

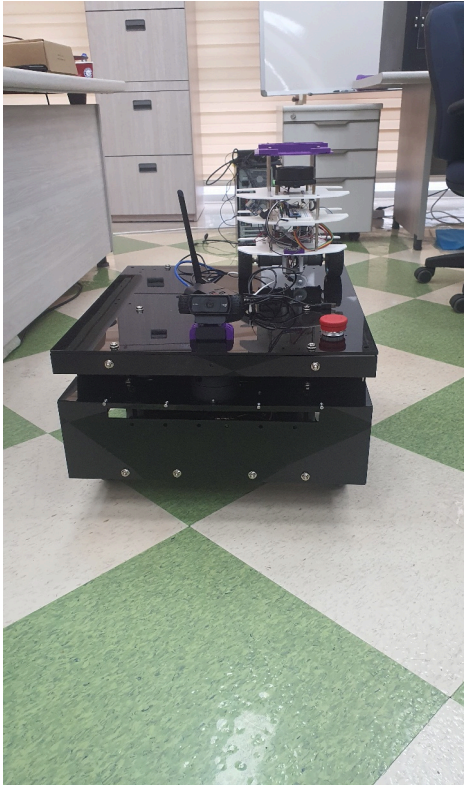


제19회 임베디드SW경진대회 개발완료보고서

[자유공모]

□ 개발 요약

팀 명	WLSISULLIVAN
  	
작품명	시청각 장애인을 위한 보조로봇 "SULLIVAN ROBOT(설리번 로봇)"
작품설명 (요약)	위 작품은 시각, 청각능력 없이 오직 촉각만을 느낄 수 있는 시청각 장애인들을 위해 만들어졌으며, 로봇이 사용자의 특정한 손짓을 인식하여 사용자를 특정 장소까지 안전하게 인도하는 로봇이다.
소스코드	https://github.com/ggh-png/MR_2
시연동영상	https://youtu.be/FCpglSeV2Cw

□ 개발 개요

○ 개발 작품 동기

- 시각 청각 능력을 상실하여 조용한 어둠속을 홀로 걸어가고 있는 한국의 시청각 장애인은 현재 복지 사각지대에 놓여 있다. 때문에 2017 시청각 중복지장애인의 욕구 및 실태조사 연구자료에 의하면 한달동안 외출하지 못하는 시청각 장애인의 비율은 전체 장애인 보다 3배 높고, 의무교육조차 받지 못하는 시청각 장애인의 비율 또한 3배 높은 것으로 알려져 있다.

비율을 보면 시청각 장애인의 이동과 의무교육은 관련이 있어 보이고 실제로 장애인은 특정 교육시설에서 교육은 받는 것이 보통의 경우이나 국내에는 시청각 장애인 교육센터가 전무한 것으로 알려져 있으며, 인력조차 턱없이 부족한 것으로 뉴스 기사가 나오고 있다.

이런 문제를 인식하게 되었고, 위 로봇을 통해 시청각 장애인이 스스로 할 수 있는 행동의 범위 즉 활동 범위를 늘려 시청각 장애인의 삶의 질을 높이기 위해 제작하게 되었다.

○ 개발 목표

- 사용자의 신호 인식

시청각 장애인이 인간으로서 필요한 것(배고픔, 목마름, 화장실,도움요청)들을 보호자 또는 보호사에게 알리고 싶을 경우 특정 손짓을 한다. (사용자에게 미리 안내를 한 후 사용하게 한다) 이 동작을 인식한 로봇은 보호자에게 사용자의 상태를 태블릿의 UI를 통하여 알려주거나 도움을 요청하는 기능을 만드는 것을 목표로 하였다.

- 사용자의 인도기능.

로봇은 등록된 장소의 좌표 정보를 바탕으로 사용자가 그 장소로 가길 원한 다는 손짓을 인식하면 그 손짓에 맞는 좌표점으로 이동한다.

사용자를 인도할 때 로봇에 장착된 센서(카메라,라이다 등)를 이용하여 장애물을 피하며 사용자를 인도한다.

○ 개발 작품의 필요성

한국의 농학교나 맹학교조차 대부분 시청각 장애인 학생을 부담스럽게 여겨 받기를 꺼려하는게 현실이다. 시청각 장애인 자녀를 둔 부모 김씨는 “찾아가 물어보면 ‘여긴 그런(시청각 장애인) 학생이 없다. 다른 곳으로 가서 배워보면 어떠냐’ 라는 대답만 돌아왔다” 고 했다. (국민일보 참고)

위의 기사에서 보듯 시청각 장애인을 위한 교육시설이 없고 다른 장애인시설 같은 경우 시청각 장애인을 꺼리는데 그 이유에는 아래와 같은 이유들이 포함된다.

- 인력 부족

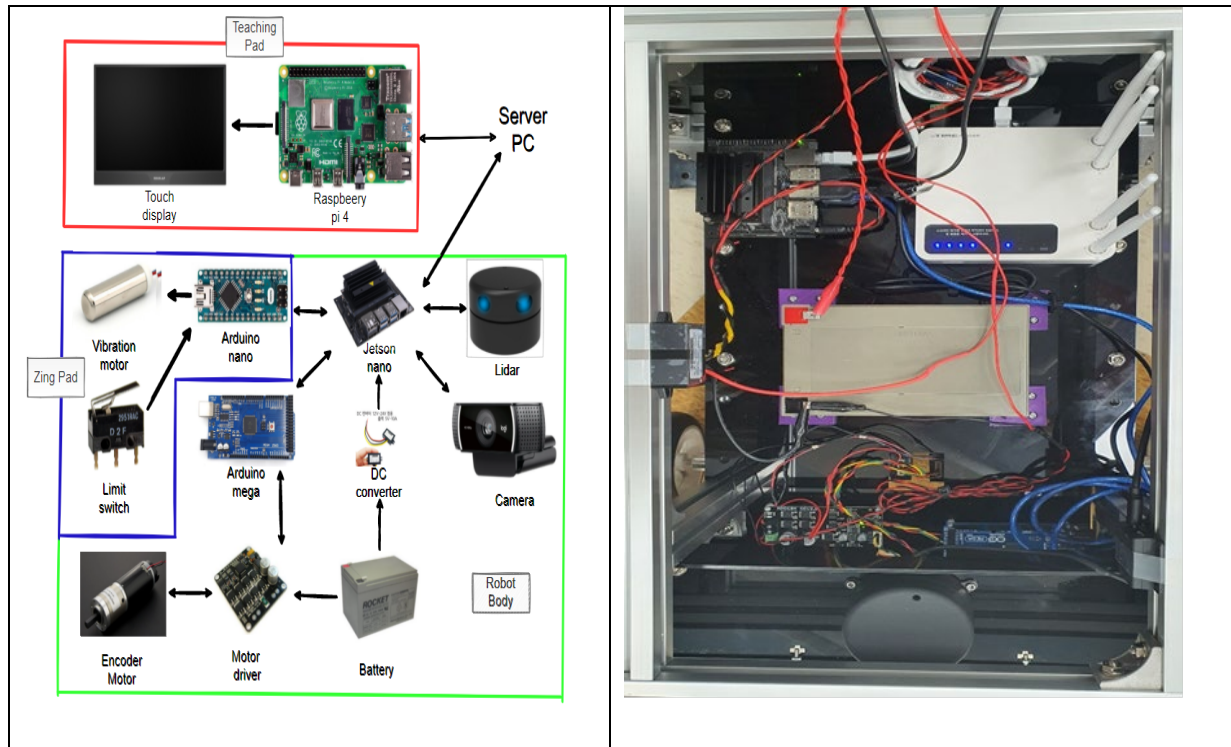
개발 동기에서 설명했던 장애인의 지원에 대한 인력이 부족한 것 특히 시청각 장애인은 위의 기사에서 다루듯이 다른 장애인과 비교했을 때조차 인력과 시설이 부족하며 이런 문제점은 결국 시청각장애인의 교육의 질 삶의 질과 연관돼 시청각장애인들을 교육할 수 있는 사람의 유무는 매우 중요하다고 할 수 있다. 이런 인력 부족의 대표적인 사례로 ‘한국의 헬렌 켈러’ 꿈이 사라졌다가 있다.

