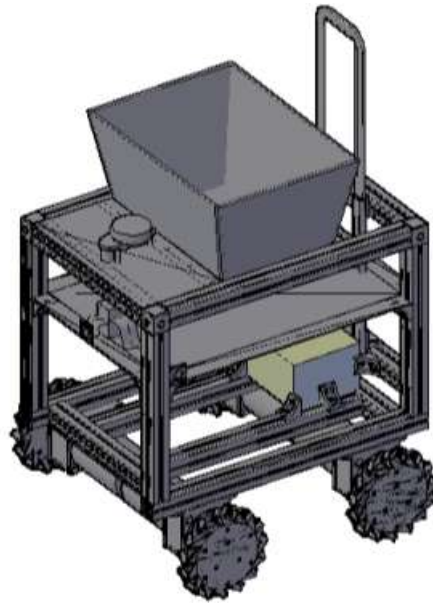


REPORT



[UMM: Useful Market Mobility]

과 목 명 : 기계공학종합설계 1
담당교수 : 조구영 교수님
소 속 : 공과대학 기계공학과
기 간 : 2023.09.06. ~ 2023.11.30.
조 장 : 32194885 한준모
조 원 : 32191618 박병춘
32195118 장한솔
32192854 윤 상
32194950 홍기원



단국대학교
Dankook University

목차

I. 서론

1. 주제	8
1.1 주제 선정 이유	9
1.2 인터뷰 정리	9
1.3 특허 조사	10
1.3.1 시각장애인 길 안내를 위한 지광게이션 장치	10
1.3.2 레이저를 이용한 거리 측정 장치 (시각장애인 안경)	11
1.3.3 시각장애인을 위한 음성 유도기	11
1.3.4 시각장애인 보행 안내 시스템 및 방법	11
1.3.5 RFID/NFC 태그에 고유번호 및 문자를 부여한 알고리즘 점자블록 시각장애인 안내유도 시스템 및 안내방법	12
1.3.6 실시간 스트리밍을 이용한 시각장애인 안내서비스 제공 시스템	13
1.3.7 자율주행 로봇의 충돌 회피 방법 및 월드 맵 관리 방법	13
1.4 시장 조사	14
1.4.1 자율주행 로봇 시장의 성장	14
1.4.2 서비스 로봇의 성장	14
1.4.3 시각장애인에 대한 수요	15
1.4.4 히어링 봇(hearing bot)	16
1.4.5 '삼성전자'의 Bespoke 제트 봇 AI	16
1.4.6 장애인의 쇼핑 시 불편한 점 질의	17
1.4.7 한국소비자원의 설문조사	17
1.5 설계 일정 및 업무 분담	18
1.5.1 제작 일정 (Gantt Chart)	18
1.5.2 업무 분담	19

II. 본론

2. 개념 설계	19
2.1 Brain Storming	19
2.2 제품의 기능 및 목적(구상단계)	20
2.3 설계 Concept	21
2.3.1 알고리즘	21
2.3.2 제품 설명	21
2.3.3 장바구니 최대 무게 계산	22
3. 기계 요소 상세설계	23
3.1 구동장치 선정	23
3.1.1 바퀴 선정	23
3.1.2 변수 정의	25
3.1.3 자유물체도	26
3.1.4 모터의 토크 계산	26

3.1.5	요구동력 계산	26
3.1.6	모터 선정	27
3.1.7	허브(HUB) 베어링 선정	28
3.1.8	배터리 선정	29
3.2	메카닉 휠을 이용한 전방향 주행형 AGV 구성원리 및 기구학적 모델링	30
3.2.1	전방향 주행형 AGV의 구성	30
3.3	전방향 주행형 AGV의 모델링	30
3.3.1	가정	31
3.3.2	전방향 주행형 AGV의 기구학적 모델링	31
3.4	선정 기본 사양	34
4.	프레임 설계 및 체결	35
4.1	프로파일 선정	35
4.2	프레임 체결방식 선정	36
4.2.1	프로파일 2개 조립	36
4.2.2	프로파일 3개 조립	38
4.3	프레임 안정성 계산	39
4.3.1	위층 프레임 처짐 계산	39
4.3.2	기둥 프레임 좌굴 해석	43
4.3.3	아래층 프레임 처짐 계산	45
4.4	Fusion 360을 이용한 전체 프레임 해석	48
5.	제품 상세 구동 알고리즘 설계	49
5.1	알고리즘 순서도	50
5.2	제어부	51
5.2.1	바퀴(PID) 제어	51
5.2.2	SLAM	51
5.2.3	SLAM의 문제점 및 해결책	52
5.2.4	SLAM 참고코드	58
5.3	통신부	60
5.3.1	제품 알고리즘 개략도 변경하기	60
5.3.2	음성 인식 과정의 블록코딩	60
5.4	센서부	61
5.4.1	RFID 작동 원리	61
5.4.2	LIDAR 작동 원리	61
5.4.3	카메라 1(전면) 작동 원리	61
5.4.4	카메라 2(측면) 작동 원리	66

6. 변경사항	
7. 기타 설계	69
7.1 통신 전용 어플 설계	69
7.1.1 어플리케이션 제작 의도	69
7.1.2 어플리케이션의 역할	69
7.1.3 어플리케이션 개발 과정	69
7.2 배터리 잔량 확인 기능	70
7.2.1 기능의 필요성	70
7.2.2 기능의 상세 설계	71
8. 모빌리티 구동 S/W 설계	72
7.1 회로도(연결 방식) 설계	70
7.2.1 모빌리티 구동 코딩	70
7.2.2 코드 부가 설명	70
III. 결론	
9. 최종 제품 사양서	73
9.1 제품 사양	73
9.2 제품 성능	73
9.3 제품 기능	74
9.4 단계별 달성 목표	74
10. 경제성 평가	76
10.1 추가 구매	
11. 추후 계획	
12. CAD 도면	83
13. 회의록	91

그림 차례

Fig. 1 지광게이션 장치 원리	10
Fig. 2 안경용 거리측정장치 원리	11
Fig. 3 점자 블록 및 RFID를 이용한 안내유도 시스템	12
Fig. 4 실시간 스트리밍을 이용한 시각 장애인 안내시스템	13
Fig. 5 로봇의 충돌 회피방법	14
Fig. 6 2030 자율시장 로봇 시장의 규모 예측	15
Fig. 7 서비스 로봇의 시장가치 상승	15
Fig. 8 Hearing Bot의 팜플렛	16
Fig. 9 비스포크 제트봇 AI 제품	16
Fig. 10 쇼핑 앱의 시각장애인 접근성 실태	17
Fig. 11 Gantt Chart	18
Fig. 12 Brainstorming mind map	19
Fig. 13 아이디어 스케치(1)	20
Fig. 14 아이디어 스케치(2)	20
Fig. 15 본 모빌리티에 사용할 장바구니 규격	22
Fig. 16 쇼핑카트 제원	23
Fig. 17 메카닉 휠 도면	24
Fig. 18 메카닉 휠 스펙 선정	24
Fig. 19 메카닉 휠을 이용한 주행 및 회전	24
Fig. 20 메카닉 휠 미끄럼 현상을 이용한 진행 방향 설정	25
Fig. 21 모빌리티의 자유물체도와 운동역학 선도	26
Fig. 22 IG-42GM+ Encoder 01 TYPE DC 12V 모터	27
Fig. 23 DC모터 선정 및 스펙, 도면	28
Fig. 24 허브(HUB) 베어링 도면	28
Fig. 25 구동 장치 연결도	29
Fig. 26 본 모빌리티의 전방향 주행형 AGV	30
Fig. 27 메카닉휠의 회전좌표계를 이용한 진행방향, 속도 계산	31
Fig. 28 모빌리티 중심에 대한 각 휠의 선속도 벡터도	33
Fig. 29 알루미늄 프로파일 선정 및 스펙	35
Fig. 30 프로파일 두 개 조립 시 필요부품	36
Fig. 31 조인트 브라켓 체결 방식	37
Fig. 32 프로파일 2개 체결한 모습	37
Fig. 33 프로파일 3개 조립시 필요부품	38
Fig. 34 프로파일 3개 체결한 모습	38
Fig. 35 모빌리티 가장 위쪽 프레임 평면도	39
Fig. 36 1번 프레임의 Free-Body Diagram	39
Fig. 37 4번 프레임의 FBD	40
Fig. 38 장바구니 분포하중 처짐 계산	41
Fig. 39 가장 위쪽 프레임 처짐 분포	42
Fig. 40 가장 위쪽 프레임 변형 분포	42

