휴먼 트래킹을 위한 자율 주행 자동차 제작 및 소프트웨어 구현

한성대학교 컴퓨터 공학과
2021-1 캡스톤 디자인

팀 명	황제펭귄			
담당 교수	황기태 교수님			
팀원	이름	학번	연락처	E-mail
	김채린(팀장)	1871071	01047352370	krin0525@naver.com
	유수미	1871384	01031548109	hugbee@naver.com
	이승현	1891080	01053399867	1891080@hansung.ac.kr
GitHub Link	https://github.com/HSEmperorPenguin/EmperorPenguin			

목차

1. 프로젝트 수행 목적

1.1 프로젝트 정의

wi-fi의 수신 신호 강도와 머신 러닝을 이용한 휴먼 트랙킹 로봇.
ROS를 이용해 다양한 센서와 디바이스들을 사용해 “Human Tracking 자율 주행 차량”를 구현했다.

1.2 프로젝트 배경

현재 구글의 웨이모와 테슬라, 애플까지 자율 주행에 대해 앞다투어 연구하고 있고, 우리나라 역시 2018년 과학기술정보통신부에서 4차 산업혁명 대응 우선 추진 분야로 자율 주행차를 선정하였으며 이어 2021년도에도 정부 연구 개발 투자 방향에 자율 주행 분야를 포함했다. 이에 저희는 각광 받는 자율 주행 기술을 도로에서 더 확장해 이를 실생활에 밀착시켜 인간의 편의를 증진시킬 방안을 고민했다. 그리하여 자율 주행 기능에 휴먼 트랙킹 기술을 결합하여 사람을 따라가는 로봇을 만들어 스마트 장바구니, 범죄자 감시 시스템, 노인 돌봄 로봇, 안심 귀가 로봇 등 다양하게 활용 가능하도록 구현하고자 계획했다.

1.3 프로젝트 목표

그림 1

RC카와 다양한 하드웨어의 통신을 위해 로봇 운영체제, ROS를 이용하여 Human Tracking 차량을 구현하고자 한다.

그림 2

본 프로젝트에는 네 가지 목표가 있다.
첫째, 사람을 인식한다. 둘째, 차량은 인식된 사람의 위치를 파악해 주행한다. 셋째, 차량과 사람 간의 거리를 파악해 일정 거리를 유지하며 주행할 수 있게 한다. 넷째, 사용자는 안드로이드 디바이스를 이용해 차량과 통신할 수 있다.

이 네 가지의 기능들로 구성된 Human Tracking 자율 주행 차량을 구현한다.

2. 프로젝트 결과물의 개요

2.1 프로젝트 구조

2,1,1 전체 구조

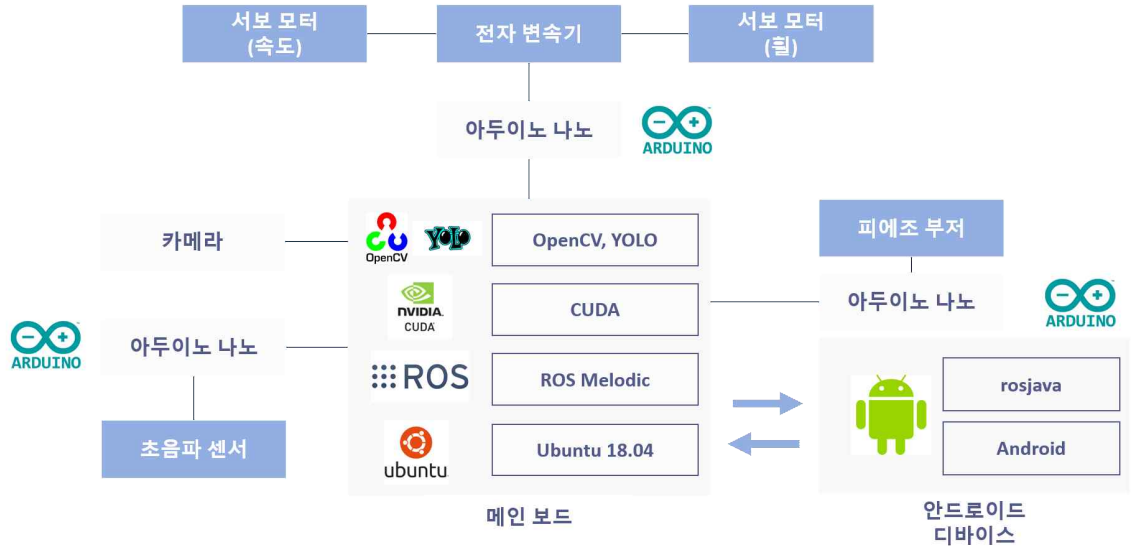


그림 3

메인 보드는 우분투 18.04를 운영 체제로 사용한다. 그리고 로봇 소프트웨어 플랫폼인 ROS Melodic을 미들웨어로 사용하며, GPU를 활용하기 위하여 NVidia CUDA를 탑재힌다. 또한, 물체 인식을 위하여 YOLO와 이미지 처리를 위한 OpenCV 라이브러리를 사용한다.