시청각 장애인의 케어 또한 중요하다 농학교나 맹학교가 시청각장애인을 꺼리는 이유 중 다른 하나는 소통일 것이다. 말을 하여 소통할 수 있는 극히 일부의 시청각 장애인도 있지만 대부분의 시청각 장애인은 수화를 만져서 이해하는 촉수화를 사용한다. 도움이 필요하거나 무언가 전달 하고싶을 경우 일반적인 농인은 직접 대상을 찾아가 수화로 의사를 전달할 수 있고 맹인의 경우 소리를 내어 소통할 수 있으나 시청각장애인은 자신이 소리를 내는 것조차 듣지 못하여 소리를 의사를 전달하기 어렵고 앞을 볼 수 없어 사람을 찾기 힘들며 장애물과 사람들로 인한 부상의 위험까지 있다.

즉 시청각장애인에게 인력이 부족해서 사람을 붙이는 것은 현재로선 힘들고 보호자가 같이 붙어 있는 것도 보호자의 삶의 질과 재정적 어려움 시간의 한계가 있기 때문에 우리는 시청각장애인이 스스로 할 수 있는 행동 반경을 넓혀 도움이 필요한 상황을 줄이면 더 적은 인원으로 시청각 장애인에게 도움을 줄 수 있을 것이라 생각하여 시청각장애인을 위한 로봇을 제작하였다.

□ 개발 환경 설명

○ Hardware 구성



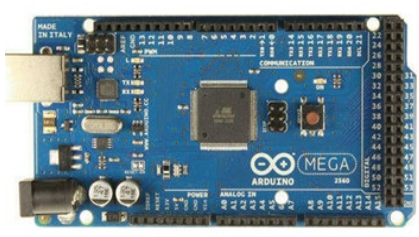
H/W 구성은 시청각 장애인을 인도할 수 있는 differential형태의 모바일 로봇, 사용자와 로봇 간의 링크 시스템인 징 패드, UI 제공과 로봇, 사용자의 상태를 보여주고 제어하는 티칭패드, 전반적인 시스템을 관제하는 서버 컴퓨터로 구성되어 있다.



로봇은 SBC(젯슨나노)를 중심으로 센서들이 연결되어 있어 1차적으로 데이터를 처리하도록 하였고, 모션인식이나, 자율주행과 같은 보다 무거운 연산들은 서버 컴퓨터에서 처리하도록 구성하였다.

○ Hardware 기능 (제어 방법 등 서술)

- 엔코더 DC모터, 2채널 모터드라이버, 아두이노 메가

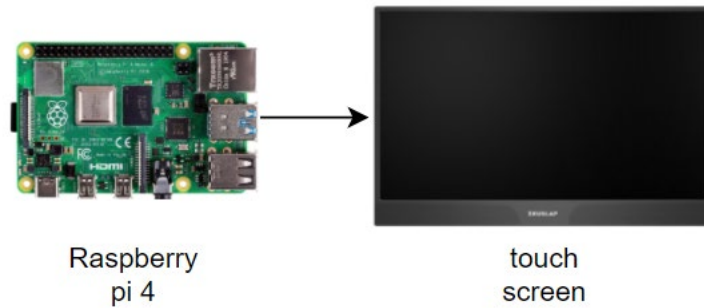


모터는 로봇을 구동하는데 쓰이며 모터 컨트롤러는 MCU(아두이노 메가)와의 GPIO를 통해 속도 값(PWM)과 방향(IO)을 제어한다. 모터에 장착된 엔코더는 모터의 속도와 방향 값을 MCU에 제공한다. MCU는 제공된 엔코더 값을 이용하여 PID 제어를 통해 모터의 속도와 방향을 제어하고 동시에 SBC(젯슨나노)와 통신을 하며 로봇을 제어한다.

	<p>라이다의 역할은 모터의 엔코더 값과 센서를 퓨징하여 실내 공간의 2D지도를 생성하고 로봇을 실행시켰을 때 로봇을 지도의 지정한 위치로 갈 수 있도록 하며, 동시에 실시간 장애물 회피를 하는 SLAM/NAVIGATION에 쓰인다.</p>
	<p>사용자의 모션을 인식할 수 있도록 이미지 데이터를 전하는 이미지 센서 역할을 하며 사용자의 이미지 데이터를 SBC(젯슨 나노)에 전송한다.</p>
	<p>젯슨나노는 각종 센서 값(카메라, 라이다, 구동 부, 징패드)들을 입력 받아 1차적으로 (SLAM/NAVI, CV 등)을 하기위한 데이터를 연산, 가공하여 TCP/IP 통신으로 서버 컴퓨터와의 쌍 방향 통신을 통해 로봇을 구동한다.</p>

○ Hardware 기능 (제어 방법 등 서술)

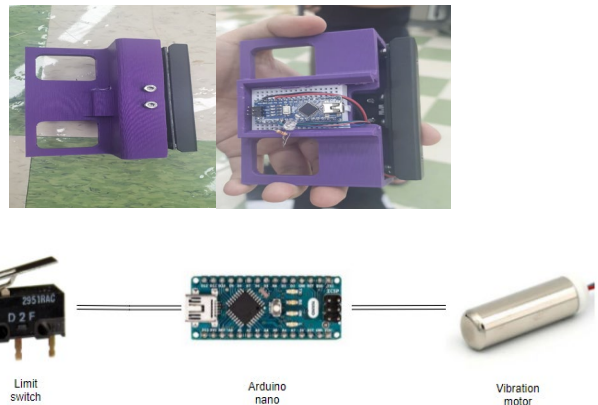
- 티칭패드 - 라즈베리파이 4, 터치 디스플레이



티칭패드는 라즈베리 파이와 터치가 가능한 디스플레이로 이루어져 있다. 티칭패드는 로봇 세팅을 개발자만이 아니라 일반 사용자도 쉽게 사용할 수 있도록 직관적인 UI 와 터치 디스플레이로 구성되어 있다.

서버컴퓨터와 연동이 되어있어 기본적인 로봇구동이 가능하며, 실시간으로 사용자, 로봇의 상태를 확인할 수 있다.

- 징 패드 - 리미트 스위치, 아두이노 나노, 진동모터



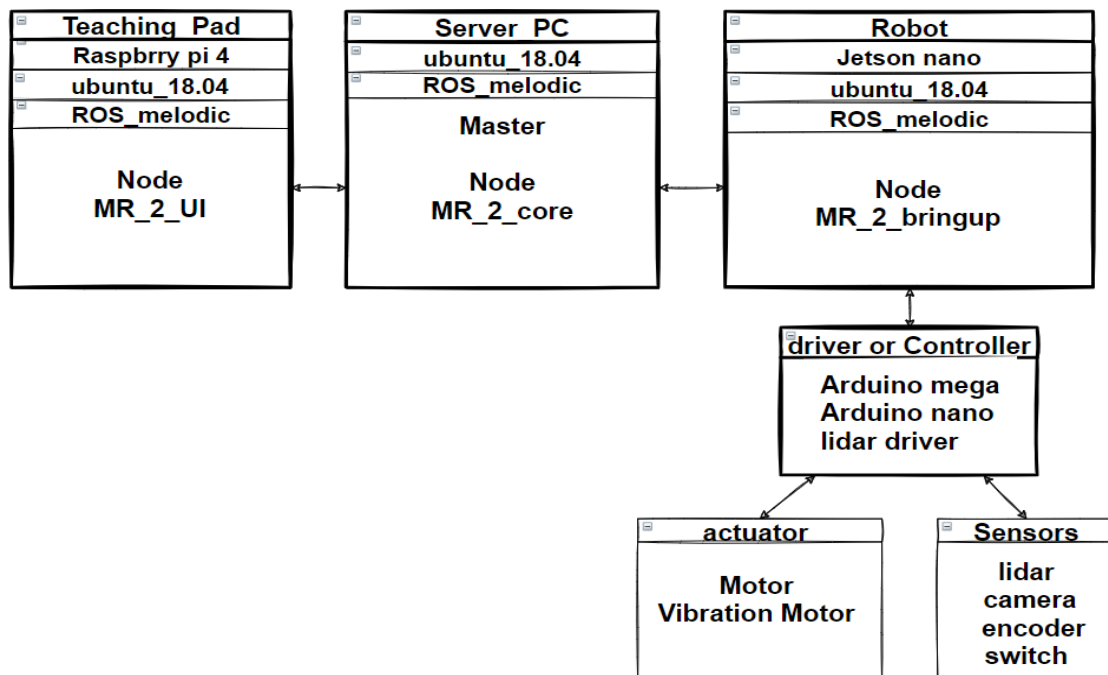
징 패드는 오직 촉각만을 느낄 수 있는 시청각 장애인들을 위해 로봇과 사용자 간의 링크 시스템으로서 제작되었으며, 실시간으로 로봇의 상태를 사용자에게 전달하고, 로봇을 제어한다. 진동모터는 진동의 패턴을 이용하여 로봇의 상태를 사용자가 인식할 있도록 하는 기능(I/O를 위해 MCU(아두이노 나노)로 제어된다. 미트 스위치는 사용자가 로봇을 제어하기 위해 쓰였으며, MCU에 IO값을 제공한다. CU는 스위치에 의해 제공된 IO값을 가공하여 버튼을 누른 회수를 기록하고 동시에 SBC(젯슨나노)와 양 방향 통신을 하여 로봇과 사용자를 링크한다.