Fig. 41 기동 프레임의 좌측면도	43
Fig. 42 기동 프레임의 정면도	43
Fig. 43 프로파일 기동부분에 대한 처짐 분포	44
Fig. 44 프로파일 기동 부분에 대한 응력 분포	44
Fig. 45 모빌리티의 아래 프레임 평면도	45
Fig. 46 1번 프레임의 Free-Body Diagram	45
Fig. 47 2번 프레임의 Free-Body Diagram	46
Fig. 48 2번 프레임의 처짐 그래프	47
Fig. 49 아래층 프레임 처짐 분포	48
Fig. 50 아래층 프레임 변형 정도	48
Fig. 51 프레임 처짐 분포 표현(Fusion 360)	49
Fig. 52 프레임 변형 분포 표현(Fusion 360)	49
Fig. 53 프레임 응력 분포 표현(Fusion 360)	50
Fig. 54 알고리즘 순서도	51
Fig. 55 Yolo v3 사용 시 모빌리티 구동 시나리오	52
Fig. 56 Yolo v3 미사용 시 모빌리티 구동 시나리오	52
Fig. 57 SLAM 처리 흐름도	53
Fig. 58 SLAM 코드	55
Fig. 59 cd.py 파이썬 코드	57
Fig. 60 move.launch 런치파일	58
Fig. 61 publisher.py 파이썬 코드	58
Fig. 62 camera.py 파이썬 코드	59
Fig. 63	55
Fig. 64 선정값 ($K_p=0.5$ $K_d=1.4$) 에 대한 응답 선도	63
Fig. 65 PD 제어 응답 선도 (1)	63
Fig. 66 PD 제어 응답 선도 (2)	64
Fig. 67 제품 구동 과정을 간략하게 표현한 개략도	64
Fig. 68 어플 실행 화면(좌), 파일 코너 음성이 입력된 화면(우)	68
Fig. 69 RFID 작동 구조도	70
Fig. 70 상대적 좌표를 알기 위해 측정해야 할 파라미터	73
Fig. 71 파라미터를 통해 얻을 수 있는 결과값	74
Fig. 72 소형 장애물의 크기와 로봇의 위치를 추출하는 코드	75
Fig. 73 모빌리티의 장애물을 피하는 알고리즘 좌표 설정	76
Fig. 74 측면 카메라를 이용한 마트 물품 인식	77
Fig. 75 모터 배선도	86
Fig. 76 RFID와 블루투스 연결 선도	86
Fig. 77 주요 부품 간 연결 방식 정리	87
Fig. 78 [목표 1] 달성 과정	98
Fig. 79 [목표 2] 달성 과정	99
Fig. 80 [목표 3] 달성 과정	101
Fig. 81 [목표 4] 달성 과정	102
Fig. 82 [목표 5] 달성 과정	103
Fig. 83 시각 장애인 증가 추세	105

Fig. 84 테크 이노베이션 사의 “ Innomake ”	105
Fig. 85 “전국학생설계경진대회” 설계제안서	106
Fig. 86 구동부분 조립도 1	107
Fig. 87 구동부분 조립도 2	107
Fig. 88 구동부분 조립도 3	108
Fig. 89 LIDAR 센서 위치 및 모습	108
Fig. 90 바구니 장착 시 모습 1	109
Fig. 91 바구니 장착 시 모습 2	109
Fig. 92 모빌리티 전체모습 1	110
Fig. 93 모빌리티 전체모습 2: 정면도	110
Fig. 94 모빌리티 전체모습 3: 우측면도	111
Fig. 95 모빌리티 전체모습 4: 평면도	111
Fig. 96 배터리 체결방식 및 위치 1	112
Fig. 97 배터리 체결방식 및 위치 2	112
Fig. 98 카메라 모듈 체결방식 및 위치	113
Fig. 99 2D 우측면도	113
Fig. 100 2D 정면도	114
Fig. 101 2D 평면도	114

표 차례

표. 1 업무 분담표	19
표. 2 마트 전용 자율주행 봇의 기능	21
표. 3 IG-42GM+ Encoder 01 TYPE DC 12V 모터의 제원	27
표. 4 STX30L-BS 배터리 제원 표	30
표. 5 배터리의 규격	30
표. 6 회전 좌표계 매개변수	32
표. 7 모빌리티 기본 사양	34
표. 8 ydlidar x2 slam	60
표. 9 Gmapping, Catographer 비교	60
표. 10 PD제어 코드	62
표. 11 아두이노 통신을 위한 블록 코딩	67
표. 12 어플 실행 버튼	69
표. 13 RFID 및 BLUETOOTH 아두이노 코드	72
표. 14 설계 변경 사항 (1)	79
표. 15 설계 변경 사항 (2)	79
표. 16 설계 변경 사항 (3)	80
표. 17 설계 변경 사항 (4)	81
표. 18 제작 과정	83
표. 19 최종 모빌리티 형상	83
표. 20 모터 구동 아두이노 코드	92
표. 21 최종 제품 사양서	95
표. 22 제품 성능표	95
표. 23 제품 기능표	96
표. 24 달성 목표 작성표	97
표. 25 제작 비용 정리	104

I. 서론

1. 주제

1.1 주제 선정 이유

설계 주제로 사회적 약자를 돕는 모빌리티를 선정하고자 하였다. 요즈음 떠오르는 자율주행과 접목시키고자 하여 시각장애인을 대상으로 하고자 하였다. 시각장애인의 평소 생활에서의 불편한 점을 조사하던 중 시각장애인이 물건을 구매하는데 어려움을 겪는다는 것을 알게 되었다.

시각장애인들의 어려움을 생각해보았다:

- 물품을 혼자 구매를 하는데 있어 물품이 보이지않아 위치 파악의 어려움.
- 물건의 위치를 알더라도 상세정보를 알 수 없음.
- 점자를 고안하더라도 점자를 읽을 수 있는 시각장애인들은 전체 인원의 7% 수준
- 온라인 구매 또한 대부분 시각적인 정보들인 그림과 글로 이루어져 있어 불가능.

또한 특정 공간에서 도움을 주고자 하여, 시각장애인이 접근하기 어려운 공간으로 선정하고자 하였다. 이 과정에서 마트를 선정하게 되었는데, 마트에서 장을 볼 때 조원 중 아무도 시각장애인이 장을 보는 것을 본 적이 없음을 알 수 있었다. 이는 시각장애인이 장을 보기 매우 불편하여 도와주는 사람이나 기계가 없다면 접근을 하지 않는 것으로 생각할 수 있다.

따라서 주제를 ‘시각장애인을 위한 마트 전용 자율주행 봇’으로 선정하였다.

모빌리티 이름은 Useful Market Mobility, ‘UMM’으로 선정하였다.

1.2 인터뷰 정리

한국시각장애인연합회¹⁾ 중도시각장애인지원센터 소속이신 천상미 분의 인터뷰를 통해 모빌리티의 필요성을 확인하고자 하였다. 시각장애인의 불편함을 알고자 직접 시각장애인께 인터뷰를 통해 질의응답을 하였다.

한국 시각 장애인 협회
중도 시각 장애인 센터 천상미

Q. 사회적 약자들이 느끼는 불편함을 조사하는 중, 시각 장애인들이 편의점이나 마트 등에서 장을 볼 때 진열된 물건에 코 앞까지 다가가셔서 어떤 물건인지 판단하며 장을 보는 경우를 보았다. 실제로 그런 사례가 많은지

A. 실제로 직접 장을 보게 된다면 거의 그런 식이다. 흔히들 시각 장애인은 세상이 까맣게 보인다고 생각하시는데, 그런 경우도 있지만 대부분의 경우는 눈이 굉장히 안좋아서 사물의 구분이 불가능한 분들이다. 따라서 말씀 해 주신 것처럼 코 앞까지 가는 경우에 물체의 형태나 색깔 등을 파악 할 수는 있지만 그럼에도 어떤 물건이라고 확실하게 단정짓기는 사실 힘든 부분이 많다. 그래서 실제로 장을 보는 경우는 거의 없고 생필품 같은 경우는 보통 쿠팡을 이용해서 구매한다.

