145˚에 의해 FoV 내에서 한 채널이 가지는 포인트의 개수는 1160개이다. 1160개의 포인트, 16개의 채널에 따라 VL-AS16은 매 프레임마다 55,680바이트 이상의 데이터를 출력하게 되며, 센서의 대수가 증가하면 처리해야할 데이터의 수 또한 비례적으로 증가하게 된다. VL-AS16은 30Hz로 구동하므로, 약 33 ms 마다 데이터를 송신하므로 통합처리 시스템은 각 LiDAR 센서의 데이터를 33 ms 내에 처리하여 외부 시스템으로 전송해주어야 하는데 이는 데이터 입력부터 처리 및 외부 시스템이나 서버로의 전송까지 걸리는 총 시간을 의미한다.

3.2. 시스템 구현

VL-AS16 LiDAR 센서는 UDP 통신을 기반으로 데이터를 전송한다. UDP 포트 및 IP를 기반으로 연결을 시도하고 센서와의 연결에 성공하면 센서는 자동적으로 수집한 데이터를 외부로 전송하게 된다. LiDAR 센서와 통신을 수행함에 따라 시스템은 다음과 같은 순서대로 동작하게 된다.

첫 번째, 데이터를 수집하는 과정이다. 연결된 LiDAR 센서의 숫자만큼 시스템은 순차적으로 이더넷 포트를 통해 데이터를 입력 받아 이를 수집한다. 두 번째로 데이터를 파싱하는 과정이다. 수집한 데이터를 프로토콜에 따라 일차적으로 분류하게 되며, 이를 시스템 설정에 따라 LiDAR 센서의 raw 데이터를 파싱하고 이를 raw 데이터를 거리 값으로 변경하거나 3차원 포인트 값으로 변경하는 과정을 수행한다. VL-AS16은 거리 값뿐만이 아니라 수신된 LASER의 강도 값, 센서의 상태 및 자체적인 객체 인식 기능을 탑재함에 따라 객체 정보 등을 추가적으로 보내주는데 이를 설정에 따라 파싱할 필요가 있다. 세 번째로 파싱한 데이터를 시스템 프로토콜에 따라 정렬하는 과정이다. 마지막으로 정렬한 데이터를 외부로 전송하는 과정을 가지게 된다. 그림 3은 시스템 동작 순서를 나타내는 flowchart이다.

이를 하나의 프로세스에서 실시간으로 선형적으로 처리하기에는 한계가 있다. 센서로부터 전송되는 데이터를 잃지 않고 모두 처리하기 위해서는 통합처리 시스템은 약 33 ms 내에 연결된 센서의 수와 상관없이 프로세스 과정을 모두 처리할 필요가 있다. 이러한 데이터 처리량을 분산하기 위해 각 LiDAR 센서를 담당하는 프로세스와 이를 수집하여 외부로 전송하기 위한 별도의 프로세스를 제작한다. 각 센서와 통신하는 프로세스를 'client' 명명한다. Client는 LiDAR 센서와 UDP 통신 및 데이터 수집을 수행하며, 수집한 데이터를 일차적으로 프로토콜에 따라 정렬하게 된다. Client로부터 데이터를 취합하여 외부로 전송하는 프로세스를 'server'로 명명한다. Server는 그림 4와 같이 client와 inter processor communication (IPC)을 통해 데이터를 송/수신한다. Server는 취합한 데이터를 최종적으로 프로토콜에 맞게 정렬한 후 이를 외부 서버나 시스템에 UDP 통신 방법을 통해 전송하게 된다.