○ Software 구성

전반적인 S/W 구성은 다음 블록선도와 같다. slam/navigation기능과 모션인식 기능을 탑재한 자율주행 모바일 로봇을 구동과 운용, 제어를 하기 위해 노드 단위의 관리가 편한 ROS(Robot Operating System)를 사용하였다.

개발환경은 Ubuntu같은 OS 위에서 동작하기에 각각 티칭패드, 서버 컴퓨터, 로봇에는 ubuntu 18.04의 ros melodic 버전을 동일하게 맞추어 사용하였다.

펌웨어 개발은 아두이노 nano, mega와 같은 MCU를 사용하여 드라이버를 만들었다. 만들어진 드라이버는 ROS를 통해 처리된 데이터들을 전달받아 하드웨어를 직접 구동 시킴과 동시에 센서 데이터를 ROS로 보내도록 하였다.



Teaching Pad - MR_2_UI

티칭패드에서 실행되며 GUI 표현 되어있어, 터치로 조작이 가능하며 크게 Setwaypoint, Waypoint, MP_state, Telop의 기능들이 포함되어 있다. 로봇의 초반 세팅, 디버깅에 쓰인다.

Setwaypoint :

로봇 구동 시 로봇이 위치한 지점의 좌표점을 저장하도록 MR_2_core 노드에 수신한다.

Waypoint :

위에서 저장된 좌표점까지 이동하도록 MR_2_core 노드에 수신한다.

MP_state :

모션 인식이 된 사용자의 현 상태를 MR_2_core 노드에서 받아 UI에 표시한다.

Telop :

로봇의 기본적인 구동을 하도록 직진, 회전 값을 MR_2_core 노드에 수신한다.

Server - MR_2_core

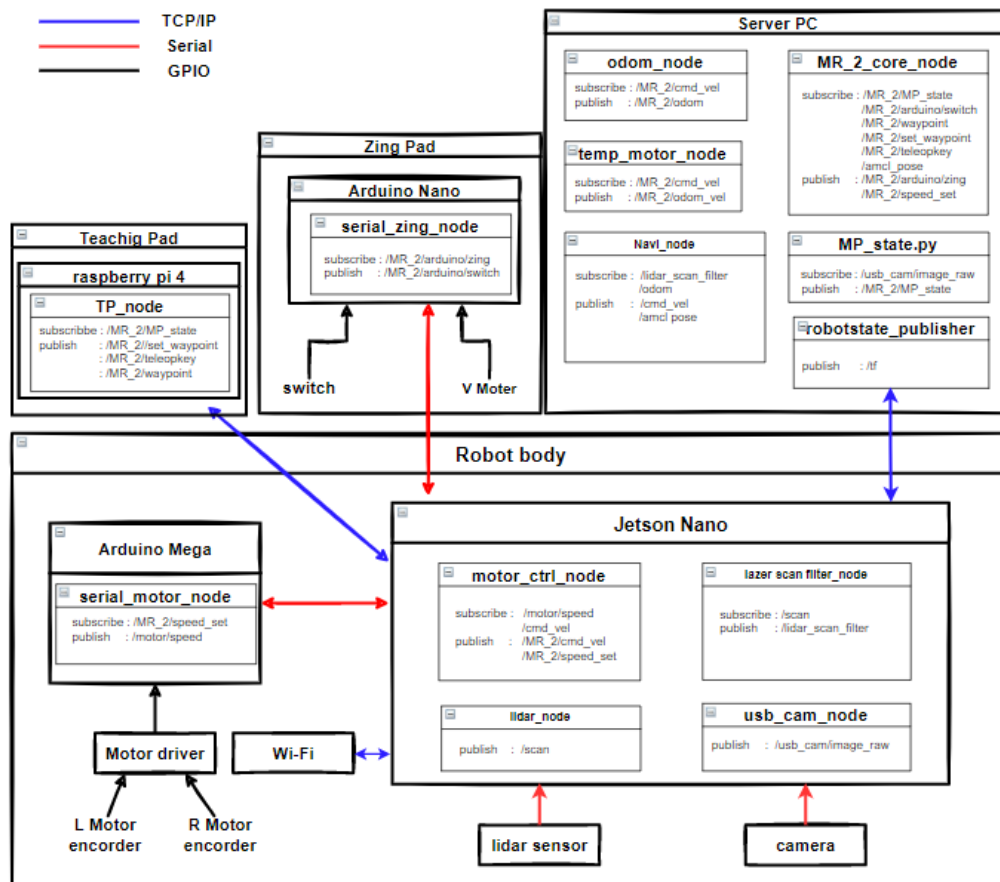
서버 컴퓨터에서 실행된다. 비교적 무거운 연산이 필요한 slam/navigation, 모션인식을 로봇에서 들어온 센서 값들을 이용하여 처리한다. 주어진 시나리오에 맞추어 로봇이 임무를 수행하도록 한다.

robot - MR_2_bringup

로봇에 내장되어 있는 SBC(젯슨나노)에서 실행된다. 센서(lidar, motor, camera 등) 값들을 MCU(arduino nano, mega, lidar drive 등)로부터 받아 1차적으로 처리 후 MR_2_core 노드로 송신하며, 직접적인 로봇의 제어를 한다.

○ Software 설계도 (흐름도 및 클래스 다이어그램 등 / 개발언어에 따라 선택)

로봇, 티칭패드, 서버컴퓨터를 포함한 전체적인 software의 흐름과 설계도는 아래의 블록선도와 같다. ROS를 사용하여 노드 단위로 나누어 설계되었으며 화살표의 방향은 통신방향을 뜻한다. 각각의 노드들은 메시지를 보내는 publisher와 메시지를 받아들이는 subscriber의 기능을 가진 토픽으로 이루어져 있으며 각각의 노드끼리 토픽의 이름에 맞추어 서버 PC에 있는 master를 거쳐 TCP/IP통신을 하게 된다.

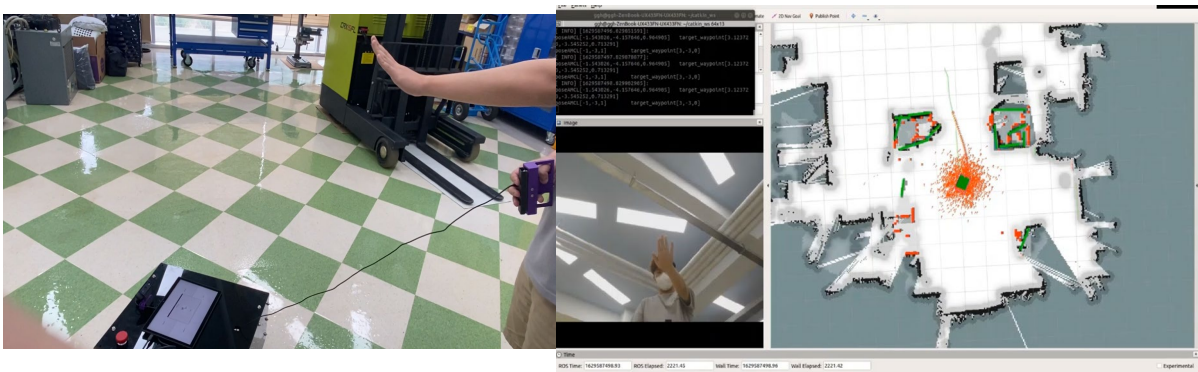


○ Software 기능

Software 기능은 상황에 따라 2가지로 나뉜다.