1) <http://www.kbuwel.or.kr/>

Q. 쿠팡을 이용해서 생필품들을 구매 한다고 하셨는데, 실례가 안된다면 쿠팡을 이용할 때는 불편함을 느끼시는 점이 없는지 ?

A. 쿠팡으로 주문하는 경우에는 핸드폰과 쿠팡에서 지원하는 음성 출력 기능을 통해 내가 찾고자 하는 제품을 담는 과정은 다른 구매 경로에 비교하면 비교적 쉬운 편이다. 다만 거기까지 가는 과정에서 접근성 권한을 부여한다거나 어플을 실행시킨다거나 하는 데에 다소 어려움이 있다. 다만 실제로 장을 보는 것에 비하면 편한 것은 사실이다.

Q. 사전에 말씀 드린 것 처럼 저희가 시각 장애인 분들이 장을 볼 때 길을 찾거나, 물건을 찾는 것을 보조 해 주는 기계를 만드려고 하는데, 실제로 이런 게 있다면 불편함이 해소 될 것 같은지

A. 장을 봐야 하는 상황에서는 그런 보조 기계가 있다면 확실히 편해질 것 같다. 다만 말씀 드린 것 처럼 보통은 쿠팡을 통해 해결하고 실제로 장을 보러 가는 경우가 적다. 계속 말이 나왔던 것처럼 코 앞까지 가서 봐야 형태를 겨우 인식 할 수 있는 정도라 일반인에게 보이는 눈치 때문에 자존감이 떨어져 장을 보러 하는 상황이 잘 없다.
다만 말씀 해 주신 것과 같은 보조 기구들이 나올수록 점점 아웃도어에서의 자유로운 생활이 가능 해 질 것 같은 기대는 있다. 여러분들의 일상 생활을 생각해 보시면 십년 전과 비교해 봤을 때 시각 장애인을 거리에서 보는 빈도가 많이 줄었을 것이다. 말씀 해 주신 마트같은 경우에는 거의 없으실 거라고 생각한다. 결과가 기대만큼 나오면의 이야기지만, 다양한 보조 기구들이 늘어 나 생활 범위가 확대 될 수 있다면 우리들에게도 굉장히 긍정적인 변화가 될 거라고 생각 한다.

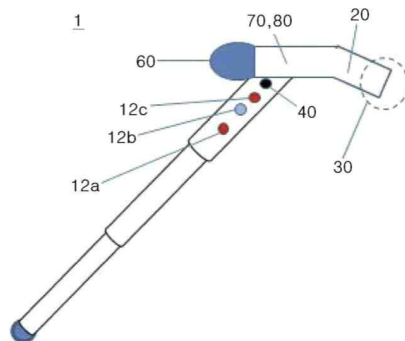
1.3 특허 조사

1.3.1 시각장애인 길 안내를 위한 지팡게이션 장치

그림과 같이 층상형 적외선 센서부로 장애물과의 거리, 장애물의 종류를 정확하게 인식할 수 있어 목적지까지의 정확한 길안내와 장애물의 정보를 제공할 수 있다.

출원 일자	2013.04.30
등록 일자	2014.11.03
출원 번호	10-2013-0048023

도면3



도면4

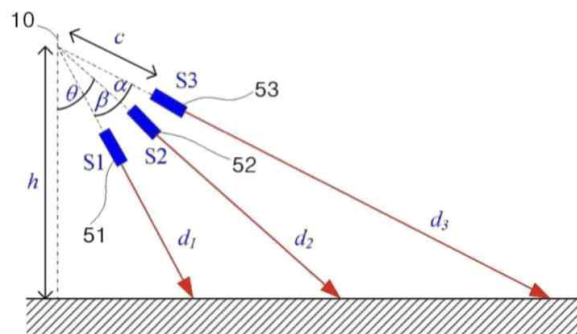


Fig. 1 지팡게이션 장치 원리

1.3.2 레이저를 이용한 거리 측정 장치 (시각장애인 안경)

바라보는 방향으로 레이저가 조사된 후, 피사체에 의해 반사된 반사광이 돌아오는 시간을 측정함으로써 피사체까지의 거리를 측정할 수 있다.

출원 일자	2009.12.11
등록 일자	2011.06.17
출원 번호	10-2009-0123472

도면2

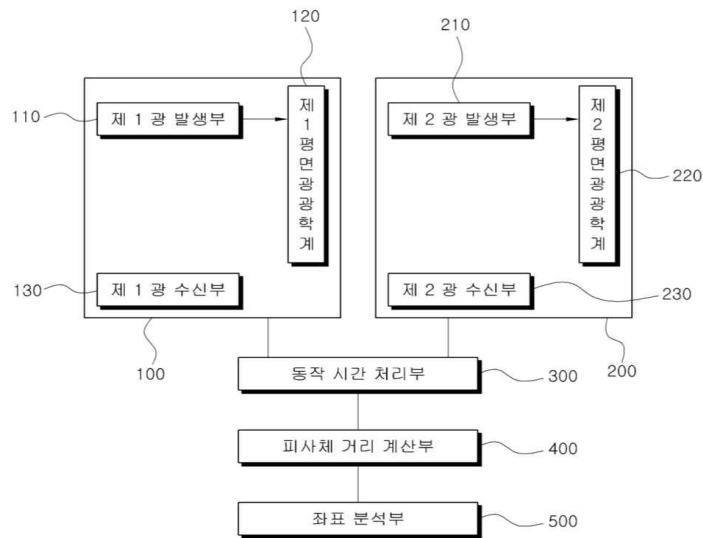


Fig. 2 안경용 거리측정장치 원리

1.3.3 시각장애인을 위한 음성 유도기

특히나 화재 발생 시에 시각장애인용 음성 유도기의 규격에 따라 시각장애인용 음성 유도기에 안내 요청 신호를 무선으로 송출하고, 이해 응하여 시각 장애인용 음성유도기가 해당 위치에서의 대피 방법 등을 음성으로 안내할 수 있도록 함으로써 시각 장애인 음성 유도기의 활용도를 극대화시킬 수 있다.

출원 일자	2011.12.14
등록 일자	2013.05.29
출원 번호	10-2011-0134677

1.3.4 시각장애인 보행 안내 시스템 및 방법

목적지를 입력하는 점자 입력부, GPS 위성으로부터 위치 정보를 수신하여 목적지의 방향 정보를 안내하는 GPS 수신기, 도로나 건물목에 설치된 RFID 태그 정보를 수신하여 도로 정보를 안내하는 RFID 리더기 및 사람이나 사물의 접근을 감지하여 보행

정보를 안내하는 초음파 센서기, 상기 방향 정보, 상기 도로 정보, 상기 보행 정보를 음성 혹은 진동으로 알려주는 출력부를 포함한다.

출원 일자	2012.11.23
등록 일자	2015.02.10
출원 번호	10-2012-0133744

1.3.5 RFID/NFC 태그에 고유번호 및 문자를 부여한 알고리즘 점자블록 시각장애인 안내유도 시스템 및 안내방법

점자블록 및 도포선에 설치된 RFID/NFC 태그를 무선(비접촉식)으로 인식하여 진행 방향을 음성으로 알려준다. 지팡이의 전후좌우 방향을 감지하여 진행방향을 음성으로 안내하는 기술이 대한민국 공개특허 10-0647069호에 개시되어 있다. 이는 시각 장애인용 보행안내시스템은 보행로에 설치된 블록에 전후좌우 진행방향의 정보를 점자블록 RFID 태그를 저장하고 장애인용 지팡이 아랫부분에 RFID리더를 장착한 후 RFID 리더의 안테나를 통해 무선신호를 방사하여 근접되는 RFID태그로부터 정보를 읽어들이 음성으로 안내하도록 되어있다.

