GPS 를 이용한 차량 자율주행 보조 및 제어

*전한결, 박범석
*강남대학교 전자공학과
han940922@gmail.com, qkr4qja5tjr6@gmail.com

A Study of making connected car using GPS

*Beom Seok Park, Han Gyoel Jeon
*Dept. of IoT Electrical Engineering ,Kangnam University

요 약

현재 자율주행은 Stand Alone 으로 차량 자체적인 센서만을 이용해서 자율주행을 하고 있다. 이번 연구로 Stand Alone 하지 않은 서버 제어 방식의 자율주행 차량 개발이 가능해져 자율주행 차량의 센서와 연산장치가 줄어 차량의 가격이 줄어들며 이 시스템의 궁극적 목표인 자율주행 차량의 사고 감소에 큰 도움을 줄 것을 기대한다.

1. 서론

현재 자율주행은 Stand Alone 으로 차량 자체적인 센서만을 이용해서 자율주행을 하고 있다. 그렇기에 기존의 차량은 자율주행의 혜택을 전혀 보지 못하고 있으며, 자율주행 차량 역시 주변에 자신을 전혀 알리지 못하고 있다. 그렇기에 이런 자율주행 차량들의 독립화를 해결하기 위해 Connected Car 의 개발이 절실하다. 이를 위해 여러 기술들이 연구 되고 있으나 아직 제대로 된 결과물이 대중에게 공표되지 않고 있다.

최근 Connected Car 를 위한 기반 연구가 계속 되고 있으나, 모바일 네트워크의 지연성과 Connected Car 자체의 보안 문제로 인한 해킹[1], 그로인한 사고 가능성으로 아직 연구소에 머물러 있는 형편이다. IoT 를 위한 제대로 된 저지연성을 가진 5G 인 mmWave 는 기술 표준과 실안은 마련되었으나 코로나 및 비용문제로 인해 실용화는 미진하다. 보안 문제 및 악의적인 사용으로 인한 사보타주의 경우에도 아직 제대로 된 방안이 없는 형편이다.

이에 본 논문에서는 GPS 를 이용한 Connected Car 의 구현 및 실증을 할 것이다. 통신으로는 LTE Cat M.1 을 이용하며, Field Test 용 RC 카를 조정하는 제어 시스템은 라즈베리 파이, 위치를 파악은 LTE 모듈에 내장된 GPS 를 이용한다.

2. 관련 연구

I. 차량의 GPS 정보 수신

라즈베리파이에서 Python 을 이용한 AT 커맨드

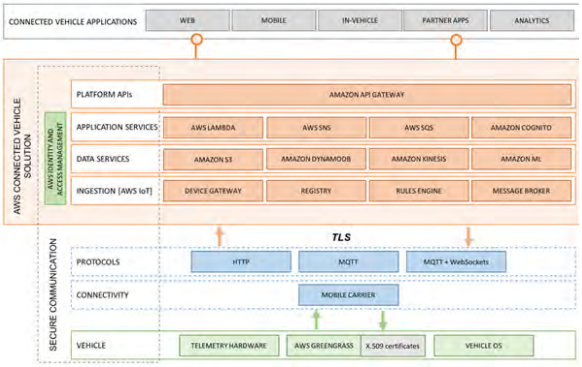
명령으로 LTE&GPS 모듈에 접속해, GPS 수신을 한다. 이 정보는 모듈의 스펙인 100ms 마다 갱신되며, 이 정보는 차량 간의 위치를 파악할 때 사용된다. A-GPS 를 지원하지 않으며, 이로 인해 GPS 기술의 최소요건인 안테나가 열린 하늘을 보고 있는 것이 강제되는 단점이 있다.

II. LTE Cat.M1 을 이용한 서버 송수신

LTE 와 GPS 가 통합된 칩셋으로 라즈베리파이의 제어를 받아 위치정보를 수신한 다음 그 데이터를 LTE Cat.M1 망을 이용해 GPS 트래킹 서버로 정보를 전송하게 된다.