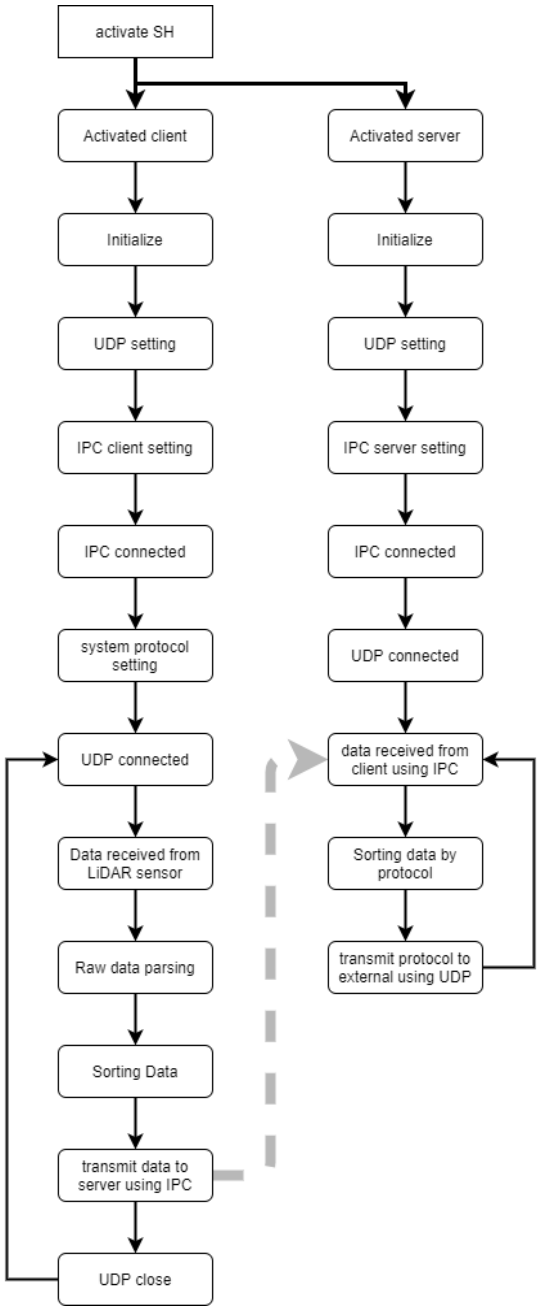


Fig. 3 System flowchart

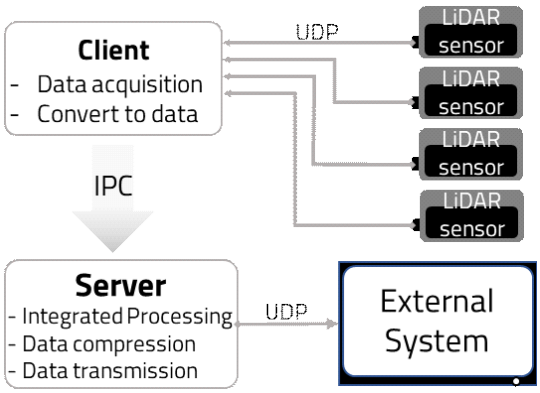


Fig. 4 System software configuration

Ⅳ. 실험 및 평가

4.1. 실험 절차

통합처리 시스템의 센서 연결 대수의 변화에 따른 시스템 응답시간을 측정하고 이를 통해 실시간성을 확인하였다. Client의 응답시간은 프로세스 초기화 이후, UDP 통신 대기 시간부터 데이터 파싱을 완료하고 이를 server로 전송하는 데까지 계측하였다. Server의 응답시간은 IPC를 통해 client로부터 데이터를 수신 받아 이를 외부로 전송하는 데까지 걸리는 시간을 계측하였다. Server는 연결된 모든 client로부터 데이터를 수신 받을 때까지 대기하고 수신이 완료되면 데이터를 처리하여 외부로 전송하도록 구현하였다.

각 프로세스를 구현하고 동작 방법에 따른 시스템의 응답시간을 확인한다. 첫 번째로 하나의 client가 다수의 센서와 통신하고 이를 선형적으로 처리하도록 구현하고 이에 대한 시스템 응답 시간을 측정한다. 두 번째로 연결되는 LiDAR 센서의 수만큼 client를 구동하도록 구현하고 server는 각 client로 데이터를 수신 받아 처리하도록 구현하였다. 이를 shell script 및 '&' 명령어를 통해 리눅스의 background에서 구동하도록 처리한다. 세 번째는 두 번째와 동일한 client 구성을 가지는데 client 동작 시 'taskset' 명령어를 이용하여 각 client 및 server를 특정 코어에 할당하여 구동시키는 multi-core processing을 진행한다 [9, 10]. 각 과정을 통해 시스템의 응답 시간을 확인하고 시스템 최적화를 진행하였다.