출원 일자	2013.03.06
등록 일자	2013.12.13
출원 번호	10-2013-0024120

도면7a

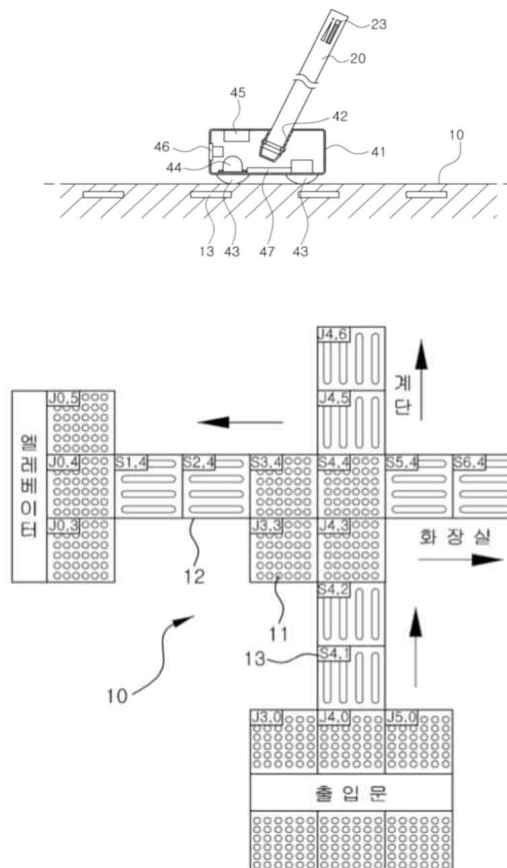


Fig 3. 점자블록 및 RFID를 이용한 안내유도 시스템

1.3.6 실시간 스트리밍을 이용한 시각장애인 안내 서비스 제공 시스템

실시간 스트리밍을 이용한 시각장애인 안내 서비스 제공 시스템이 제공되며, 설정된 방향을 촬영하는 웨어러블 기기, 안내 요청을 입력받고, 웨어러블 기기로부터 촬영된 영상을 실시간으로 업로드하는 사용자 단말, 사용자 단말의 안내 요청으로 연결되고, 웨어러블 기기로부터 스트리밍되는 영상을 화면에 출력할 때 화면 내 각 객체에 대한 방향 및 거리를 오버레이하여 출력하는 안내 단말 및 사용자 단말과 웨어러블 기기가 연동되도록 매핑 하여 저장하는 등록부, 사용자 단말에서 안내 요청을 전송하는 경우, 안내 단말과 사용자 단말 간을 연결하며 안내 단말로 웨어러블 기기에서 촬영된 영상이 스트리밍되도록 하는 실시간 중계부, 안내 단말의 화면 내 각 객체에 대한 방향 및 거리를 오버레이 되도록 표시하는 표시부, 안내 단말에서 사용자 단말의 요청에 대응하는 객체의 방향 및 거리를 안내하여 전달하는 안내부를 포함하는 안내 서비스 제공 서버를 포함한다.

출원 일자	2023.01.30
등록 일자	2023.07.21
출원 번호	10-2023-0011643

도면1

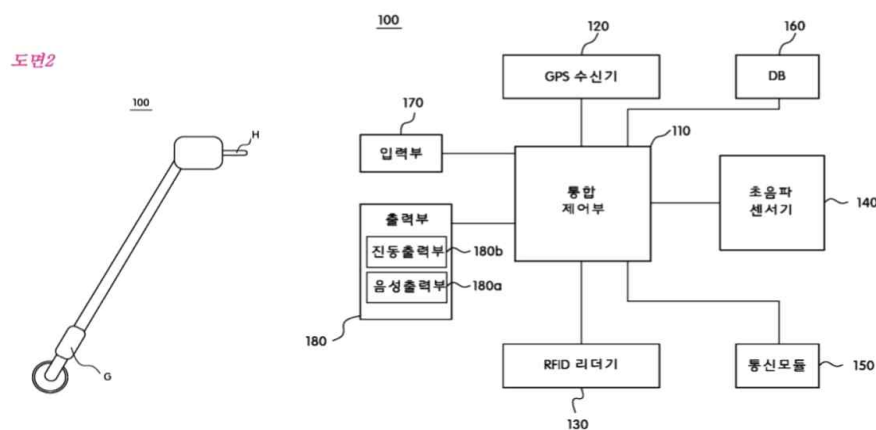


Fig 4. 실시간 스트리밍을 이용한 시각 장애인 안내시스템

1.3.7 자율주행 로봇의 충돌 회피 방법 및 월드 맵 관리 방법

작성 되어 있는 월드 맵을 기반으로 주행하고 있는 로봇이, 센서를 이용하여 받아들이는 정보를 통해 예상치 못한 장애물을 만났을 때 자율 주행을 통해 회피 할 수 있다. 이 때 이미 매핑되어 있는 월드 맵과 상호작용을 항시 유지하여 피드백을 주고 받음으로서, 실시간으로 정보를 갱신하여 정밀도를 높인다.

출원 일자	2020.10.26
등록 일자	2022.05.03
출원 번호	10-2020-0139366

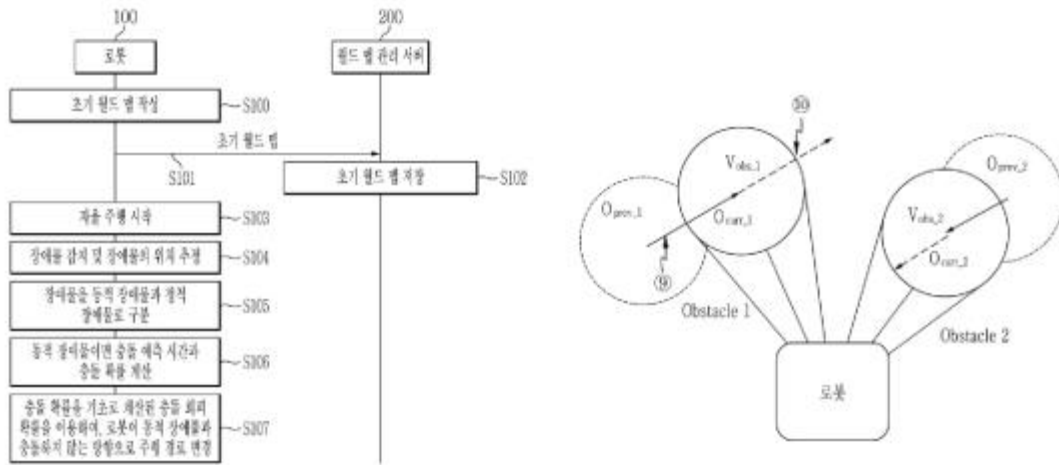


Fig. 5 로봇의 충돌 회피방법

1.4 시장 조사

1.4.1 자율주행 로봇 시장의 성장²⁾

자율주행 로봇 시장은 research&market에서 제공한 자료에 따르면 2022년 기준 30억 8000만 달러 규모에서 2030년까지 106억 6000만 달러 규모까지 성장할 것으로 예측된다고 하였다. 즉 연평균 16.8%의 성장세가 예상되는 것이다.

이처럼 로봇 공학 및 시장은 사물인터넷(IoT), 가상현실(VR), 5세대 이동통신(5G) 기술과 융합하여 미래 산업의 핵심이 될 것이란 전망이다.

1.4.2 서비스 로봇의 성장³⁾

서비스 로봇은 크게 전문 서비스 로봇과 소비자 서비스 로봇으로 구분된다. 이 중 전문 서비스 로봇은 운송 및 물류 로봇을 중심으로 급 성장하고 있다. 보스턴 컨설팅 그룹(Boston Consulting Group, BCG)은 전문 서비스 로봇이 미래 로봇 산업의 중심이 될 것이라고 분석하였다. 또한, 전문 서비스 로봇의 시장 가치가 전통적 산업 로봇 시장의 두 배 이상이 될 것으로 예상한 바가 있다.