상황 1. 일상적인 상황인 경우

사용자(시청각 장애인)가 로봇에 내장된 징 패드의 버튼을 누르게 되면 징 패드가 진동하면서 사용자는 로봇에게 손짓으로 명령을 내릴 수 있는 조건이 된다. 로봇은 총 3가지의 손짓을 인식할 수 있으며 손짓에 해당되는 의미는 hungry, thirsty, toilet으로 로봇이 사용자의 손짓을 인식할 때까지 울리게 된다. 로봇이 사용자의 손짓을 인식하면 진동이 멈춤으로써 사용자에게 성공적으로 손짓이 인식이 되었다고 알린다. 인식된 손짓은 손짓이 의미하는 지점(hungry - 식당, thirsty - 식수대, toilet - 화장실)으로 스스로 장애물을 인식 회피하며 사용자를 인도함과 동시에 로봇위에 올려진 티칭패드에 로봇이 인도하는 장소의 이미지가 띄어 지면서 인도하고 있음을 제3자에게 알린다.

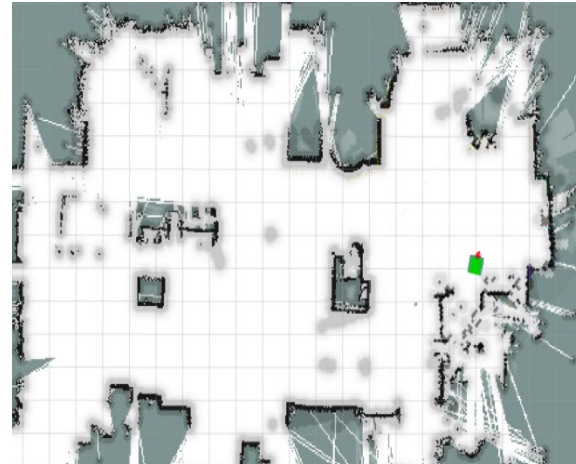
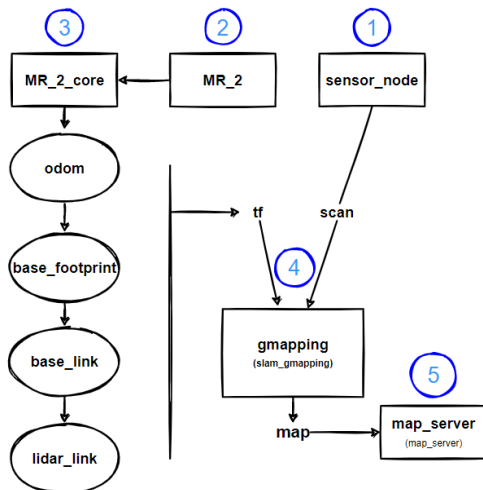


(사용자의 손짓을 인식하고 해당 목적지까지 인도하고 있는 모습)

상황 2. 긴급상황인 경우

긴급상황으로 사용자가 징 패드를 4번 연속으로 누르게 되면 징 패드에서 6번의 진동이 울리게 된다. 진동이 울린 뒤 사용자의 상태가 위급한 상태를 알리기 위해 로봇의 티칭패드 UI에 긴급함을 알리는 이미지가 띄어 지게 된다. 진동이 2번 울리게 되면서 사용자에게 출발을 알리고 중앙관리센터로 이동하게 된다. 목적지에 도착을 하게 되면 징패드의 진동이 3번 울림으로써 사용자에게 목적지 도착을 알린다.

SLAM(Simultaneous Localization And Mapping)을 우리말로 번역하면 동시적 위치 추정 및 지도 작성이라고 표현할 수 있다. 로봇이 자율주행을 하기 위해 로봇 스스로가 어떤 장소에 어디 지점에 위치해 있는지를 알아야 한다. 때문에 slam을 통해 로봇이 읽을 수 있는 map을 만들고 동시에 지도를 기준으로 로봇이 어느 지점에 위치해 있는지 계산해 낼 수 있으며, 이를 통해 로봇은 자신의 위치를 파악할 수 있게 된다. 아래의 블록선도를 통해 slam 알고리즘을 설명하겠다.



slam을 하기 위한 패키지)

(slam을 통해 만들어진 map)

1. 로봇을 구동 시키게 되면 라이다 센서에서 측정한 2D 평면 상의 거리 정보 raw data 가 메시지로 전달되며 토픽 이름은 scan 이라 전달된다.
2. MR_2_teleop 노드는 실제 로봇의 병진 속도, 회전 속도 값을 전달하여 로봇을 구동한다.
3. 구동한 로봇의 왼쪽, 오른쪽 바퀴 엔코더 값을 통해 바퀴의 위치를 측정할 수 있고, odometry 값을 이용해 이동 거리를 알 수 있다 이러한 정보를 사용하여 상대 좌표로 이루어진 base_footprint(바퀴와 바퀴 사이의 중심위치), base_link(로봇의 중심점), lidar_link(라이다의 위치)를 구할 수 있고, 이는 다시 tf(상대좌표 값) 메시지로 gmapping 노드에 전달된다.
4. Gmapping 은 지도를 만들어내고, 이를 map_server 를 통해 저장되며, 저장된 맵은 pgm 형식의 portable gray map, yaml 문서로 표현되며 로봇이 읽을 수 있게 된다.

Odometry : 로봇의 주행거리 정보를 뜻하며, 위치정보를 기록한다.

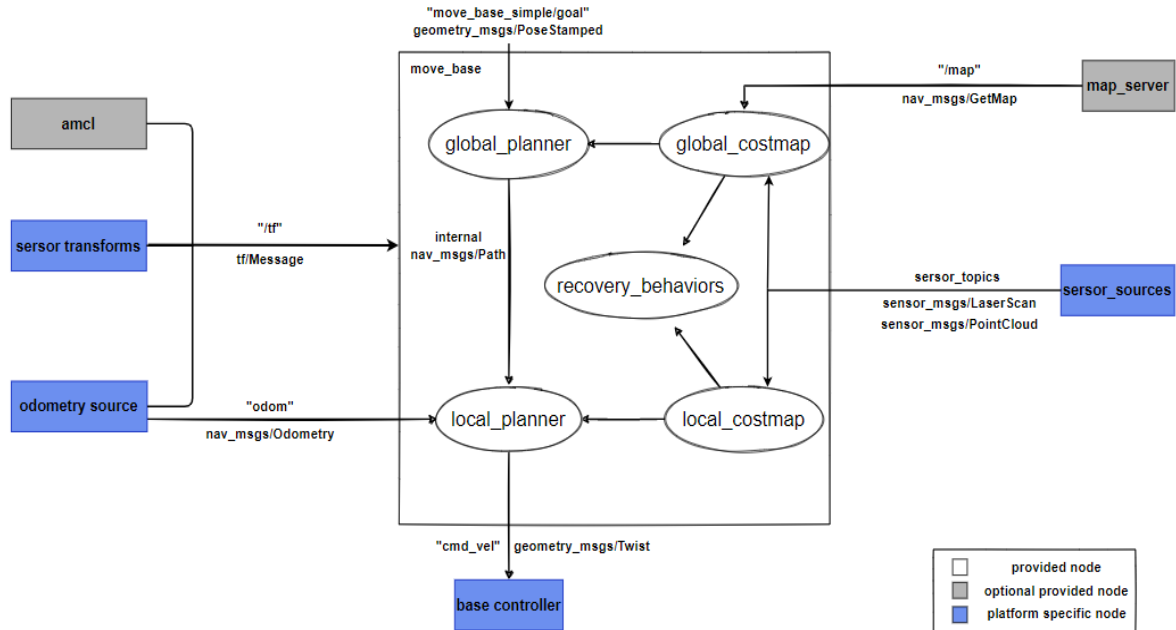
Tf : 로봇의 상대 좌표값을 실시간으로 계산해주는 ros 패키지

Gmapping : ros의 slam 패키지 중 하나, navigation시에 쓰이는 map을 제작한다.