4.2. 실험방법 1. Linear processing

그림 5의 그래프는 linear processing의 시스템 응답시간을 측정한 결과이다. LiDAR 센서의 연결 수가 증가할수록 client의 응답시간 또한 약 19.7 ms에서 75.4 ms까지 증가하는 것을 확인할 수 있었다. 서버의 응답시간은 약 3.3 ms에서 12.4 ms까지 증가하는 것을 확인할 수 있었다.

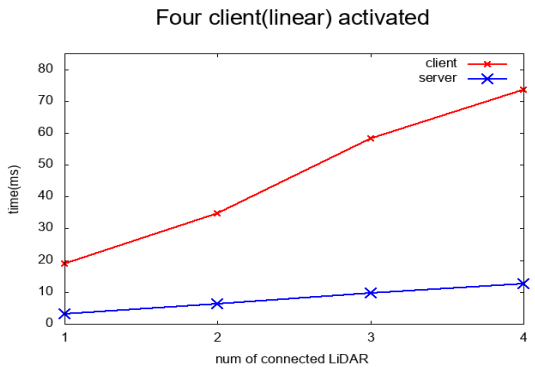


Fig. 5 System's linear processing response time

4.3. 실험방법 2. Background processing

Background 및 multi-core processing을 위해 하나의 client는 하나의 LiDAR 센서와 통신하도록 만들고 server는 그림 6과 같이 다중 client와 IPC를 기반으로 UDP 통신하도록 구성하였다. Background processing은 linux 계열 OS에서 지원하는 '&' 명령어를 통해 동작하게 된다. Background processing은 프로세스 실행 시, '&' 명령어를 붙여서 실행하게 되면 해당 프로세스는 background 상에서 동작하게 되며 그림 16.(a)와 같이 OS의 스케줄링에 의해 자동으로 코어에 할당되어 동작하게 된다.

그림 7, 8, 9, 10은 background processing을 통해 LiDAR 센서 연결 대수를 증가시켜가면서 시스템의 응답시간을 측정한 결과이다. 연결된 LiDAR 센서가 한 대일 경우 client 응답시간은 약 18 ms이며, server의 응답시간은 약 3 ms였으며, 두 대의 LiDAR 센서를 연결했을 때, 각 client와 server의 응답시간은 각각 약 14.8 ms와 6.6 ms로 계측되었다. 센서의 연결 대수가 4대까지 증가하였을 때, 각 client 및 server는 약 13.2 ms와 12.6 ms의 응답시간을 확인할 수 있었다. 연결된 LiDAR 센서의 개수가 증가할수록 client의 응답시간이 약 13 ms로 수렴되었고 server의 응답시간은 3 ms에서 12.6 ms로 증가하였다.