2) <https://www.weeklytrade.co.kr/news/view.html?section=1&category=136&no=86735>

3) https://dream.kotra.or.kr/kotranews/cms/news/actionKotraBoardDetail.do?MENU_ID=170&pNttSn=198633

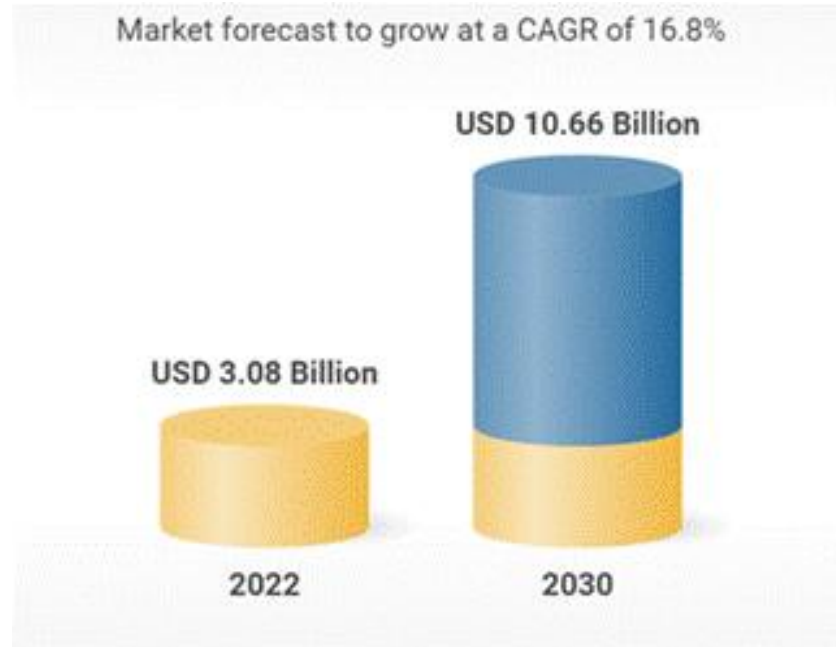


Fig. 6 2030 자율시장 로봇 시장의 규모 예측



Fig. 7 서비스 로봇의 시장가치 상승

1.4.3 시각장애인에 대한 수요⁴⁾

2023년 4월 기준 보건복지부가 발표한 자료에 의하면 전국에 등록되어있는 장애인은 265만 3000명이며 이 중 시각장애인은 9.5%를 차지한다고 한다. 즉 전국에 약 25만 2000여명이 시각에 심각한 불편함을 느끼고 있는 것이다. 이 수치로 봤을 때 시각장애인의 장보기를 도와주는 자율주행 붓은 결코 소수에 국한한 기계가 아니라는 것을 알 수 있다.

4) <https://www.futurechosun.com/archives/75130>

1.4.4 히어링 봇(hearing bot)⁵⁾

청각장애인을 위한 자율주행 봇이며 보청기를 대신해주는 스마트 홈 시스템이다. 히어링(hearing bot)은 진동 기능을 이용하여 청각장애인의 알람이 되어주는 역할과 수화를 하는 기능이 있어서 청각장애인과 비장애인간의 대화를 가능하게 해준다. (가격정보 없음.)



Fig. 8 Hearing Bot의 팸플렛

1.4.5 '삼성전자'의 Bespoke 제트 봇 AI⁶⁾

저시력자 장애인들을 위해 삼성전자에서 출시한 제품으로, 제품명암대비를 7대1로 설정하여 저시력자의 제품 인지를 쉽게 하였고, 제품자체의 화면에 안내 자막이 나오는데 저시력자인 것을 감안하여 재생속도 조절 및 또렷한 일러스트를 사용하였다고 한다. 한계가 있다면 색마저도 구별하지 못하는 시각장애인이라면 사용하기 힘들다는 점이다. 출고가는 159만 9000원이다.



Fig. 9 비스포크 제트봇 AI 제품

5) <https://www.behance.net/gallery/107405643/H-E-A-R-I-N-G>

6) <https://www.samsung.com/sec/vacuum-cleaners/jetbot-vr50t95935w-d2c/VR50T95935W/>

1.4.6 장애인의 쇼핑 시 불편한 점 질의⁷⁾

한국일보 윤한슬 기자는 시각장애인을 상대로 인터뷰 (2021년 4월 19일)를 하였다. 인터뷰 대상에게 대형마트를 이용하면서 불편한 점을 물어봤는데 인터뷰 대상자는 글자를 읽을 수 없어 제품 정보를 확인할 수 없어서 마트의 기존 정보를 알고 가지 못하면 마트 쇼핑이 힘들다고 하였다.

이러한 문제점은 시각장애인의 장보기를 도와주는 봇의 대체 텍스트(음성)를 통하여 보완할 수 있다고 생각하였다.

1.4.7 한국소비자원의 설문조사⁸⁾

한국소비자원은 2022년 소비생활과 밀접한 어플 16개의 편의제공 실태를 조사하였는데, 조사 대상 장애인들의 92.2%가 상품 정보 및 서비스 확인이 어렵다고 답변하였다. (조사대상 : 쇼핑, 배달 및 동영상 스트리밍 어플 이용 경험이 있는 시각장애인 193명) 조사 결과를 보면 시각장애인이 가장 불편함을 겪는 부분은 대체 텍스트가 주어지지 않는다는 점이다. 즉 이미지 정보나 제품 설명을 음성으로 전달해주지 못 한다는 점이다.

< 쇼핑앱의 시각장애인 접근성 실태 > (단위: %(개))

구 분		양호	미흡	해당없음	계
메인 페이지	팝업/배너	55.6(5)	44.4(4)	-	100.0(9)
	하단탭	77.8(7)	11.1(1)	11.1(1)	100.0(9)
	검색창	100.0(9)	-	-	100.0(9)
	소계	77.8(21)	18.5(5)	3.7(1)	100.0(27)
상품 페이지	상품명	100.0(9)	-	-	100.0(9)
	상품가격	55.6(5)	44.4(4)	-	100.0(9)
	상품상세정보	-	100.0(9)	-	100.0(9)
	배송정보	88.9(8)	11.1(1)	-	100.0(9)
	소계	61.1(22)	38.9(14)	-	100.0(36)
주문 페이지	주소	100.0(9)	-	-	100.0(9)
	연락처	100.0(9)	-	-	100.0(9)
	배송 시 요청 사항	88.9(8)	11.1(1)	-	100.0(9)
	소계	96.3(26)	3.7(1)	-	100.0(27)
결제 페이지	카드등록	22.2(2)	66.7(6)	11.1(1)	100.0(9)
	계좌등록	55.6(5)	44.4(4)	-	100.0(9)
	간편결제 비밀번호	66.7(6)	33.3(3)	-	100.0(9)
	최종 상품 가격	100.0(9)	-	-	100.0(9)
	소계	61.1(22)	36.1(13)	2.8(1)	100.0(36)
총계		71.4(90)	27.0(34)	1.6(2)	100.0(126)

Fig. 10 쇼핑 앱의 시각장애인 접근성 실태

7) <https://www.hankookilbo.com/News/Read/A2021041916290002878?did=kk>

8) Korea Consumer Agency 보도자료: 시청각 장애인, 모바일 앱 이용에 어려움 많아 -쇼핑, 배달, 동영상 스트리밍 앱, 시청각 장애인에게 이용 편의성 강화해야

1.5 제작 일정 및 업무 분담

1.5.1 제작 일정 (Gantt Chart)

주차 별 계획	월	3월				4월				5월				
	주차	1	2	3	4	5	6	7	8	9	10	11	12	13
	날짜	3/4 - 3/10	3/11 - 3/17	3/18 - 3/24	3/25 - 3/31	4/1 - 4/7	4/8 - 4/14	4/15 - 4/21	4/22 - 4/28	4/29 - 5/5	5/6 - 5/12	5/13 - 5/19	5/20 - 5/26	5/27~
설계 복기 및 제작 계획														
물품 수령 및 하드웨어 제작														
S/W 사용할 패키지 선정														
앱인벤터 앱 제작														
LiDAR SLAM 구현														
RFID 읽어오기														
PID제어														
메카닉 휠 제어														
프로세스 통합														
추가 계획 실행														
미비점 보완														
발표 준비														
보고서 작성														
피드백 반영														
피드백 반영														

Fig. 11 Gantt Chart

종합설계 1의 설계일정을 토대로, 3월과 4월 하순을 이용하여 하드웨어 조립을 마친 후, 각 센서별 및 부품 별로 구동에 필요한 S/W 시스템을 설계한다. 개별 센서의 S/W 코딩이 완료 되면 모든 프로세스를 통합하여 모빌리티의 구동 시스템을 완성한다. 통합이 끝난 후, 보완해야 할 부분은 보완하고, 후속 계획으로 남겨두었던 SLAM 추가 시스템 설계를 고려하여 계획을 완성하였다.