MR_2_teleop : 로봇에 병진, 회전속도 값을 주어 로봇을 조종할 수 있도록 하는 노드

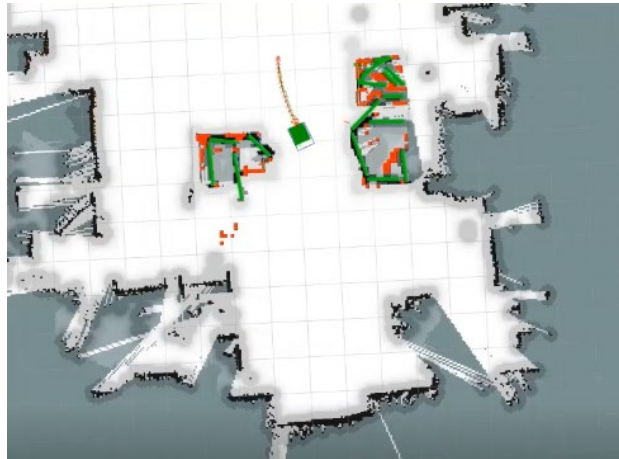
Navigation

Navigation 번역하면 항법이라 표현할 수 있으며, 여기선 한 장소에서 다른 장소로 장애물을 회피하며 최단경로로 이동하는 기술로서 쓰인다. 위 Slam을 통해 얻어진 map, odometry, scan값을 이용하여 로봇은 자신의 위치를 파악하고 장애물을 피해 목적지로 자율주행을 할 수 있는 navigation을 할 수 있는 조건이 된다. 아래의 블록선도를 통해 navigation 알고리즘을 설명하겠다.



1. /odom, nav_msgs/Odometry: odom 정보를 받아 local path 를 만들거나 장애물 회피에 사용한다.
2. /tf, tf/Message: 이전에 보았던 odom -> base_footprint -> base_link -> base_scan 의 변환을 거쳐 토픽을 퍼블리시 한다. move_base 노드가 이 정보를 받아 로봇의 위치와 센서의 위치를 가지고 이동 경로를 계획한다.
3. 거리 센서 /scan, sensor_msgs/LaserScan 를 사용하여 측정된 거리 raw data 이며 로봇 위치 추정 방법인 AMCL 을 이용하여 로봇의 현재 위치 추정, 로봇의 모션 계획에 사용된다.
4. 지도 /map, nav_msgs/GetMap: 내비게이션에서는 occupancy grid map 을 사용합니다. 위 사진상에 회색 부분은 로봇이 이 회색 지점에 도달할 수록 충돌할 확률이 커진다는 의미를 가진다.
5. 목표 좌표 /move_base_simple/goal, geometry_msgs/PoseStamped: 사용자가 직접 지정하는 좌표로, 2 차원 좌표 x,y 와 자세 i 로 구성되어 있으며, 로봇은 x, y 좌표로 이동한 뒤 자세를 맞추게 된다.

6. 속도 명령 /cmd_vel, geometry_msgs/Twist: 최종적으로 계획된 이동 궤적에 따라 로봇을 움직이는 속도 명령을 퍼블리시하여 로봇을 목적지까지 이동시킨다.

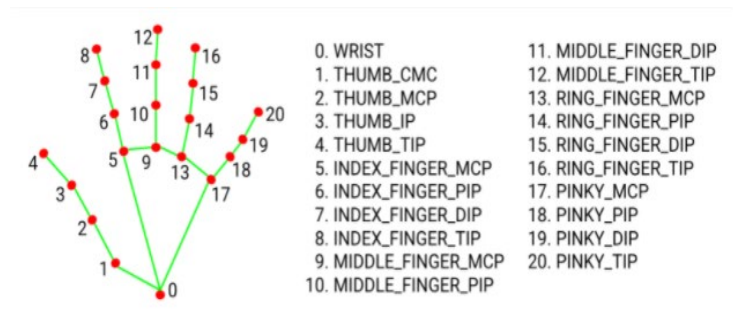
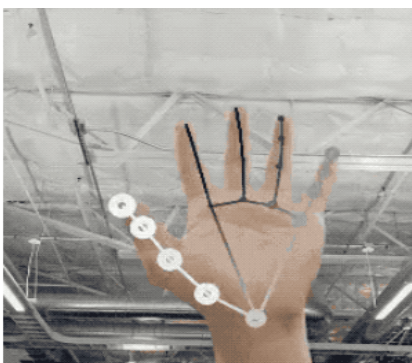
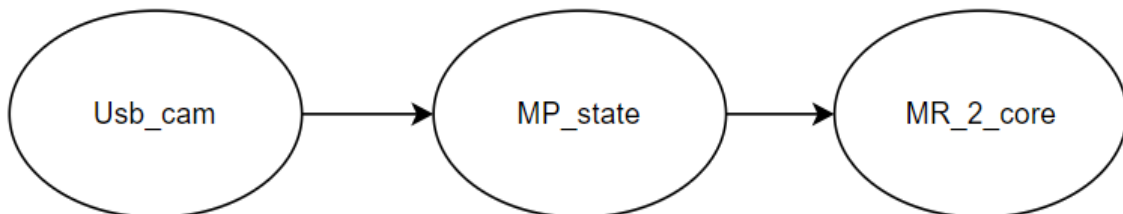


(map을 바탕으로 실행된 navigation)

Local path : navigation시 생성되는 로봇의 경로로, global, local두개의 경로가 생성된다.

손짓인식

손짓 인식은 로봇에 장착된 웹캠을 통해 rgb값을 usb_cam 노드로 받아 ros 통신을 통하여 MR_2_MP 노드로 보내지게 되고 MR_2_MP에서 미리 학습이 된 모델을 통해 특정 손짓을 인식 후 메시지를 검출, MR_2_core 노드로 전송하게 된다. 아래의 노드를 통해 인식이 이루어지는 과정을 설명하겠다.



1. Usb_cam을 통해 들어온 손짓 이미지(rgb 값)는 MP_state 노드를 통해 각각의 관절마다 랜드마크가 생겨 각각의 좌표 값(x, y, z)들이 생성된다.

2. 생성된 좌표 값은 미리 학습 시켜 둔 제스처 데이터를 통해 점과 점 사이의 상대 좌표와 각도를 통해 계산, 학습된 제스처와 맞는 결과를 도출 한다.

3. Ros pub을 통해 결과값을 송출한다.

- UI control

Qt 프레임워크를 활용하여 버튼 입력을 인식하여 ros topic으로 특정 메시지를 전달하여

로봇을 제어한다.

UI로 보내는 ROS topic은 다음과 같다

ROS publisher

/MR_2/waypoint : 저장된 waypoint로 로봇을 이동 시킵니다

/MR_2/set_waypoint : 로봇의 현재 위치를 waypoint로 저장합니다.

/MR_2/teleopkey : 로봇의 기본적인 구동을 합니다.(RC카 모드)

ros subscriber

/MR_2/TP_state : 손동작 인식 결과를 받고 알림창을 띄웁니다.

○ 프로그램 사용법 (Interface)

로봇 세팅

- 맵 생성 (slam)

Launch 파일 실행

Robot : `roslaunch slam bringup.launch`

Teaching pad : `roslaunch MR_2_ui ui.launch`

Server : `roslaunch slam gmapping.launch`

로봇을 사용할 장소에서 Teaching pad를 이용해 맵을 그린 뒤 저장

- 로봇 실행(navigation, 손짓인식)

Launch 파일 실행

Robot : `roslaunch slam bringup.launch`

Teaching pad : `roslaunch MR_2_ui ui.launch`

Server : `roslaunch MR_2_core MR_2_core.launch`

위 조건이 맞춰지면 아래와 같은 기능을 사용할 수 있다.

로봇 사용

사용시 로봇의 몸체 위에 놓여있는 티칭패드를 이용하여 사용하는 것과 징패드를 이용한 모션인식으로 사용하는 것 두가지 방법이 존재한다. 티칭패드를 사용할 경우 티칭패드 ui의 waypoint를 클릭하면 로봇은 waypoint에 저장된 지점까지 자율주행을 한다. 징패드를 이용할 경우 징패드를 한번 눌러준 뒤 보낼 신호를 로봇에게 인식시키면 작동하고 이때 징패드를 4번 누를 경우 서버PC에 메시지를 보내고 지정된 장소로 로봇이 이동한다. 로봇이 출발할 때 징패드가 2번 울리며 도착 직전에 징패드가 3번 울려 사용자에게 이동과 도착을 알린다.