아두이노 나노에 전자 변속기가 연결되어 있으며 전자 변속기에 속도와 휠을 각각 제어하는 서보 모터 두 개가 연결된다. 또 다른 아두이노 나노에는 피에조 부저와 초음파 센서가 하나씩 연결된다. 그리고 마지막으로 USB 카메라가 연결된다.

모바일 기기는 안드로이드 운영체제가 탑재된 기기를 사용하며, 안드로이드에서도 ROS를 사용하기 위해 rosjava 라이브러리를 이용하여 메인 보드와 통신한다.

2,1,2 ROS(Robot Operating System)

로봇 애플리케이션 개발을 도와주는 소프트웨어 라이브러리와 툴의 집합으로, 표준화된 통신을 이용해 각 디바이스와 센서를 모듈화해 사용할 수 있는 미들웨어의 일종이다. 이때, 통신은 비동기 방식은 TCP/IP 방식을 사용한다.

즉, ROS는 하드웨어의 의존성이 적어서, 다양한 기종을 사용하는 로봇을 개발하는데 아주 유용하다. 본 프로젝트에서는 ROS Melodic를 이용했다. ROS Melodic은 Ubuntu 18.04 위에서 동작한다.

ROS의 동작은 MQTT와 유사하다. ROS는 노드(node) 단위로 동작하는데, 노드란 실행되는 최소 단위의 프로세서, 즉, 프로그램을 의미한다.

노드는 토픽(topic)이라 부르는 데이터를 마스터(master)에 전달하고, 마스터는 토픽을 필요로 하는 노드에게 전달한다. 이를 통해 노드는 다른 노드들과 통신한다. 본 프로젝트에서는 메인 보드(TX2)에 마스터를 두고, 나머지 노드들이 토픽을 주고받으며 Human Tracking과 Joystick 동작을 가능하게 한다.

또한, 노드는 subscriber와 publisher로 구분된다. subscriber는 토픽을 전송하는 노드이고, publisher는 토픽을 받는 노드이다. 노드는 subscriber인 동시에 publisher일 수 있다. 본 프로젝트에서(그림 “순서”에서) play_drive와 같은 노드가 그 예이다.

2.1.3 Nvidia CUDA

GPU 사용을 위한 것이다. 실시간 이미지를 받아 darknet_ros에서 물체를 인식한다. 이 때문에, 이미지 처리 속도 향상을 목적으로 사용했다.

2.1.4 YOLO

물체 인식을 위해 YOLO를 사용했는데, ROS와 함께 사용해야 하기 때문에 ROS 위에서 YOLO를 구동시킬 수 있는 darnet_ros 오픈 소스를 활용했다.

2.2 프로젝트 결과물

2.2.1. 개발 환경

2.2.2.1 하드웨어


Nvidia jetson TX2


Arduino Nano


Traxxas Slash


전자 변속기


Nvidia jetson TX2


Nvidia jetson TX2


오리코 CT2U3-10AB 10포트
유전원 멀티허브

어안렌즈 USB 카메라


모터 배터리 Traxxas


모터(속도)


모터(방향)

그림 4

2.2.2.2 소프트웨어

그림 5

2.2.2 하드웨어 구조

먼저, 차량에 Nvidia TX2를 메인보드로 하여 장착하였고, 이 메인보드는 메인 배터리로부터 전원을 공급받아 작동한다. 또한, 서브 배터리를 유전원 USB 허브에 연결하여 전원을 공급하였고 이를 메인보드에 연결했다. USB 허브에는 카메라와 네 개의 아두이노를 연결하였는데, 각각의 아두이노는 전자 변속기, 초음파 센서, 그리고 피에조 부저와 연결된다. 조향과 속도를 담당하는 서보 모터 두 개에 아두이노가 연결된 전자 변속기를 연결하여 모터를 제어했다. 이때 모터는 모터 배터리로 전원을 공급받는다. 그리고, 초음파 센서 여덟 개에 아두이노 두 개를 연결하여 초음파 센서를 사용하였으며, 피에조 부저는 아두이노 하나와 연결하여 사운드를 제어했다.

2.2.3 하드웨어 소개 및 구현

2.2.3.1 초음파 센서

장애물 인식을 위해 초음파 센서를 이용했다.