1.5.2 업무 분담

이름	업무
한준모	조장, 발표, slam, 카메라
윤 상	아두이노, 앱 인벤터
박병준	아두이노, 통신, 제작
장한솔	아두이노, 통신
홍기원	자료조사, 보고서 작성

표 1. 업무 분담표

II. 본론

2. 개념 설계

2.1 Brain Storming

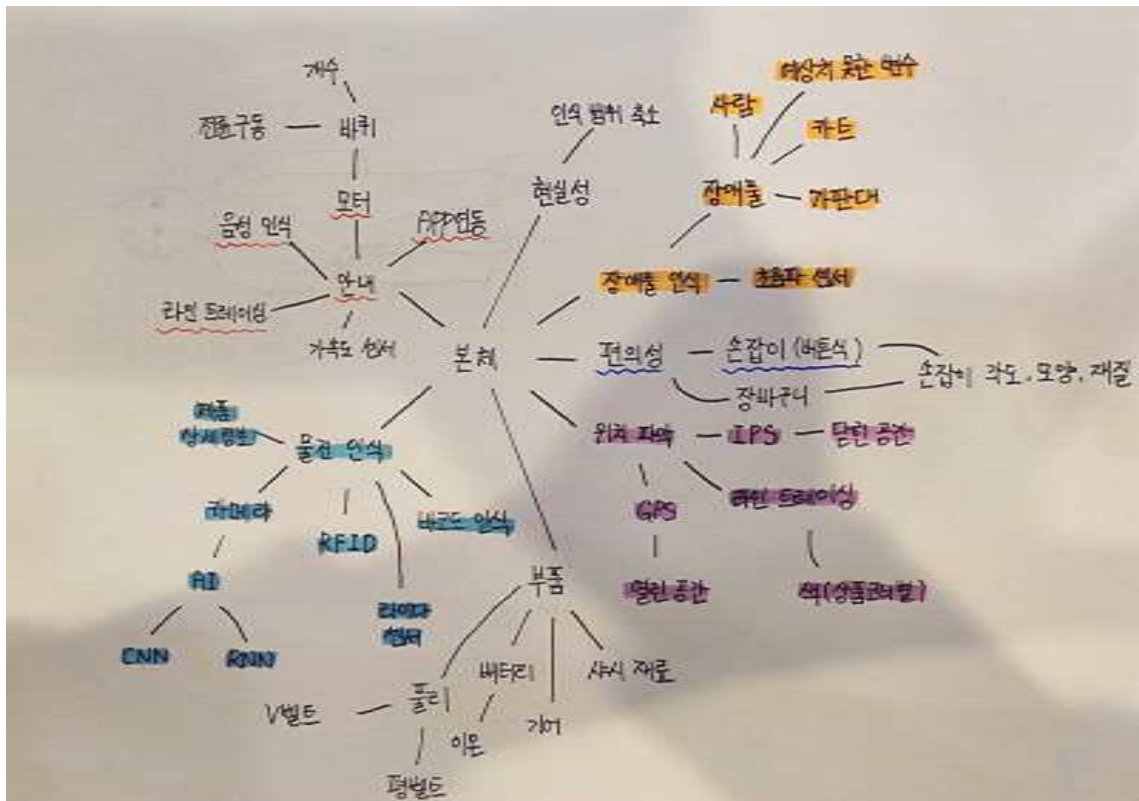


Fig. 12 Brainstorming mind map

2.2 제품의 기능 및 목적(구상단계)

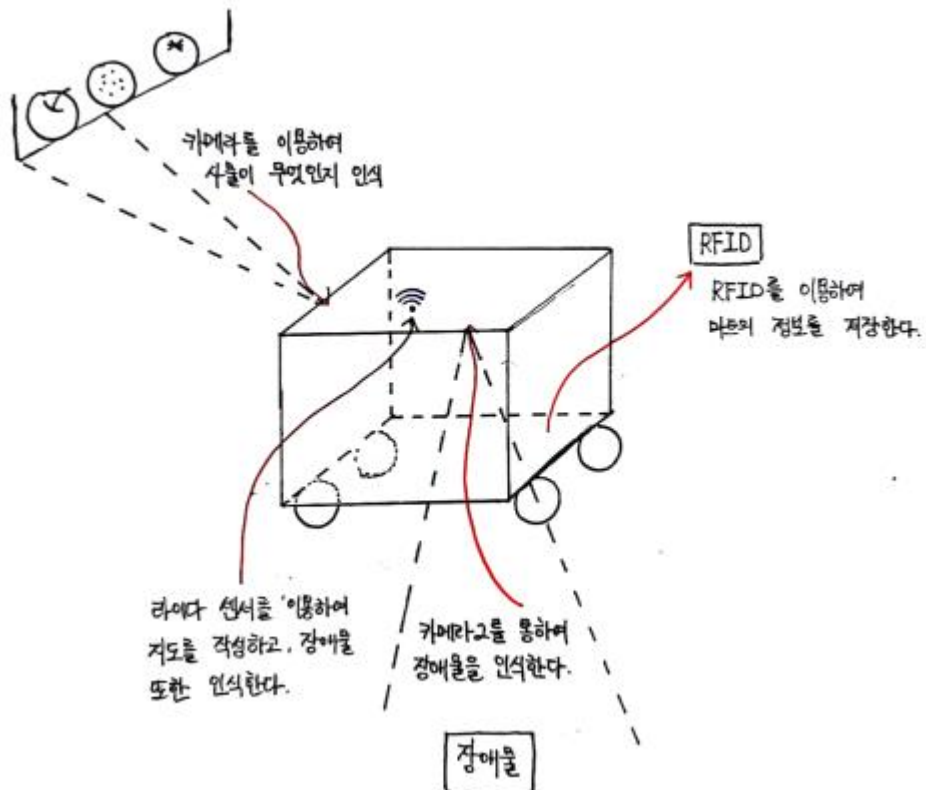


Fig. 13 아이디어 스케치(1)

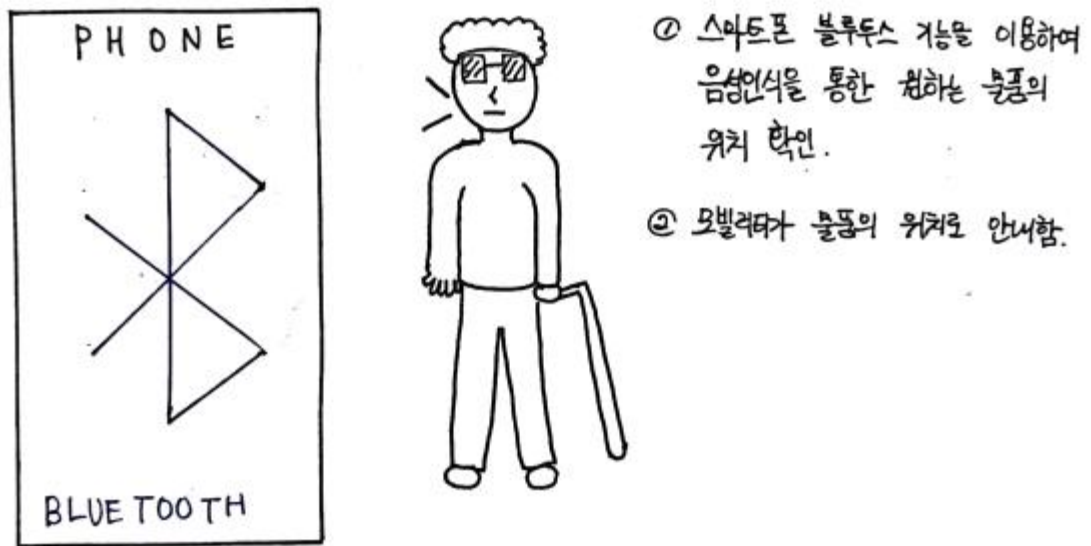


Fig. 14 아이디어 스케치(2)

마트 전용 자율 주행 봇의 주기능	사용자를 원하는 제품이 있는 곳으로 안내
	제품의 상세 정보 안내
	시작/도착 지점에 따른 이동 가능
	장애물 피해 이동 가능
부가 기능	장바구니 역할 대체 가능
	스마트폰과 연결하여 물체 인식 가능
	APP 이용한 사용자의 음성 인식 가능

표 2. 마트 전용 자율주행 봇의 기능

자율주행을 주기능으로 하여 사용자를 안내하고, 부가기능을 시각장애인을 위한 요소를 추가하여 시각장애인을 위한 마트 전용 자율주행 봇의 기능을 설계하였다.