○ 개발환경 (언어, Tool, 사용시스템 등)

- ui 개발

- 운영체제 ubuntu 18.04
- 언어 cpp
- 사용한 Tool - QT Creator / catkin / ros /
- 라이브러리 - QT
- 개발 보드 라즈베리 파이 4

-ROS 개발

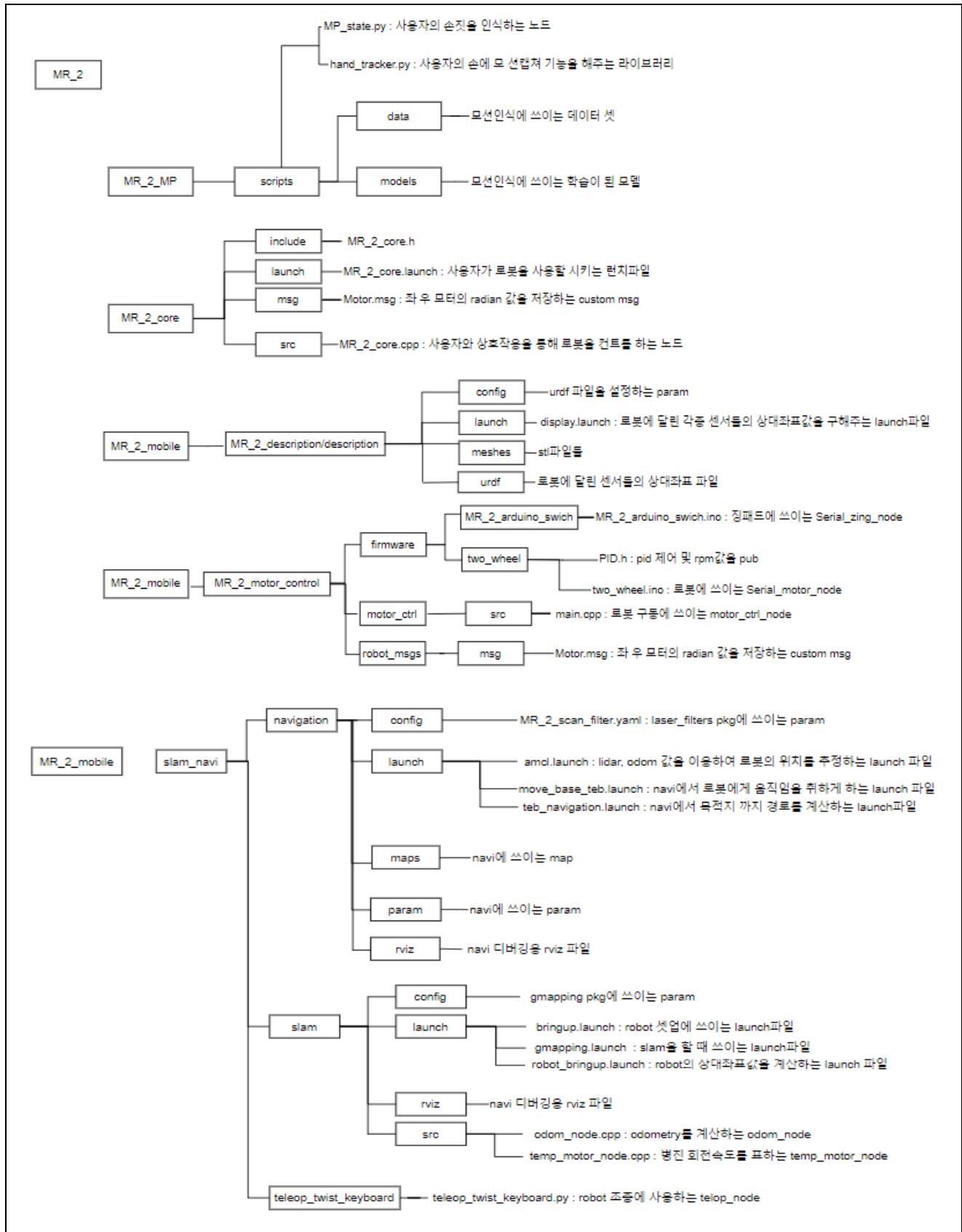
- 운영체제 ubuntu 18.04
- 언어 cpp, python
- 사용한 Tool - Ros pkg / gmapping / tensorflow / tf / navigation / ydlidar_ros
/ teb_local_planner / cv_bridge / usb_cam / laser_filters / rospy
/ roscpp / robot_state_publisher

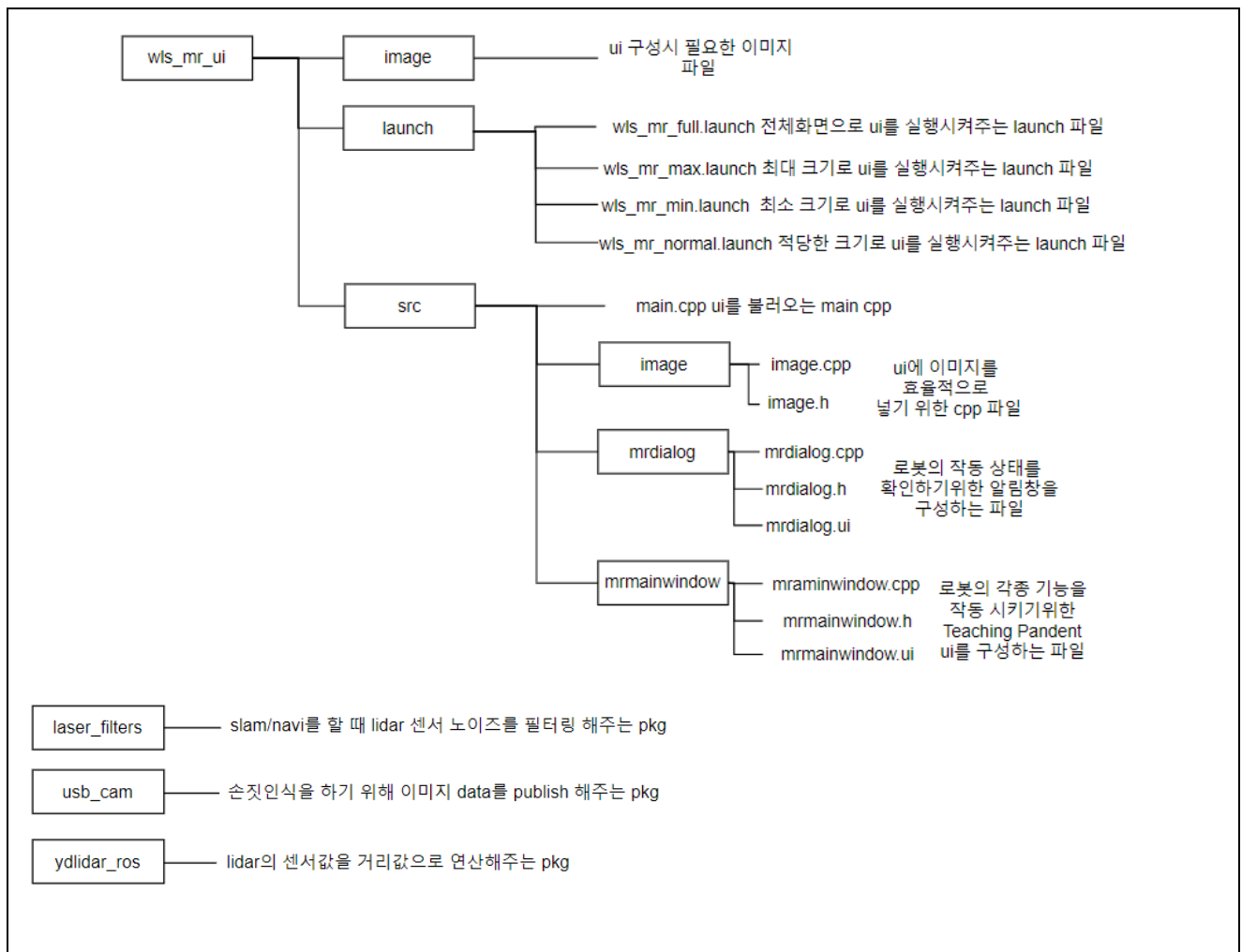
- 라이브러리 - QT / opencv2

- 개발 보드: 라즈베리파이 4, 젯슨나노, 아두이노(nano, mega)

