프로젝트 이름

이도혁(16)*, 김인엽(18), 이용민(17), 이도균(18)

*dohyeoklee16@kaist.ac.kr

Abstract

자율주행 배달로봇 프로토타입을 만들었다.

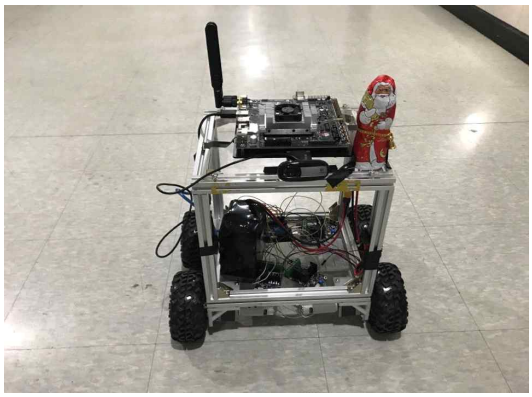
1. Introduction

동기: 배달 받기가 너무 귀찮아서.

필요성: 필요성은 주관적이기에 모르겠지만, 잠재 수요를 생각한다면 상당한 수준이라고 본다.

초기 목표: 진짜 배달시키기.

2. Result



진짜 배달은 파워 문제로 안 되지만,(젯슨에게 220V를 공급하려면 좀 더 로봇을 키워야 한다) 좀 scale-up해서 파워 문제를 해결하면 가능할 수 있다. LIDAR를 제외하고, GPS, compass 센서, 네비게이션, object detection 등 필요한 기능은 모두 구현했다.

3. Challenges

1) 구동 시스템

구동 시스템은 처음엔 스텝모터+일반 휠로 했는데, 무게는 꽤 나가는데 토크가 안 나온다. 제발 DC모터 쓰자. 또한 모터 브라켓은 3D 프린팅으로 제작했는데, 열공차때문에 잘 안 된다. 프로토타입이 목표였으니 일단은 계속 뽑아서 썼는데, 생산을 생각한다면 철판에 구멍 뚫는 게 낫겠다.

2) GPS, compass 센서

야식이의 현재 위치를 실시간으로 알아내기 위해서 GPS를 사용하게 되었다. GPS 모듈은 NEO-6M 아두이노 모듈을 사용했다. 우선 GPS 센서에서 받은 값을 그대로 컴퓨터로 화면을 출력하는 코드를 짰다. 그 결과 현재 위치, 속도, 날짜와 시간 등 여러 데이터가 한꺼번에 출력되었다. 이 중에서 필요한 정보만을 찾는 과정, 즉 '파싱' 작업이 필요하였다. 현재 위도와 경도 데이터만을 뽑아내기 위해서 GPGGA(Global Positioning System Fix Data)를 찾아서 그 줄에 있는 데이터를 분석해야했다. 처음에는 TinyGPS 라이브러리를 이용하여 해결하려 했지만 encode 함수가 아예 작동하지 않았다. 이에 TinyGPS++ 라이브러리 등도 이용해보았지만 결과는 같았다. 아두이노 보드와 GPS 모듈에 문제가 있었을 가능성을 생각해서 교체해보았지만 역시 encode 함수가 작동하지 않았다. 결국 라이브러리를 이용하지 않고 직접 파싱을 하는 코드를 짜게 되었다. 매우 간단한 작업이었다. 위도와 경도가 그 줄 몇 번째부터 나타나있는지 확인한 후 숫자들을 계산하여 구하면 되었다. 또 만약 GPS 센서

가 작동은 하지만 실내에 있어서 신호가 잘 잡히지 않을 때의 예외처리도 해야했다. 원래 위도와 경도가 있어야 할 곳에 다른 데이터가 들어가 전혀 다른 값이 나올 수도 있기 때문이다. 이는 위도, 경도 값이 크게 달라질 일은 없기 때문에 백의 자리와 십의자리가 맞는 값일때만 데이터를 넘겨주는 방식으로 해결했다. 이러한 과정을 통해서 오차 없이 정확한 GPS 값을 얻을 수 있었다.

compass 센서는 크게 어려운 점 없었다.

3) Tmap API

메소드를 익히는 것 제외하고 크게 어려운 것 없었다.

4) object detection

(1) 젯슨 설치에서 YOLO 설치까지

0. USB 포트가 하나다. 허브는 미리 구매할 것.
1. host 컴퓨터는 라우터를 통해서 인터넷에 연결이 되어있어야한다.
2. 젯슨도 시작전에 랜선을 라우터에 꼽아두자.
3. 꼽아둬도 'Determing IP address..' 이런 오류가 뜨는데, 당황하지 말고 화면을 끄자.
4. https://devtalk.nvidia.com/default/topic/1002081/jetson-tx2/jetpack-3-0-3-1-install-with-a-vm/
5. 위 그대로 한다.
6. Jetpack이 모두 설치가 된다. 중요한 cuda가 설치가 완료된다
7. cuda 확인법: nvcc -V
8. host PC는 가상머신도 괜찮다만..
9. 가상머신의 용량은 40G쯤 만들어두는게 좋다
10. 가상머신 + 젯슨 모두 설치가 완료되었는가?
11. 그럼 OpenCV 설치로 넘어가자.
12. 만약 yolo를 쓰고 싶다면, OpenCV 3.4.1이 아닌 3.4.0이 필요하다.

13-1: 3.4.1: https://shiroku.net/robotics/install-opencv-on-jetson-tx2/

13-2: 3.4.0: https://jkjung-avt.github.io/opencv3-on-tx2/

14. yolo 설치: https://jkjung-avt.github.io/yolov3/
15. 하지만 젯슨은 yolo(4GB GPU RAM)를 다 돌리기엔 버겁다. tiny yolo(1GB GPU RAM)를 사용하자.

(2) YOLO 사용

카이스트를 지나다니며 찍은 사진들 천수백장 모두 bounding box를 쳐줬는데, 잘 안되고 힘드니까 가능하면 이미 있는 데이터셋 사용하고 그 중 필요한 레이블만 뽑아 쓰자.

(3) 그 다음..

YOLO의 작동에 뭔가 변화를 주고 싶다면 보통 src 폴더의 image.c를 건드리면 된다. 인식률 별로긴 한데 쓸 만하다.

4. Further Works

일단 LIDAR를 달아서 완벽하게 다 피해야한다. 현재 상태는 image input으로 회피를 결정하기 때문에, depth 정보가 없고 정면만 피할 수 있다.

젯슨에게 파워를 공급해서 야외에서도 주행이 가능하게 해야한다. 지금 젯슨은 너무 크고 파워도 많이 잡아먹는다.