III. AWS GPS Tracking Sevice

라즈베리파이에서 보내는 위치 정보를 수신한 후 각 차량의 위치를 식별하고 GPS 위치가 위험수위일 경우 각 차량으로 경고 및 STOP 제어 신호를 보낸다. DB 를 이용해 각 차량 간의 위치를 등록하며, DB 는 5 분간 저장되며 그 뒤에는 위치정보가 삭제된다. GPS 의 에러를 예방하기 위해 도로 이외의 정보가 나오면 무시한다.



그림(1) AWS GPS Tracking Sevice 의 구조도

IV. 라즈베리파이의 Web 기반 차량제어

Python 의 smbus 를 이용한 모터드라이브 제어와 jsQuery 및 HTML 을 클라이언트로 사용한 제어 시스템으로 서버에서 명령이 내려오면 이를 차량의 모터드라이브에 명령해 차를 멈추거나 LED, 스피커, 센서 등으로 주변 차량에 위급함을 알린다.

3. 설계 및 구현

I. 시스템 구조도

그림 2 는 이번 연구의 시스템 구조도이다.

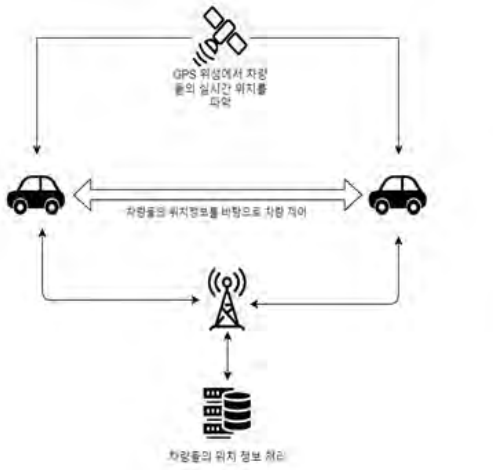
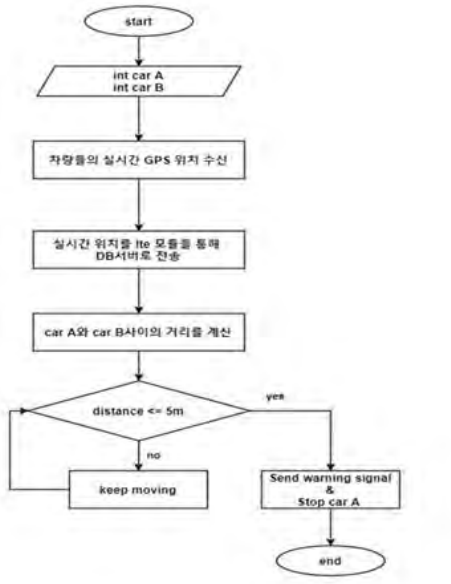


그림 (2) 시스템 구조도

각각의 차량에서 GPS 정보를 수신 받아 LTE CATM.1 을 이용해서 서버로 송신한다.
서버에서 각각의 차량의 위치를 수신 받으며 차량 간의 위치가 가까워질 경우 차량에게 경고를 보낸다. 차량 간의 위치가 부딪칠 위기일 시 차량을 강제로 멈춘다.

II. 알고리즘 Flow Chart

그림 3 은 이번 연구의 Flow Chart 이다.

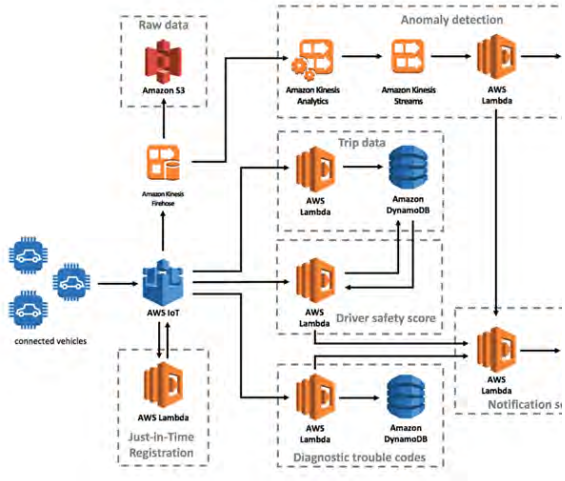


그림(3) Flow Chart

자동차와 자동차 사이의 GPS 로 측정된 거리를 계산하여 적절한 차량간격을 유지한다. LTE Cat.M1 을 이용하여 탐색된 위치 정보를 AWS 서버로 전송한다. 차량의 거리 계산은 람다서버에서 코드를 자동으로 트리거하도록 설정한다. 차와의 간격이 5M 이내로 줄어들면 차를 정거시킨다.