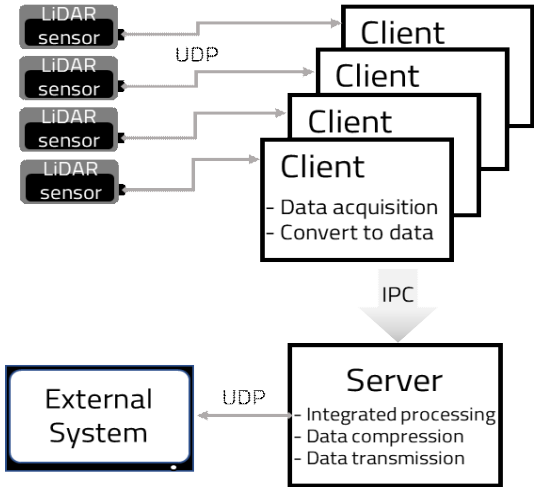


Fig. 6 Multi process configuration

4.4. 실험방법 3. Multi-core processing

Multi-core processing 구동을 위해서 Linux 계열 OS에서 지원하는 'taskset' 명령어를 사용하여 프로세스를 동작시켰다. 'taskset'은 프로세스 동작 시, 해당 프로세스를 특정 코어에서 동작시키도록 지정할 수 있다. 본 논문에서는 그림 16.(b)와 같이 3개의 client는 LS1028A의 0x01번 코어에, 나머지 하나의 client와 server는 0x02번 코어에 할당하였다. 그림 11, 12, 13, 14는 multi-core processing한 시스템의 응답 시간을 나타낸다. Multi-core processing 시, 통합처리 시스템의 응답시간은 이전 실험인 background processing과 유사하게 LiDAR 센서의 연결 대수가 4대일 때 각 client의 응답 시간은 약 13.16 ms, server의 응답 시간은 약 12.5 ms를 보였다.