- 초음파 센서 동작 원리

초음파 센서 외관을 보면 눈처럼 보이는 원 두 개가 있다. 하나는 초음파를 쏘고, 나머지 하나는 초음파를 받는 역할은 한다. 거리 측정을 위해 송신부(trigger)에서 일정한 시간의 간격을 둔 짧은, 초음파 펄스를 방출하고, 대상물에 부딪혀 돌아온 신호를 수신부(echo)에서 받아 이에 대한 시간차를 기반으로 거리를 산출한다. 이를 통해 장애물의 유무, 장애물과의 거리를 측정할 수 있다. 이때, 초음파의 속도는 일반 공기중에서 약 340m/s이다.

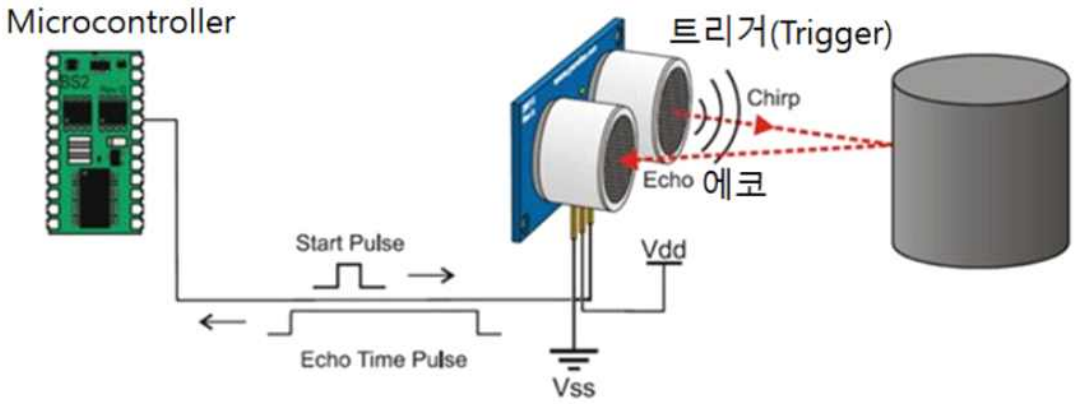


그림 6

- 초음파 센서를 이용한 거리 계산하기
위 두 값을 아래 식에 대입하여 물체와의 거리(cm)를 구했다.

$$\text{물체와의 거리}(cm) = ((340 \times \text{duration}) / 10000) / 2$$

그림 7

- 회로도

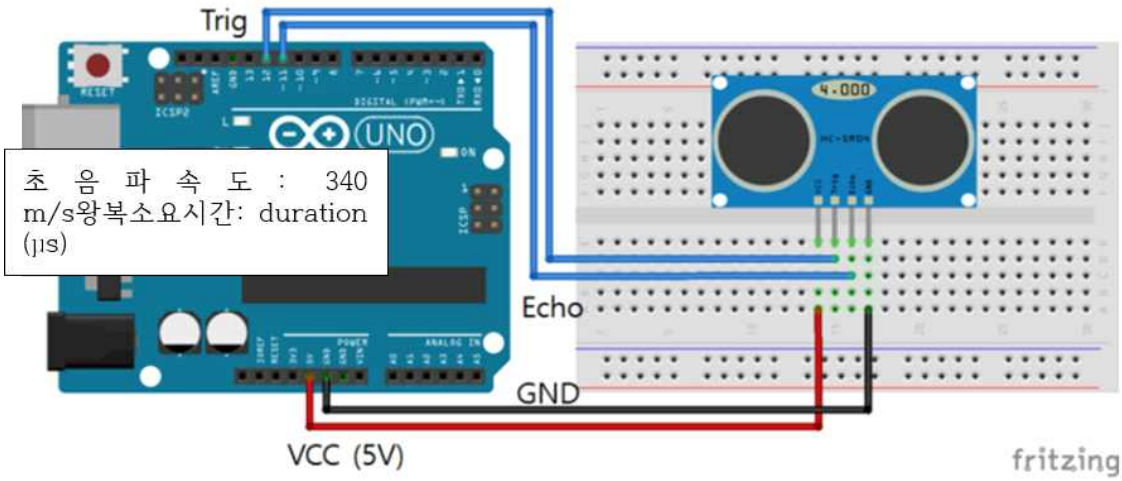


그림 8

다음 그림은 초음파 센서의 회로도이다. 초음파 센서는 4개의 핀을 가지고 있고, 4가닥의 점퍼선을 통해 아두이노에 직접 연결한다. Vcc와 GND는 센서에 전원을 넣어주기 위한 핀이다. Vcc는 아두이노의 5V, GND는 아두이노의 GND로 연결하면 된다. 나머지 Trig, Echo 핀은 사용자가 원하는 디지털 핀에 연결하면 된다.