□ 개발 프로그램 설명

○파일구성





○ 함수별 기능

- MR_2_MP - 손짓 인식

패키지명	파일명	함수명	기능
MR_2_MP	MP_state.py	transform_frame()	이미지 데이터 형 변환
		img_to_cv2()	Usb_cam_node에서 받은 img값 cv2 데이터 형 변환
		get_estimator()	사용자의 손짓 평가
		num_to_gesture()	0, 1, 2로 들어온 결과값을 toilet, hungry, thirsty로 형 변환
		play_loop()	손짓 인식 및 결과값 publish

- MR_2_mobile

파일 및 노드명	함수명	기능
Two_wheel.ino /Serial_motor_node	Setting()	Ros통신과, 아두이노 메가의 pin들을 세팅함.
PID.h	messageCb()	/MR_2/speed_set인 토픽으로부터 RPM값이 들어올 때마다 실행되며, 들어온RPM 값을 좌, 우 모터의 목표 속도 값에 저장함
	Enc_ISR()	모터의 엔코더 값이 변할 때 마다 실행되며, 엔코더 값을 카운트 함.
	vel_PID()	목표 속도 값을 입력 받으면 PID제어를 통해 RPM 값을 조정함.
	RPM()	타이머 토글이 작동할 때 마다 실행되며, 토글의 주기와 엔코더 값을 이용해 좌, 우 모터의 RPM을 계산하고, 그 값을/Motor/speed의 토픽명으로 publish함
	domotor()	모터의 방향, 속도 값을 입력 받고, 속도와 방향을 제어함.
	T5ISR()	10ms 마다 작동하는 토글이며, vel_PID()에 목표 속도를 명령함.
MR_2_arduino.ino /Serial_zing_node	messageCb()	/MR_2/arduino/zing인 토픽으로부터 정수 값이 들어올 때마다 작동하며, 입력 받은 정수 값에 따른 각기 다른 패턴의 진동을 울림
	Loop()	징패드의 버튼이 눌린 횟수를 카운트하고 /MR_2/arduino/swich의 토픽명으로 눌린 횟수를 publish함

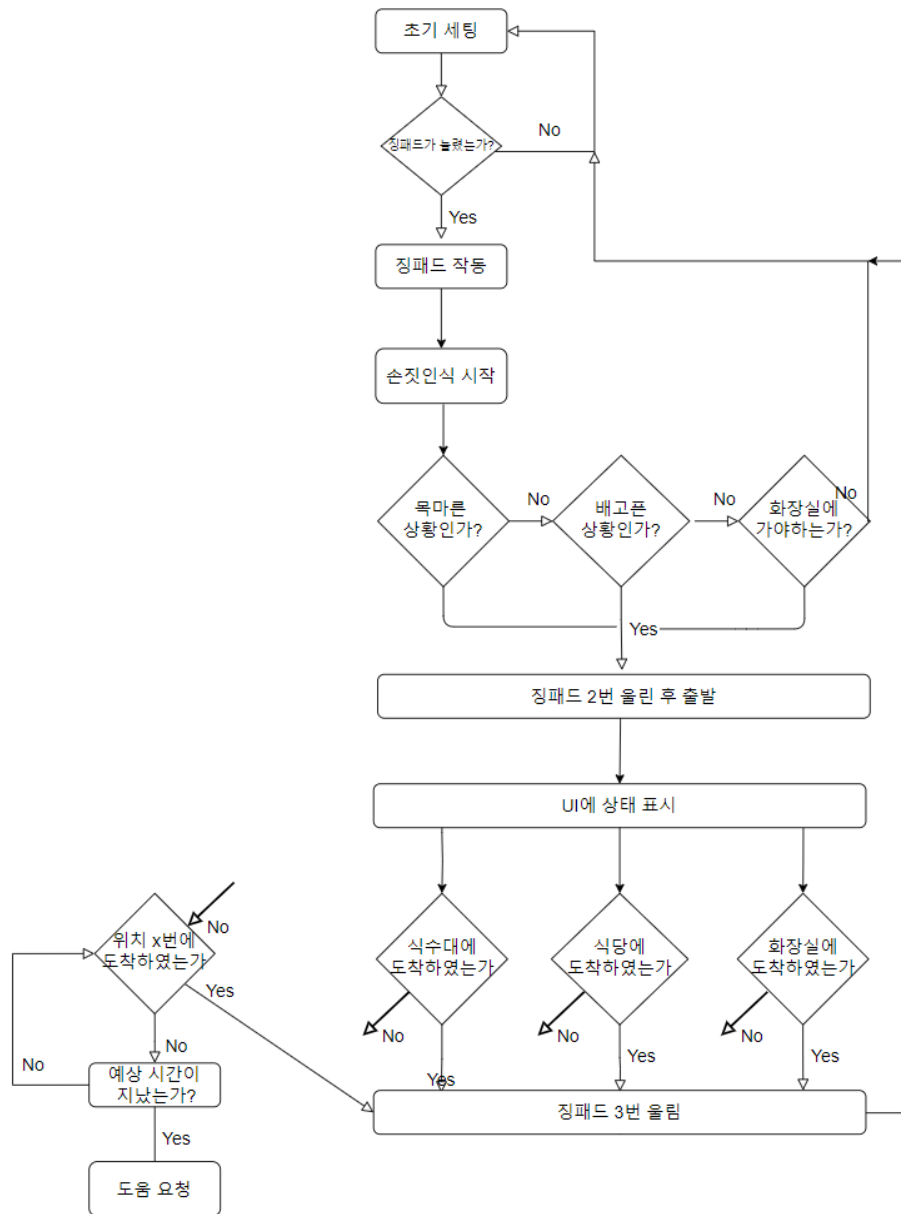
패키지명	파일 및 노드명	함수명	기능
motor_ctrl	main.cpp /motor_ctrl_node	CallBack_1()	/Motor/speed인 토픽으로부터 RPM값이 들어오면 실행되며, 들어온 RPM값을Radian으로 변환 후 로봇의 병진, 회전속도 계산 후 그 값을 /MR_2/cmd_vel의 토픽명으로 publish함
		CallBack_2()	/cmd_vel인 토픽으로부터 병진, 회전속도 값이 들어오면 Radian값을 RPM값으로 변환 후 그 값을 /MR_2/speed_set의 토픽명으로 publish함
slam	odom_node.cpp /odom_node	twistCB()	MR_2/cmd_vel인 토픽에서 메시지를 받아올 때마다 실행되며, 병진, 회전속도 값이 들어오면 변수에 저장 후 토글이 켜진다.
		main()	토글이 켜지면 변수에 저장된 병진, 회전속도를 이용하여 odometry를 계산하고, 그 값을 MR_2/odom의 토픽명으로 publish함.
	Temp_motor_node.cpp /Temp_motor_node	updateTargetVel()	MR_2/cmd_vel인 토픽으로부터 병진, 회전속도 값이 들어오면 변수에 저장 후 토글이 켜진다
		main()	토글이 켜지면 변수에 저장된 병진, 회전속도 값을 MR_2/odom_vel의 토픽명으로 publish함.