Fig. 7 System background processing when one LiDAR sensor connected

Fig. 8 System background processing when two LiDAR sensor connected

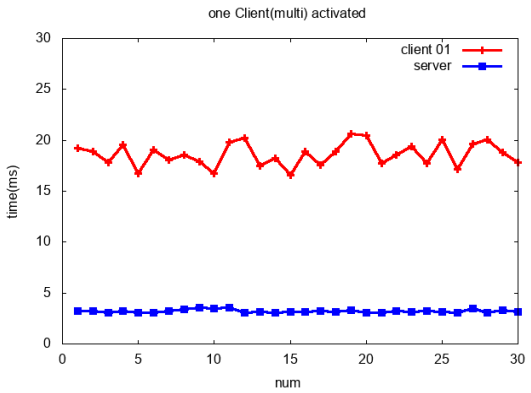


Fig. 11 System multi-core processing when one LiDAR sensor connected

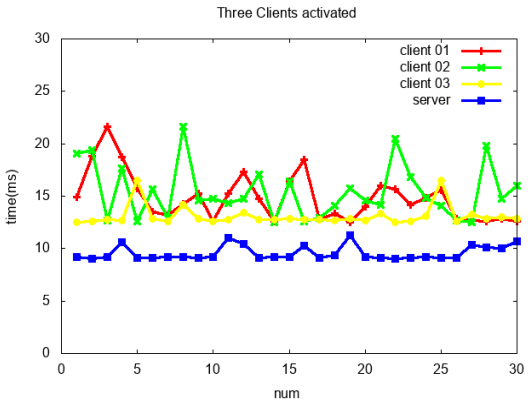


Fig. 9 System background processing when three LiDAR sensor connected

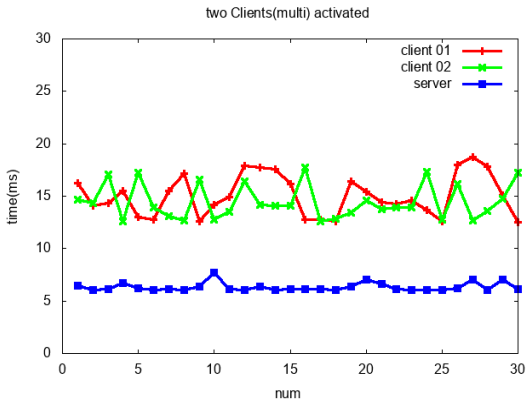


Fig. 12 System multi-core processing when two LiDAR sensor connected

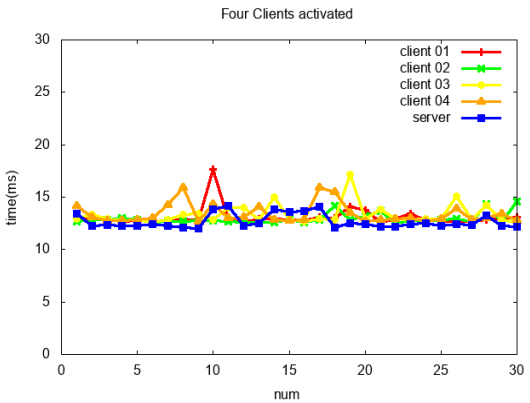


Fig. 10 System background processing when four LiDAR sensor connected

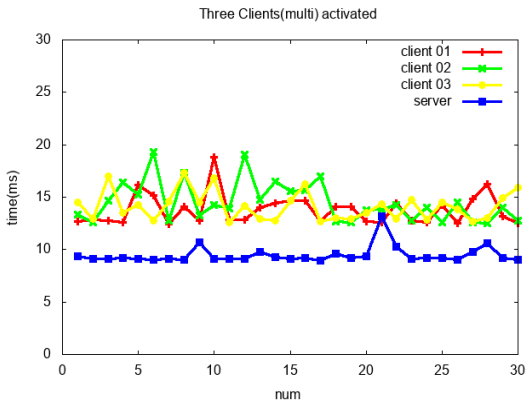


Fig. 13 System multi-core processing when three LiDAR sensor connected

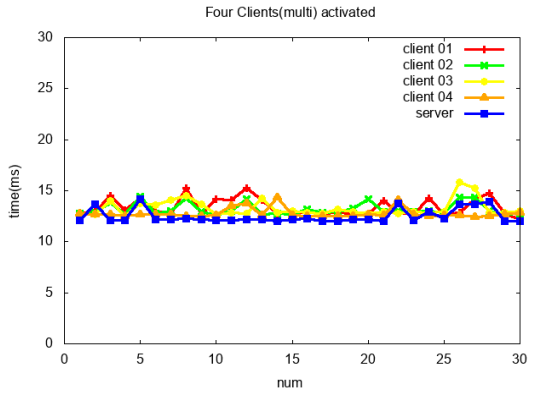


Fig. 14 System multi-core processing when four LiDAR sensor connected

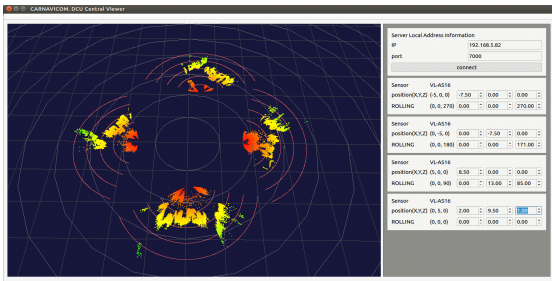
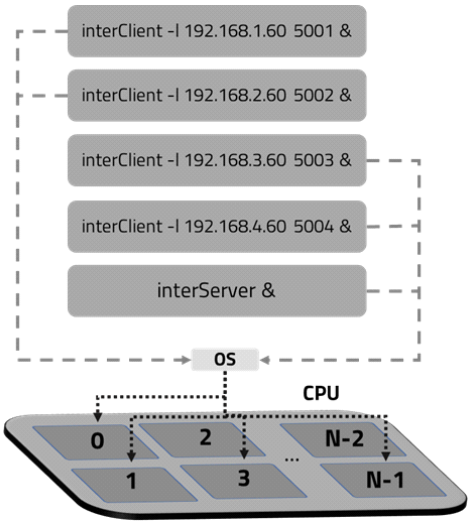
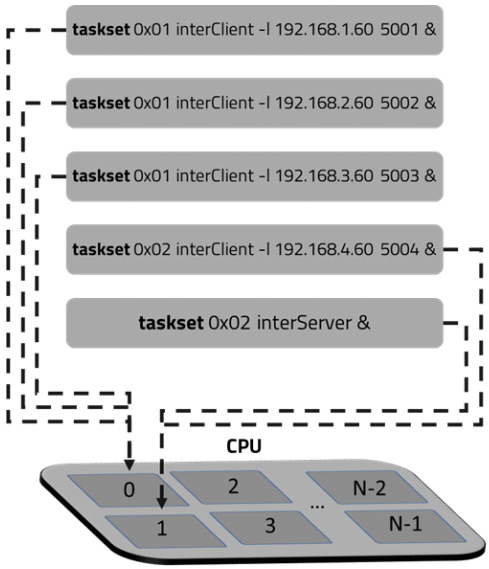


Fig. 15 Data Visualization from server



(a) Background processing using command '&'



(b) Multi-core processing using command 'taskset'