- 초음파센서 S/W 구조

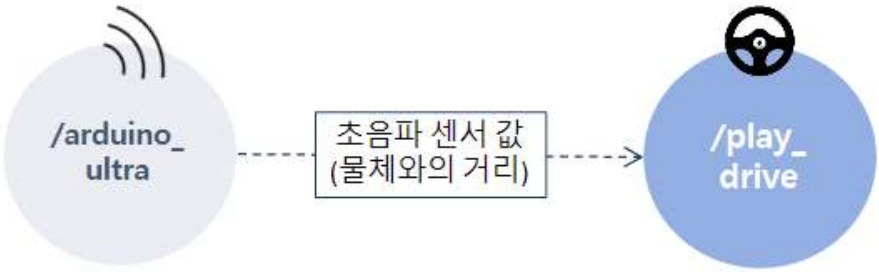


그림 9

초음파 센서는 계속해서 초음파를 쏘아 물체와의 거리를 계산하고, 계산된 값을 토픽으로 보낸다. 그러면, play_drive노드가 이 토픽을 받아 전체적인 주행을 통제한다.

2.2.3.2 피에조 부저

- 소리 출력

피에조 부저를 이용해 소리를 출력했다.

- 피에조 부저 원리

그림 10

피에조 부저는 피에조 효과를 이용하여 소리를 내는 작은 스피커이다. 피에조 효과란 수정이나 세라믹 같은 결정체의 성질을 이용하는 것으로 압력을 주게되면 변형이 일어나면서 표면에 전압이 발생하고, 반대로 전압을 걸어주면 응축, 신장을 하는 현상을 말하며 압전효과라고도 한다. 여기에 얇은 판을 붙여주면 미세한 떨림으로 인해 소리가 나게 된다.

(단위 : Hz)

옥타브 / 음계	1	2	3	4	5	6	7	8
C(도)	32.7032	65.4064	130.8128	261.6256	523.2511	1046.502	2093.005	4186.009
C#	34.6478	69.2957	138.5913	277.1826	554.3653	1108.731	2217.461	4434.922
D(레)	36.7081	73.4162	146.8324	293.6648	587.3295	1174.659	2349.318	4698.636
D#	38.8909	77.7817	155.5635	311.1270	622.2540	1244.508	2489.016	4978.032
E(미)	41.2034	82.4069	164.8138	329.6276	659.2551	1318.510	2637.020	5274.041
F(파)	43.6535	87.3071	174.6141	349.2282	698.4565	1396.913	2793.826	5587.652
F#	46.2493	92.4986	184.9972	369.9944	739.9888	1479.978	2959.955	5919.911
G(솔)	48.9994	97.9989	195.9977	391.9954	783.9909	1567.982	3135.963	6271.927
G#	51.9130	103.8262	207.6523	415.3047	830.6094	1661.219	3322.438	6644.875
A(라)	55.0000	110.0000	220.0000	440.0000	880.0000	1760.000	3520.000	7040.000
A#	58.2705	116.5409	233.0819	466.1638	932.3275	1864.655	3729.310	7458.620
B(시)	61.7354	123.4708	246.9417	493.8833	987.7666	1975.533	3951.066	7902.133

그림 11

위의 사진은 옥타브 및 음계별 표준 주파수를 나타내는 표이다. 피에조 부저에 주파수에 맞는 신호를 줌으로써 원하는 음계의 소리를 낼 수 있다. 본 프로젝트에서는 330Hz의 주파수를 가지는 4옥타브 E(미)음으로 소리를 출력했다.

- 회로도

피에조 부저에는 2개의 핀이 있다. 긴 쪽이 +이고, 짧은 쪽이 -이다. 짧은쪽은 아두이노의 GND에 긴쪽은 아두이노의 디지털 단자에 연결해준다.

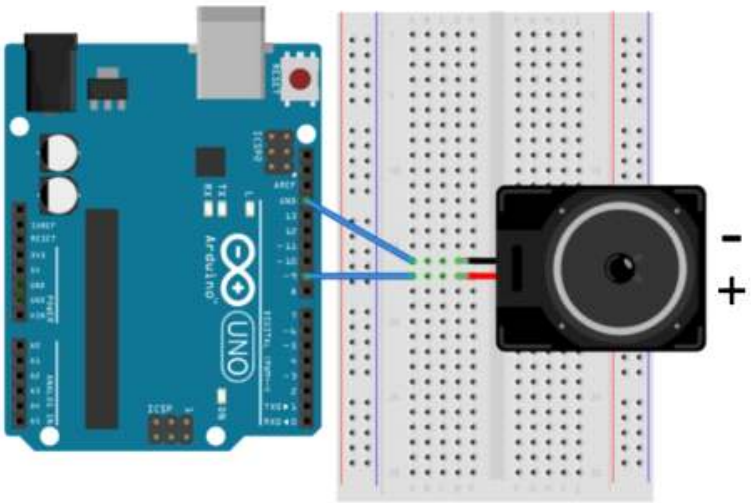


그림 12

- 피에조 부저 S/W 구조

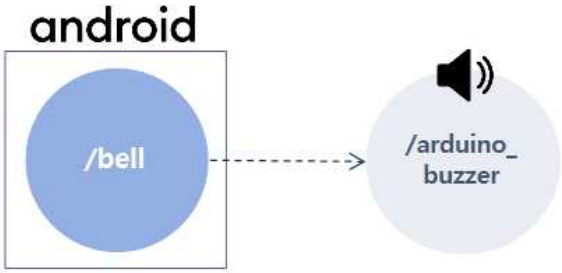


그림 13

사용자가 안드로이드 앱에서 소리 출력 버튼을 누르면 /bell 노드에서 버튼을 눌렀다는 신호가 토픽으로 날아가고, 이 토픽을 /arduino_buzzer 노드에서 받아 피에조 부저를 동작시켜 소리를 출력한다.