- MR_2_core

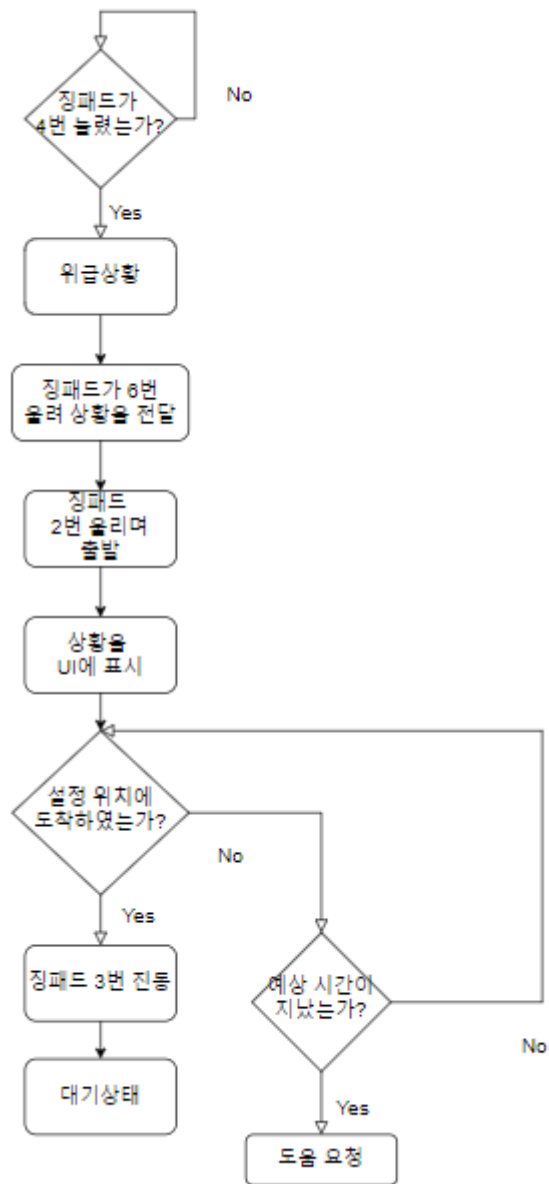
패키지명	파일 및 노드명	함수명	기능
MR_2_core	MR_2_core.cpp /MR_2_core_node	sub_MP_state_callback()	손짓이 인식될 때 마다 실행되며, 징패드와 손짓인식을 통해 로봇의 움직임을 결정함.
		sub_swich_callback()	징패드의 버튼이 눌릴 때 마다 실행되며, 4번 눌릴 시 긴급 상황실로 로봇이 인도함 .
		sub_amcl_pose_callback()	로봇이 움직일 때마다 실행되며, 로봇의 현 위치좌표 값을 저장함.
		sub_waypoint_callback()	티칭패드에서 waypoint 버튼이 눌릴 때 마다 실행되며, waypoint변수에 저장된 좌표 값으로 로봇이 이동함.
		sub_set_waypoint_callback()	티칭패드에서 setwaypoint 버튼이 눌릴 때 마다 실행되며, 눌리는 순간의 로봇 위치를 waypoint 변수에 저장함.
		sub_teleopkey_callback()	티칭패드에서 조향 버튼이 눌릴 때마다 실행되며, 로봇의 병진, 회전속도를 조작함.
		go_waypoint()	좌표값을 입력 받으면 x, y, w 값에 맞는 장소로 로봇이 이동함.

○ 주요 함수의 흐름도

MR_2_core.cpp flow chart



MR_2_core.cpp 위급상황 flow chart



○ 기술적 차별성

- 도착 인식
- 시뮬레이션이 아닌 현실 세계에서 로봇을 구동 할 때
로봇이 도착 지점까지 주행을 완료했더라도 로봇의 좌표값이 목표와 0.1%정도 오차가 생겨서 주행 완료가 되지 않는 문제가 있었다.
따라서 그런 부분을 해결하기 위해 목표의
xy값의 소수점 첫번째 자리 이하는 반올림 처리하였으며
w값은 소수점 두번째 자리 이하는 반올림 처리 하였더니
로봇이 정상적으로 주행완료 하는 것을 알 수 있었다
- 징 패드를 이용한 로봇과 사용자 사이의 링크 시스템
-
- 징 패드는 오직 촉각만을 느낄 수 있는 시청각 장애인들을 위해 로봇과 사용자 간의 링크 시스템으로서 제작되었으며, 이를 통해 시청각 장애인 사용자가 실시간으로 로봇의 상태를 사용자에게 전달하고, 로봇을 제어할 수 있게 된다.
- UI를 통해 로봇의 간단한 제어 및 waypoint를 저장하여 로봇을 제어할 수 있다.

□ 개발 중 발생한 장애요인과 해결방안

○ HW 시행착오

- 여러 waypoint를 생성하여 연속적으로 로봇을 이동시키는 테스트를 하던 중 점점 로봇이 자신의 위치를 잃어 결국 시스템이 다운되어 멈춰버리는 문제가 있었다. 디버깅 프로그램인 rviz를 통해 문제를 확인한 결과 높은 라이더의 위치와 하단 구동부의 균형이 맞지 않아 로봇이 오투기처럼 흔들리는 것이 문제인 것을 확인하였으며, 이를 라이더의 높이를 낮추고 캐스터 바퀴에 브라켓을 제작하는 것을 통해 이를 해결하였다.

○ 실시간 장애물 회피

- 기존 자율주행 모바일 로봇에 쓰이는 경로 생성 pkg는 비교적 연산이 가벼운 dwa_local_planner 패키지를 쓴다. 이는 비교적 로봇과 떨어져 있는 장애물들은 잘 피해가지만 예측할 수 없는 장애물이 등장하는 경우에는 피하지 못하거나 시스템이 다운되는 문제점이 있다는 것을 확인하였으며, teb_local_planner라는 패키지를 사용하여 해결하였다. teb 패키지는 기존 dwa 패키지가 한 개의 경로를 생성하는 것과 3~5개의 경로를 생성하여 예측할 수 없는 장애물이 등장 하더라도 시스템이 꺼지지 않고 다른 최적의 경로를 이용하여 이를 해결한다.

○ 도착 인식

- 현실 로봇을 가동 시 목표지점에 도착했음에도 도착 완료가 되지 않는 문제가 있었다. x, y, w 값을 완전히 정확하게 맞추어 지지 않는 것이 원인 이었으며 목표 값과 로봇의 위치 값이 유의미한 차이가 있지 않는 부분의 경우 로봇의 위치에서 반올림하여 처리 하였다.

○ SBC(젯슨나노)의 한계

- 손짓인식이나 slam/navi와 같은 기능들을 SBC에 올려 테스트 주행을 한 결과 무거운 연산 때문인지 발열로 인하여 5분마다 꺼지고, 제대로 된 성능이 나오지 못하는 것을 확인할 수 있었다. SBC를 성능이 뛰어난 다른 제품으로 바꿔도 되지만 비용, 시간상의 문제로 다른 방법을 찾아야 했고, 결국 서버를 따로 두어 slam/navi, 손짓인식과 같은 무거운 연산을 처리하도록 하였고, SBC에는 연산이 가벼운 센서 값(모터, 라이더, 카메라)을 1차적으로 처리하고 서버에 송출하는 방식으로 해결하였다.

□ 개발결과물의 차별성

○ 노령자용 실내 안내 로봇

- 징패드

노령자용 실내 안내 로봇 은 사용대상이 우리와 같은 시청각장애인이 아닌 고령의 비장애인을 대상으로 하였기 때문에 이동하면서 시각적으로 사용자에게 길을 안내하나 제작한 로봇은 시청각장애인을 대상으로 생각하여 위와 같은 안내방식이 아닌 자체 제작한 징패드를 이용하여, 진동과, 로봇과 연결된 줄을 통해 사용자와 링크되는 방식으로 인도한다.

- 동작 인식

사용자의 동작을 인식하는 것 비장애인 같은 경우 다양한 시각정보 와 소리를 들으며 원하는 곳을 이동할 수 있고 자신이 원하는 장소를 지도에서 입력 또는 주어진 정보를 바탕으로 지정하는 것이 가능하나 대부분의 시청각장애인의 경우 특수한 상황이 아닐 경우 듣는 이가 보편적 이지 않은 축수화를 이해해야 의사를 전달할 수 있다.

이런 이유로 로봇 스스로 시청각 장애인들의 표현을 알아내는 방법은 그들의 수화를 비전기술로 인식하거나 또는 그들이 하는 수화를 촉각으로 인식해야 한다.

로봇은 수화와 같이 손짓을 인식하여 사용자의 의사를 전달받고 그에 맞는 진동을 통해 사용자에게 정보를 전달해줄 수 있다.

□ 개발 일정

No	내용	2021年											
		6月			7月			8月			9月		
1	주제선정												
2	프로토타입 제작												
3	Slam/navi 테스트												
4	로봇제작												
5	손짓인식												
6	UI 제작												
7	SW 통합												
8	테스트 및 성능개선												

□ 팀 업무 분장

No	구분	성명	참여인원의 업무 분장
1	팀장	윤우성	Robot system 구축 및 손동작 인식 알고리즘 구현
2	팀원	김건우	UI 프로그램 제작 및 회로 및 배선 작업, 동영상 편집
3	팀원	유창민	기구 설계 및 제작