Fig. 16 Control of multi process using command

4.5. 실험 결과

Linear processing 시, 4개의 LiDAR 센서를 연결한 경우 client의 응답시간이 약 75.4 ms까지 증가하는 결과를 보였으며, server의 응답 시간은 12.4 ms까지 증가하는 결과를 확인할 수 있었다. UDP 통신을 통한 데이터의 입력부터 데이터 처리 등을 선형적으로 처리하는 만큼 센서 연결 개수가 증가함에 따라 client 및 server의 응답시간이 선형적으로 증가하는 것을 확인할 수 있었다. 서버의 응답시간은 최대 4대의 LiDAR 센서의 데이터를 처리하더라도 33 ms 내의 응답 시간을 보였지만, client의 경우 LiDAR 센서 연결 수가 증가할수록 UDP 수신 대기 시간의 증가, 데이터 파싱 횟수 증가 등에 의해 33 ms를 만족하지 못하는 것을 확인할 수 있었다. Background processing 및 4개의 LiDAR 센서 연결 시, 각 client의 응답 시간은 약 13.2 ms, server의 응답 시간은 약 12.6ms의 처리 속도를 보였다. Multi-core processing 동작시켰을 경우 각 client는 13.16 ms, server는 12.5 ms의 background processing과 유사한 응답 시간을 보였다. LiDAR 연결 대수가 증가할수록 client의 응답시간은 줄어들어 약 13 ms에 수렴하는 현상을 보였는데 이는 background 및 multi-core processing을 적용함에 따라 다수의 프로세스 동작에 따른 각 client의 대기 시간

이 줄어들어 client의 응답시간이 줄어드는 것으로 판단된다.

실험 결과를 바탕으로 background 및 multi-core processing 시, 통합처리 시스템은 LiDAR 센서의 연결 대수가 증가하더라도 데이터 전송 속도인 33 ms 내에 데이터를 받아 이를 처리하고 외부로 전송하는 것을 확인할 수 있었다. 그림 15는 통합처리 시스템으로부터 전송된 데이터를 최종적으로 시각화한 결과이며, 센서를 구분하기 위해 각 LiDAR 센서의 출력 위치를 달리하여 시각화하였다 [11].

Ⅴ. 결 론

본 논문에서는 최대 4대의 LiDAR 센서로부터 데이터를 수신 받고 이를 처리하여 외부 서버나 시스템으로 전송하는 임베디드 시스템을 개발하였다. LiDAR 센서별로 데이터를 처리하고 이를 통합하기 위해 각 프로세스, client와 server를 제어하기 위해 background processing 및 multi-core processing을 도입하여 각 프로세스의 응답 시간을 측정하고 최적화를 진행하였다.

Background와 multi-core processing은 LiDAR 센서의 데이터 전송 속도 33ms 내의 유사한 응답 시간을 보였는데, 이는 현재 시스템이 별도의 복잡한 연산 과정을 실행하지 않고 LiDAR 센서의 raw 데이터를 거리 값으로 변환하는 과정만을 갖기 때문으로 파악된다. 하지만, LiDAR 센서 외의 카메라, RADAR 센서와 같은 다른 센서가 추가되거나 타사 LiDAR 센서와 통합 처리, 보다 많은 데이터 처리 및 딥러닝을 기반으로 하는 객체 인식 등과 같은 많은 계산량을 요하는 프로세스와 동시에 동작할 경우 시스템이 33 ms 이내의 응답 시간을 가지기 위한 최적화 과정이 필요할 것이며, 이 때, multi-core processing이 유용하게 사용될 것으로 예상된다.

본 논문의 연구를 기반으로 타 센서 동시 처리, 포인트 클라우드 기반 객체 인식, 전송 데이터 량 감축 방안, 병렬 프로세스 등을 이용한 응답 시간 개선과 같은 연구를 지속해갈 예정이다.

ACKNOWLEDGEMENT

This work was supported by the Technology Innovation Program (P0013847, 10%, Development of automatic steering-based accident avoidance system for electric-driven port yard tractors operating at low speed (less than 30 km/h)) funded By the Ministry of Trade, Industry & Energy(MOTIE, Korea) and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (NRF-2019 R1A2C2005099, 10%), Ministry of Education (NRF-2018 R1A6A1A03025109, 10%), and partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-00944, Metamorphic approach of unstructured validation/verification for analyzing binary code, 70%)

REFERENCES

[1] J. Giacalone, L. Bourgeois, and A. Ancora, “Challenges in aggregation of heterogeneous sensors for Autonomous Driving Systems,” *IEEE Sensors Applications Symposium (SAS)*, pp. 1-5, 2019. doi: 10.1109/SAS.2019.8706005.