2.2.3.3 카메라

그림 14

usb 카메라는 160도 광각 카메라를 이용하였다. usb 포트를 통해 메인 보드와 연결된다.

그림 15

usb 카메라를 지원하는 uvc-camera 패키지를 설치하여 ROS에서 카메라를 사용할 수 있게 했다. uvc-camera 패키지의 /usb_cam 노드에서는 image_raw 토픽을 publish한다.

2.2.3.4 모터

모터를 제어하는 하드웨어는 휠을 제어하는 서보 모터와 속도를 제어하는 모터, 그리고 전자 변속기와 수신기로 구성된다. 속도 제어 모터는 전자 변속기와 연결되고 전자 변속기와 휠 제어 모터는 수신기와 연결된다. 그리고 수신기와 아두이노 나노를 연결하여 제어하는데, 점퍼 선 세 개를 이용하여 그라운드와 휠, 속도 선 각각을 아두이노 나노와 연결하였다. 그리고 아두이노 나노는 메인 보드와 연결되어 메인 보드에서 모터를 제어한다.

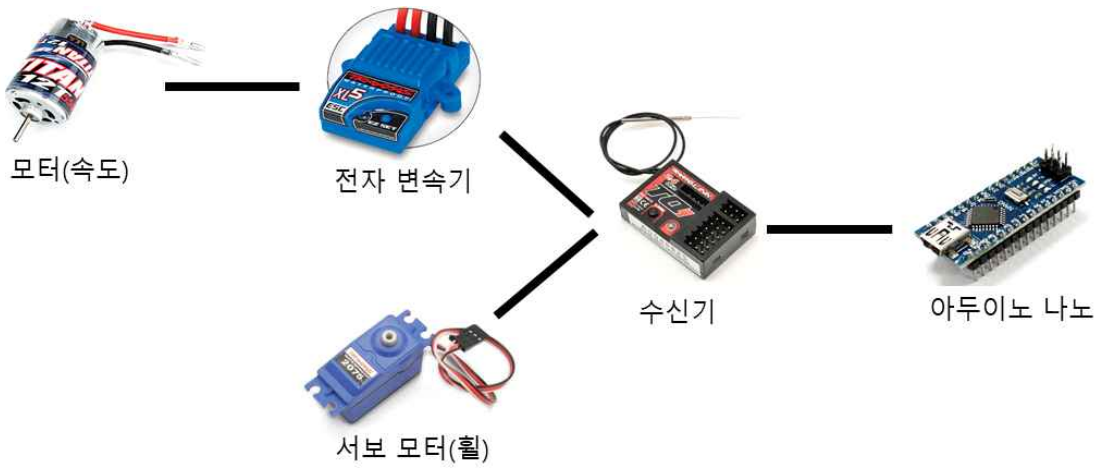


그림 16

2.2.4 소프트웨어 구조

2.2.4.1 휴먼 트래킹

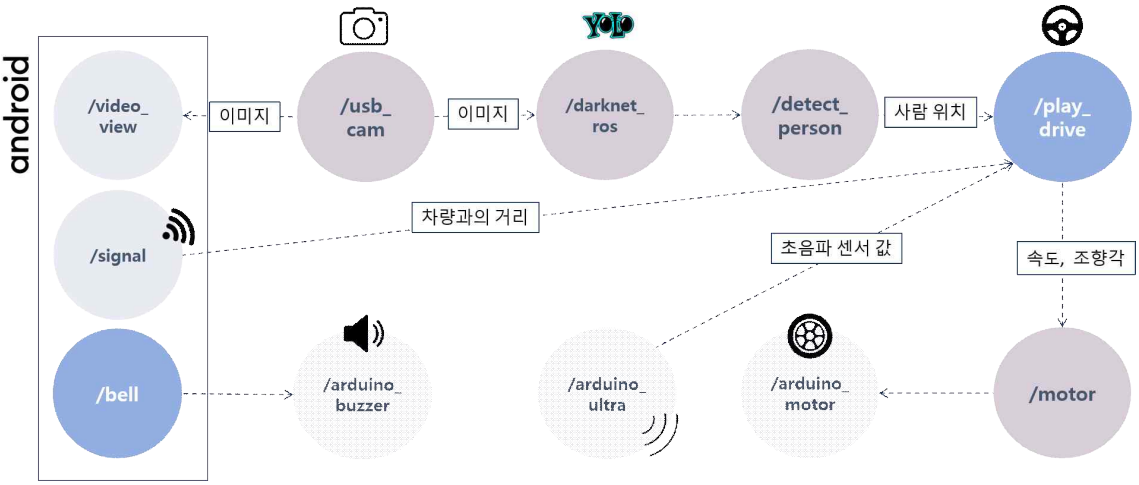


그림 17

휴먼 트래킹 기능에서 중심이 되는 노드는 play_drive 노드이다. play_drive 노드는 signal 노드와 arduino_ultra 노드에서 받은 값으로 속도 값을 정하고, detect_person 노드로부터 받은 값으로 조향각을 계산하여 motor 노드에 전송한다.

사용자는 안드로이드 애플리케이션을 이용해 메인 보드(NVidia TX2)의 AP에 접속할 수 있다. 애플리케이션 속 signal 노드는 메인 보드의 와이파이 신호의 세기를 이용해 차량과의 거리를 계산한다. 계산한 거리를 play_drive 노드에 전송한다.

그림 18