III. 서버 설계

그림 4 는 AWS GPS Tracking Sever 를 이용한 본 논문의 Sever Block Diagram 이다.

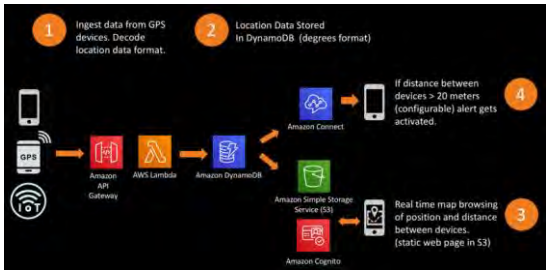


그림(4) Sever Block Diagram

서버의 Data Flow 를 위한 서버 구조에 대한 연구이다. 앞서 말한 AWS GPS Tracking service 는 거리 계산 및 위치 확인에만 쓰이게 된다. 서버는 amazon API gateway 를 이용한 송수신 클라이언트를 가지며 서버 클라이언트의 통신 규격은 HTTP + Web Socket 이다. DB 와 서버 관제 및 연산은 Amazon DynamoDB 와 AWS Lambda 을 사용하며, AWS Lambda 는 서버 내부 DB 의 데이터를 AWS GPS Tracking service 에게 넘겨 차량간의 거리 연산을 통해 DB 를 통한 차량간의 거리 판별을 가능하게 한다. Amazon S3 를 이용해 서버 상황 확인 및 관리를 할 수 있다.

IV. 구현

그림 5 는 본 논문의 총괄적 Block Diagram 이다.



그림(5) Entire Block Diagram

클라이언트인 Connected Car 는 Python 의 smbus 를 이용한 제어 시스템을 가지며 jsQuery 와 HTML 을 이용한 클라이언트를 가지고 있다. 이 클라이언트는 서버에 GPS 데이터를 송신한다.

서버에서는 Amazon API Gateway 로 데이터를 수신 받아 AWS Lambda 를 거쳐 DynamoDB 에 각 차량을 Tagging 하고 저장한다. 이 정보로 차량의 근접 및 충돌 유무를 파악해 위기 상황 시 경고 및 강제 제어 코드를 Amazon Cognito 를 이용해 클라이언트의 WebSocket 으로 제어 신호를

전달한다.

4. 결론

GPS 를 이용한 위치 정보 탐색 기술을 효율적으로 사용하기 위한 연구가 추가적으로 진행되어야 할 것이며, 서버의 위치정보 취합 및 판단이 프로젝트의 성공을 판가름할 것이다. LTE Cat.M1 모듈은 통신 속도가 기대 이하의 specification 을 제공하고 있기 때문에 개발 알고리즘을 보완하거나 통신 속도면에서 더 개선된 새로운 알고리즘을 찾는 방향으로 연구를 진행할 예정이다. 그리고 현 GPS 모듈이 A-GPS 를 지원하지 않아 GPS 의 자체적인 부팅 및 스캔 속도가 느리고 천장이 뚫려 있지 않은 경우 GPS 가 잡히지 않는 문제가 있어 GPS 모듈의 개선이 필요하다. 추가적으로 스마트 도로에서의 self-driving car assistant 와 연동하여[2], connected car 의 발전을 기대하고 있다.

참고문헌

'본 논문은 과학기술정보통신부 정보통신창의인재양성사업의 지원을 통해 수행한 ICT 멘토링 프로젝트의 결과물입니다.'

[1] Simon Weckert, "Google Maps Hacks ". Performance & Installation, 2020

[2] 심창수, 이상윤, 심성한 "Smart Construction Technologies for Roadway Structures"