[2] L. Xiaoming, Q Tian, C Wanchun, and Y. Xingliang, “Real-time distance measurement using a modified camera,” *IEEE Sensors Applications Symposium (SAS)*, pp. 54-58, 2010. doi: 10.1109/SAS.2010.5439423.

[3] L. Zhaohua and G. Bochao, “Radar Sensors in Automatic Driving Cars,” *5th International Conference on Electromechanical Control Technology and Transportation (ICECTT)*, pp. 239-242, 2020. doi: 10.1109/ICECTT50890.2020.00061.

[4] A. U. Kulkarni, A. M. Potdar, S. Hegde, and V. P. Baligar, “RADAR based Object Detector using Ultrasonic Sensor,” *1st International Conference on Advances in Information Technology (ICAIT)*, pp. 204-209, 2019. doi: 10.1109/ICAIT47043.2019.8987259.

[5] S. Lee, D. Lee, P. Choi, and D. Park, “Accuracy-Power Controllable LiDAR Sensor System with 3d Object Recognition for Autonomous Vehicle,” *Sensors*, vol. 20, no. 19, 2020.

[6] R. O. Chavez-Garcia and O. Aycard, “Multiple Sensor Fusion and Classification for Moving Object Detection and

Tracking," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 525-534, Feb. 2016. doi:10.1109/TITS.2015.2479925.

[7] V. Venugopal and S. Kannan, "Accelerating Real-Time LiDAR Data Processing using GPUs," *IEEE 56th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1168-1171, 2013. doi: 10.1109/MWSCAS.2013.6674861.

[8] NXP. Layerscape LS1028A Family of Industrial Applications Processors [Internet]. Available: https://www.nxp.com/docs/en/fact-sheet/ls1028afs.pdf.

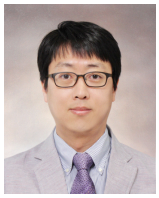
[9] J. Dybedal, A. Aalerud, and G. Hovland, "Embedded Processing and Compression of 3d Sensor Data for Large Scale Industrial Environments," *Sensors*, vol. 19, no. 3, 2019.

[10] K. M. Lee, T. H. Song, S. H. Yoon, K. H. Kwon, and J. W. Jeon, "OpenMP Parallel Programming using Dual-Core Embedded System," *11th International Conference on Control, Automation and Systems*, pp. 762-766, 2011.

[11] S. Zhou, M. Xie, Y. Jin, F. Miao, and C. Ding, "An End-to-end Multi-task Object Detection Using Embedded GPU in Autonomous Driving," *22nd International Symposium on Quality Electronic Design (ISQED)*, pp. 122-128, 2021. doi: 10.1109/ISQED51717.2021.9424308.

정태원(Taewon Chong)

2016년: 한국산업기술대학교 전자공학부 학사
2018년: 전남대학교 전자컴퓨터공학부 석사
2018년~현재: 한양대학교 물리학과 박사과정
2015년~현재: ㈜카네비컴
※관심분야 : LiDAR 센서, 저전력 임베디드 시스템, 병렬 프로세싱, 객체 인식



박대진(Daejin Park)

2001년: 경북대학교 전자전기공학부 학사
2003년: KAIST 전기 및 전자공학과 석사
2014년: KAIST 전기 및 전자공학과 박사
2003년~2014년: SK Hynix/ Samsung (차세대 LSI 설계) 수석연구원
2014년~2016년: 경북대학교 전자공학부 연구조교수 (2014년 대통령 Postdoctoral Fellow 선정)
2016년~현재: 경북대학교 전자공학부 조교수
※관심분야 : 저전력 SoC 설계, 하드웨어-소프트웨어 Co-design, Dependable 스마트 IoT 시스템, Robust 임베디드 시스템