2.3 설계 Concept

2.3.1 알고리즘

1. 로봇과 스마트폰을 연동하여 스마트폰의 음성인식으로 로봇에게 명령을 내린다.
2. 로봇은 명령을 바탕으로 물품의 위치에 안내를 진행한다. 이는 자율주행으로 진행된다.
3. 물품 코너에 도착하면 주변 사진을 바탕으로 물품의 상세 위치를 파악한다. 이는 이미지 구별이 가능한 ai를 활용한다.
그리고 그 상세 위치를 알려준다. 이는 물품들이 어떤 종류의 물품들이 있는지도 확인한다.
4. 물품을 고르고 나면 봇에게 원하는 물품이 맞는지 확인을 진행한다. 확인을 진행하면서 이 물품의 상세정보를 알려준다.
5. 모든 물품을 고르고 나면 다시 계산대로 안내를 진행한다.

2.3.2 제품 설명

- 이는 간단히 물건을 사는 과정이며 이러한 루트가 아닌 일반적인 소비자 입장에서 진행을 한다면 우선 과자 코너에 가서 과자가 뭐가 있는지 확인하고 고르는 과정도 포함 할 수 있다.

- 상세정보 또한 스마트폰 연동으로 음성으로 알려주도록 진행을 하여 스마트폰을 통해 로봇과 대화를 하고 원하는 물품을 고르는 것이다.

- 안내의 경우에는 손잡이를 잡았을 때 로봇이 움직이고 그 로봇을 따라가는 방식으로 진행된다. 이 과정은 안내견의 방식과 비슷하다. 하지만 가는 길의 위험도를 로봇이 스스로 파악하여 속도를 조절하고 피해서 간다.

- 추가적으로 물품을 담을 수도 있게 설계를 한다면 더 편할 것이고 이는 자율주행 카트로도 활용할 수 있을 것이다.

2.3.3 장바구니 최대 무게 계산

연간 가구 당 평균 장바구니 무게: 650kg

고려대와 한국갤럽연구소가 조사한 바⁹⁾에 의하면(전국 609가구를 대상으로 시장을 보는 형태와 장보는 시간 조사) 전체적으로 대형할인마트를 이용하는 경우가 가장 많았고, 집 근처 대형할인마트를 이용하는 빈도는 평균적으로 주 3.76회라고 한다.

본 모빌리티에서 사용하는 장바구니 부피: $(31.5\text{cm}) \times (22.5\text{cm}) \times (19\text{cm}) = 13.5\text{L}$

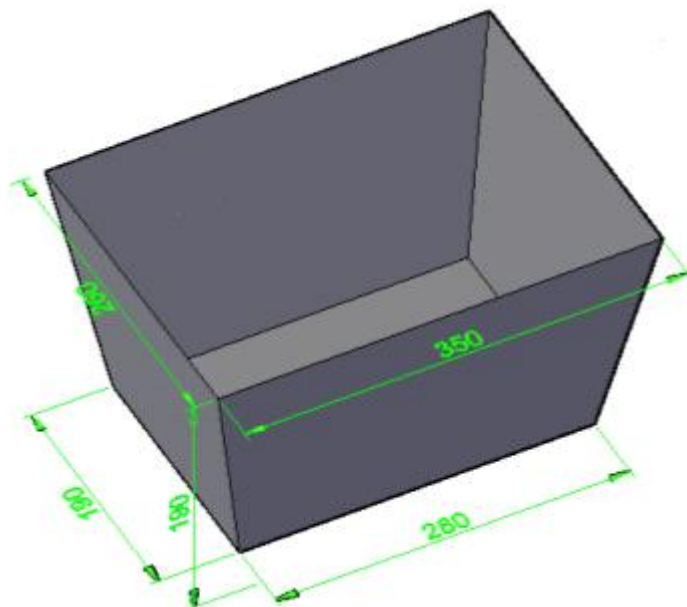


Fig. 15 본 모빌리티에 사용할 장바구니 규격

9) 가정에서의 장보기 행태 조사 결과: 고려대, 한국갤럽연구소



Fig. 16 쇼핑카트 제원¹⁰⁾

부피 비는 60L: 13.5L, $60/13.5=4.44$

따라서 $650\text{kg}/12/3.76/4.44 = 3.2\text{kg}$

=(본 모빌리티의 장을 1회 볼 때 장바구니 용량)

본 모빌리티에 사용할 장바구니에 최대 3.2kg을 담을 수 있다.

3. 기계적 요소 상세설계

3.1 구동장치 선정

3.1.1 바퀴 선정

바퀴와 코딩만으로 조향제어가 가능한 메카넘 휠을 이용하고자 한다.

메카넘 휠의 주요 특징 및 장점:

- 조향장치 없이 좌·우 이동 및 회전이 가능해 구조/비용 효율이 높다.
- 주행 반경이 작아 공간 활용과 기동성이 높다.
- 효율적으로 공정 간 운반/이·적재가 가능해 공정 시간을 단축시킬 수 있다.
- AGV 플랫폼 내 부품 수를 최소화함으로써 부피 및 중량, 비용을 절감할 수 있다.

Fig. 18은 기구에 사용된 메카넘 휠의 외형을 나타낸다. 메카넘 휠은 둘레에 45° 기울어져 장착된 15개의 Free Roller로 구성된다.

¹⁰⁾

<https://pro9000.co.kr/product/%EC%97%A0%ED%8C%8C%EC%9D%B4%EC%96%B4-60l-%ED%95%B8%EB%93%9C%EC%B9%B4%ED%8A%B8-%EC%87%BC%ED%95%91%EC%B9%B4%ED%8A%B8-%EB%A7%88%ED%8A%B8%EC%B9%B4%ED%8A%B8-%EC%88%98%EB%A0%88-%EC%9A%B4%EB%B0%98-%EC%9B%8C%EC%BB%A4%EC%B9%B4%ED%8A%B8/631/>



Fig. 17 메카넘 휠 도면 11)



Fig. 18 메카넘 휠 스펙 선정

메카넘 휠 스펙: 직경 97mm, 두께 48mm. 커플링 직경 12mm, 내부 구멍 4mm

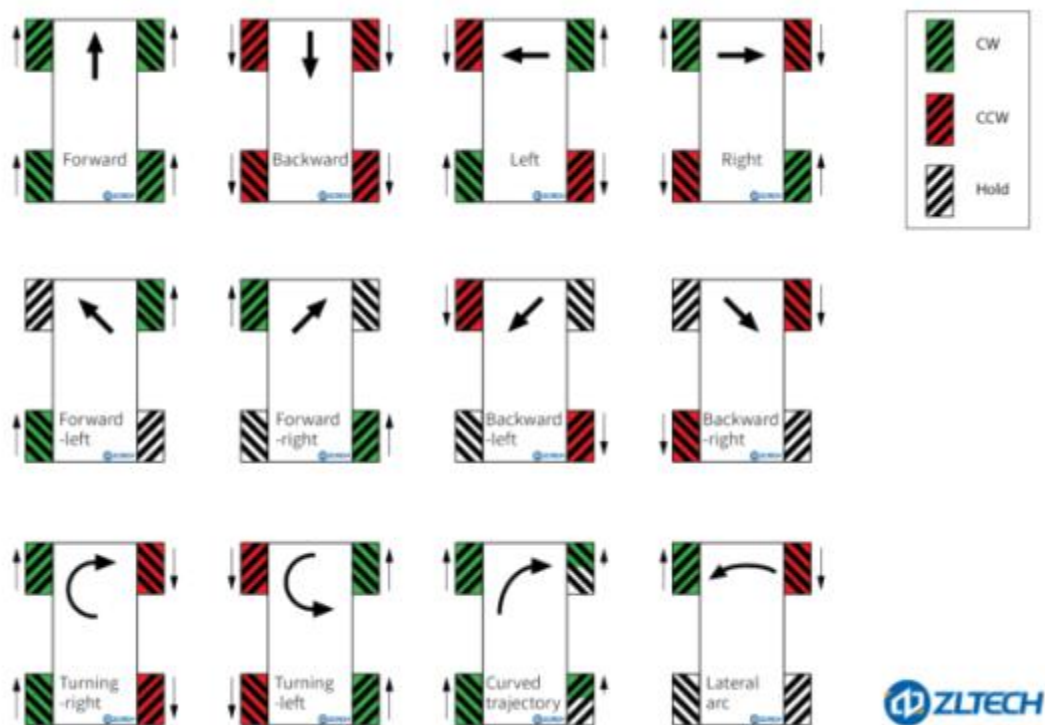


Fig. 19 메카넘 휠을 이용한 주행 및 회전¹²⁾

11) <https://eleparts.co.kr/goods/view?no=3191315>

Fig. 19는 메카넴 휠이 X축을 중심으로 Y축 방향 정회전 할 경우, 롤러에 의한 휠의 진행방향을 나타낸다. 메카넴 휠의 경우, 구동 시 휠의 회전동력은 Free Roller에 의해 AGV의 좌·우측 휠에서 각각 휠의 회전방향의 $\pm 45^\circ$ 방향으로 작용하기 때문에 X방향의 속도 성분이 발생하여 미끄럼 현상을 일으키게 된다.