aruino_ultra는 초음파 센서에서 측정한 값을 play_drive 노드에 전송한다.

usb_cam 노드는 카메라를 제어하여 실시간 이미지를 안드로이드 애플리케이션 속 video_view 노드와 메인 보드의 darknet_ros 노드에 전송한다.
따라서, 사용자는 안드로이드 애플리케이션을 통해 차량의 카메라에서 촬영되는 실시간 이미지를 확인할 수 있디.

darknet_ros 노드는 YOLO 라이브러리를 이용해 물체를 인식하고, 인식된 물체를 detect_person 노드에 전송한다. detect_person 노드는 인식된 물체 중 사람의 좌표값, 즉 위치를 play_drive 노드에 전송한다. 이때, 좌푯값은 이미지 상의 좌푯값을 의미한다.

signal, detect_person, arduino_ultra 노드에서 토픽을 받은 play_drive 노드는 계산한 조향각과 속도를 motor 노드에 전송한다. motor 노드는 전달받은 속도와 조향각 값을 아두이노에서 처리하는 단위로 변환하여 arduino_motor 노드에 전송한다. arduino_motor 노드는 전달받은 속도와 조향각으로 아두이노를 제어하여, 차량은 사용자와 일정 거리를 유지하며 추적 주행한다.

안드로이드 애플리케이션 상의 “내 카트 찾기” 버튼을 누르면 bell 노드에서 arduino_buzzer 노드에 토픽을 전송하고, arduino_buzzer 노드가 피에조 부저를 제어하여 소리를 낸다. 이 소리를 통해 사용자는 차량의 위치를 파악할 수 있다.

2.2.4.2 조이스틱

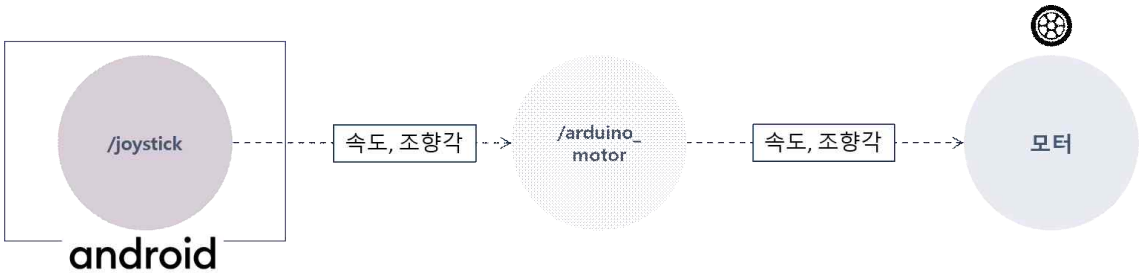


그림 19

사용자는 안드로이드 애플리케이션을 이용해 차량을 조종할 수 있다. 애플리케이션 상의 steering wheel과 pedal들을 이용해 차량의 속도와 조향각을 조절해 차량의 arduino_motor 노드에 전송한다.

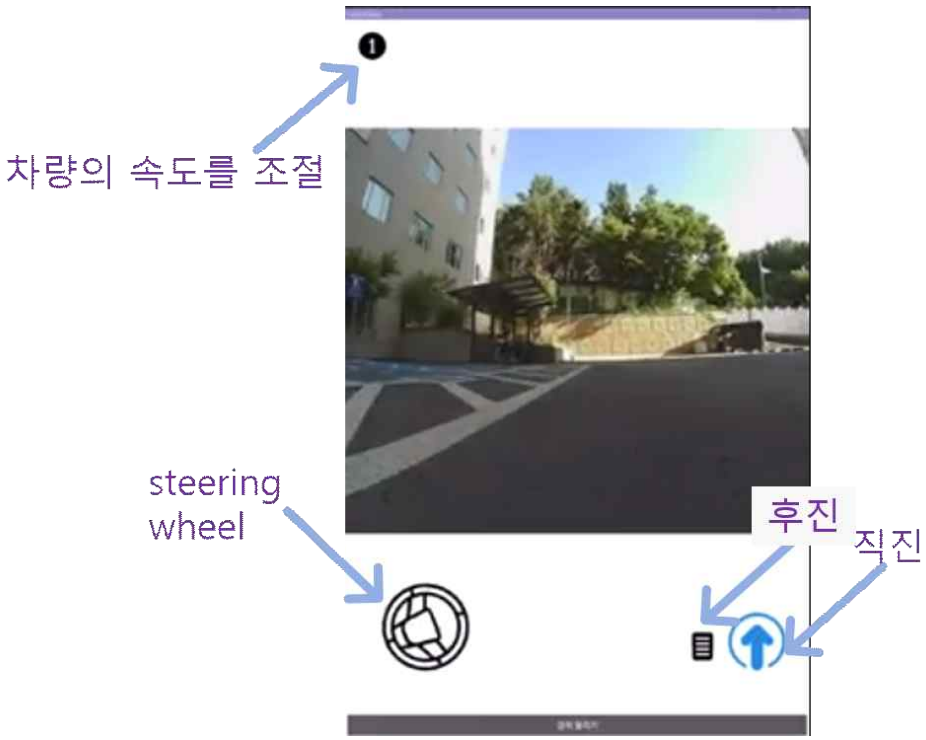


그림 20

2.3. 프로젝트 결론 및 기대 효과

그림 21

본 프로젝트에서는 자율 주행하는 휴먼 트래킹 차량을 구현하였다. 아스팔트 도로에서 벗어나, 일상에도 적용할 수 있지 않을까? 하는 생각에서부터 본 프로젝트가 시작되었다.

처음엔 두 손이 자유로운 쇼핑 시간을 떠올렸다. 버거운 장바구니를 이끄느라 고통받았을 쇼핑 시간이 쾌적해지길 바랐다. 이러한 생각에서 장바구니가 스스로 사용자를 따라가기 위해 트래킹 기술을 자율 주행과 접목했다.

Human Tracking과 자율 주행 차량은 감시, 관리의 형태로 확장할 수 있다. 노인 돌봄 서비스와 같은 관리의 형태부터, 범죄자 감시 시스템 같은 감시의 형태로 확장할 수 있을 것이다.

더불어, 인력 낭비 감소의 효과도 기대할 수 있다. 범죄자 관리 시스템에 적용한다면, 범죄자 한 명을 감시하기 위해 많은 인력이 동원되지 않아도 될 것이다.

스마트 장바구니 같은 경우는 두 손이 자유로워진다는 큰 장점이 있다. 마트나 시장에서의 사용에 국한되지 않고, 아이를 챙기기 바쁜 양육자, 혹은 짐을 챙기기 힘든 노약자와 같은 대상들의 일상의 불편함을 해소할 수 있을 것이다.

바쁜 현대 사회, 본 프로젝트를 적용하여 일상을 윤택하게 하고, 쾌적해지길 기대한다. 이로부터 쇼핑 카트 스스로 사용자를 따라가기 위해, 타겟 추적, 즉 불편한 짐에서부터 벗어나 자유로운 시간을 갖는 것이 본 프로젝트에서 기대하는 효과이다.

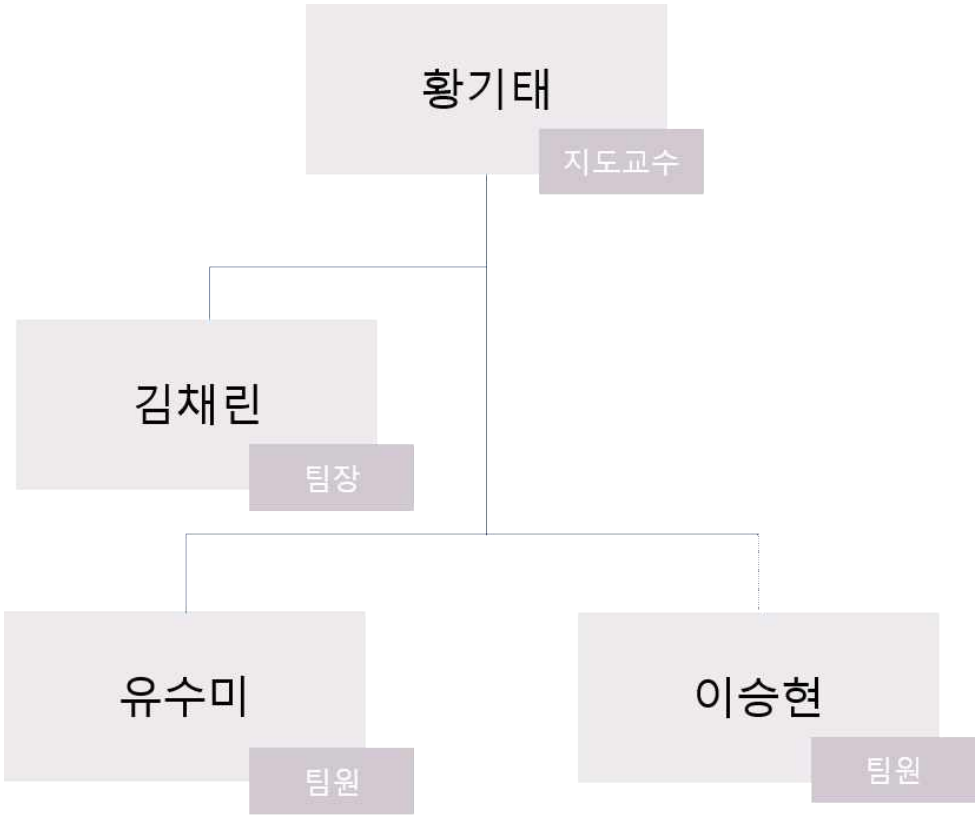
2.4. 프로젝트의 한계

본 프로젝트의 차량은 사람을 쫓아가기 때문에, 사람의 뒷모습만 보고 타겟을 구분해야 하지만, 뒷모습만으로 타겟을 추적하기에는 어려움이 있다. 따라서, 이미지가 아닌 새로운 방법으로 타겟을 추적할 필요가 있었다.

프로젝트 진행 초기에는 차량의 블루투스 신호 세기나, 와이파이 신호 세기를 이용해 군중 속에서 타겟을 확정지으려 했다. 차량이라는 하나의 '점'에서 사용자의 위치한 범위는 알아낼 수 있지만, 정확한 방향은 알아내기 어려웠다. 또한, 신호가 불안정해서 유효한 결과는 도출해내기 어려웠다. 이 때문에 군중 속의 타겟을 찾아내는 방법은 고안해내지 못한 점이 아쉽다.

3. 프로젝트 수행 추진 체계 및 일정

3.1 각 조원의 조직도



3.2 역할 분담

이름	역할
김채린	usb 카메라 작동 구현. 와이파이 신호 이용한 거리 측정 구현. 안드로이드 와이파이 신호 측정 가능 구현
유수미	초음파 센서 설계 및 구현, 안드로이드 조이스틱 어플 구현, 피에조 부저 설계 및 구현, 물체 인식 구현
이승현	모터 작동 구현, 전체 주행 구현, 안드로이드 장바구니 어플 구현, 물체 인식 구현

3.3 주 단위의 프로젝트 수행 일정

주차	수행 일정
2주차	주제회의, 관련 자료 수집, RC카 개조
3주차	주제회의, 관련 자료 수집, RC카 개조, 아두이노 프로그래밍
4주차	엔비디아 보드 프로그래밍 환경구축, 아두이노 프로그래밍
5주차	USB 랜카드 이용한 AP모드 설정 방법 연구, python3 가상환경을 위한 아나콘다 설치 및 세팅
6주차	tensorflow를 이용해 사람 전신에 대한 학습 모델 생성
7주차	USB 랜카드를 이용해 메인보드의 AP모드 설정, GPU를 사용해 메인 보드의 성능 개선
8주차	물체 인식을 위한 tensorflow 코딩 및 테스트
9주차	버전 충돌 문제로 버전 업그레이드, rosjava 관련 자료 수집 및 공부
10주차	특정 타겟을 찾기 위한 모델 학습
11주차	이미지 학습 횟수와 개수에 따른 결과의 차이를 비교해 최적의 환경 연구
12주차	타겟과 타겟이 아닌 이미지로 클래스를 분류해 학습하는 모델을 구현, 차량과의 거리에 따른 와이파이 신호 세기를 수집해 방정식을 세움
13주차	구현한 차량을 야외에서 시연 및 오류 개선

4. 참고 자료

- ROS.org
- 피에조 부저-소리내기. kocoafab
- Do It Yourself!: [아두이노] 피에조 부저(Piezo Buzzer)를 이용해서 소리를 내어보자. 네이버 블로그
- 도매키트 아두몰의 블로그: 아두이노 초음파센서(HC-SR04)사용 예제설명. 네이버 블로그
- Setting up wifi Access Point on TX!. Nvidia developer Forum
- https://github.com/ros-drivers/usb_cam. 깃허브
- ROS에서 USB 카메라 사용 & Subscriber 노드 작성, Youtube
- darknet_ros를 이용하여 사물 인식하기, https://github.com/leggedrobotics/darknet_ros, 깃허브