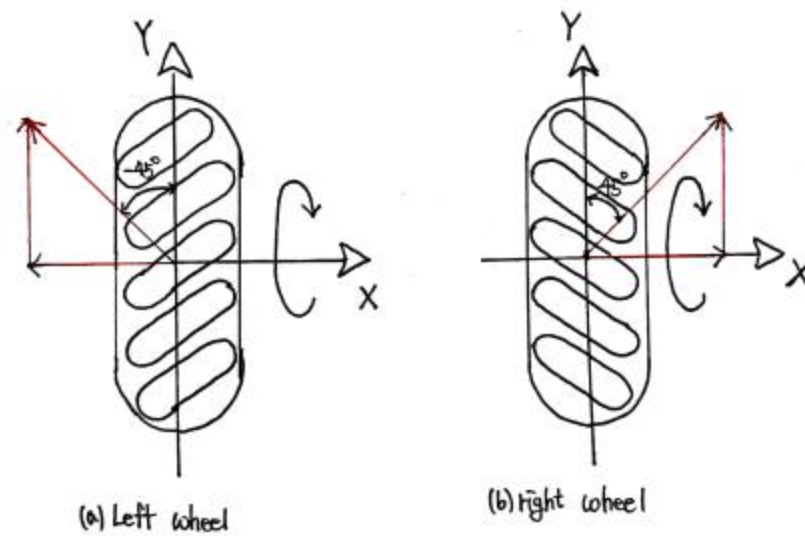


Fig. 20 메카넴 휠 미끄럼 현상을 이용한 진행 방향 설정¹³⁾

3.1.2 변수 정의

차량 질량 $m = 25[kg]$

차량 가속도 $a = 1.1[m/s^2]$

주행속도 $V_{\max} = 1.11[m/s]$

차량 감속도 $b = 0.55[m/s^2]$

가속시간 $t_a = 1[sec]$

바퀴 반지름 $R = 0.076[m]$

제동시간 $t_b = 2[sec]$

정지마찰계수 $\mu_s = 0.7$

요구동력을 계산하기 전, 주행속도 V_{\max} 는 인간의 평균 걷는 속도 $4km/h$ 를 차량의 최대속도로 설정하였고, 정지마찰계수 μ_s 는 마트의 바닥을 일반적인 타일 바닥으로 가정하였을 때 평균적인 값으로 설정하였다.

12) <http://yeogienews.com/today/254586>

13) 공학 석사 학위 논문: 메카넴 휠을 이용한 전방향 주행형 AGV의 평행 주차 제어, (2010. 2), 부경 대학교 대학원, 기계공학부 신종훈

3.1.3. 자유물체도

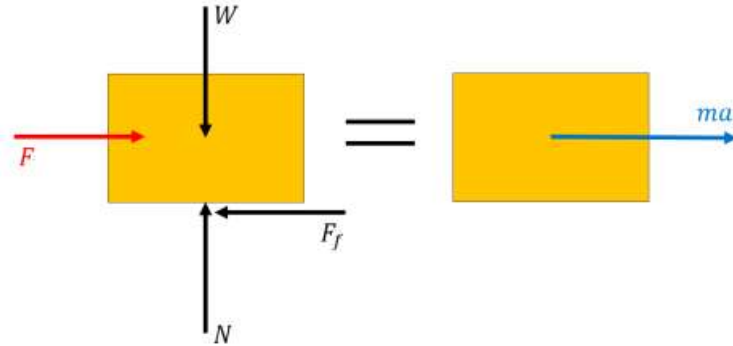


Fig. 21 모빌리티의 자유물체도와 운동역학 선도

3.1.4. 모터의 토크 계산

(바퀴의 최대 각속도)

$$w_{\max} = \frac{V_{\max}}{R} = \frac{0.97m/s}{0.076m} = 12.76[rad/s] \approx 12.76 \frac{rad}{sec} \times \frac{60sec}{1min} \times \frac{1rev}{2\pi rad} = 121.85[rpm]$$

(부하 토크)

$$T_L = \frac{\mu_s WR}{2} = \frac{(0.7)(25kg)(9.81m/s^2)(0.076m)}{2} \approx 6.52[N \cdot m]$$

(가속 토크)

$$T_a = \frac{2\pi Jf}{gt_a} = \frac{2\pi WR^2 w_{\max}}{8gt_a} = \frac{2\pi(25kg)(9.81m/s^2)(0.076m)^2(12.76rad/s)}{8(9.81m/s^2)(1s)} \approx 1.45[N \cdot m]$$

(전체 토크)

$$T_{\max} = T_L + T_a = 6.52 + 1.45 = 7.97[N \cdot m]$$

이 때, 안전계수 $n = 1.5$ 를 선정하여 바퀴에 걸리는 전체 토크 값을 계산한다.

$$\therefore T_{wheel, \max} = nT_{\max} = (1.5)(7.97) \approx 11.96[N \cdot m]$$

구동 바퀴는 4개이므로, 각 바퀴에 걸리는 최대 토크는 $T_{wheel, \max} = 2.99[N \cdot m]$ 이다.

3.1.5. 요구동력 계산

구동 바퀴 하나에 걸리는 토크의 크기와 바퀴의 각속도를 알 때, 요구동력 및 출력을 계산할 수 있다.

$$\therefore P = T \times \omega = (2.99)(12.76) = 38.15[W]$$

따라서, 바퀴 하나에 필요한 요구동력은 약 $40[W]$ 이다.

3.1.6 모터 선정¹⁴⁾

41.3[W] 5700[rpm]인 DC모터를 선정하였고 속도 제어, 순시 정지, 정회전 및 역회전 조절은 아두이노를 통하여 조절할 것이다.

$$\begin{aligned}\frac{T_{wheel}}{T_{motor}} &= \frac{N_{motor}}{N_{wheel}} \\ \therefore T_{motor} &= \frac{N_{wheel}}{N_{motor}} T_{wheel} \\ &= \frac{121.85}{5700} \times 2.99 = 0.064[N \cdot m]\end{aligned}$$

위의 계산식을 근거로 모터는 IG-42GM+Encoder 01TYPE DC 12V 모터로 선정하였다. 또한, 모터에 걸리는 최대 토크가 0.0686[N · m]보다 작으므로 안전하다.



Fig. 22 IG-42GM+Encoder 01 TYPE DC 12V 모터¹⁵⁾

Output [W]	Voltage [V]	Rated Load(부하 시 정격 사양)			NO LOAD Speed [rpm]	Weight [kg]
		Speed [rpm]	Torque [N · m]	Current [A]		
41.3	12	5700	0.0686	5.5	7000	2.0

표. 3 IG-42GM+Encoder 01 TYPE DC 12V 모터의 제원

안전계수를 곱하지 않았을 때

$$\therefore T_{wheel, \max} = T_{\max} = 7.97 \approx 7.97[N \cdot m]$$

14) https://eduino.kr/product/detail.html?product_no=1560&cate_no=55&display_group=1

15) http://ds-parts.co.kr/goods_detail.php?goodsIdx=15602

구동 바퀴 4개 이므로, 각 바퀴에 걸리는 토크는 $T_{wheel,max} = 1.99[N \cdot m]$

$$\begin{aligned} \therefore T_{motor} &= \frac{N_{wheel}}{N_{motor}} T_{wheel} \\ &= \frac{121.85}{5700} \times 1.99 = 0.042[N \cdot m] \end{aligned}$$

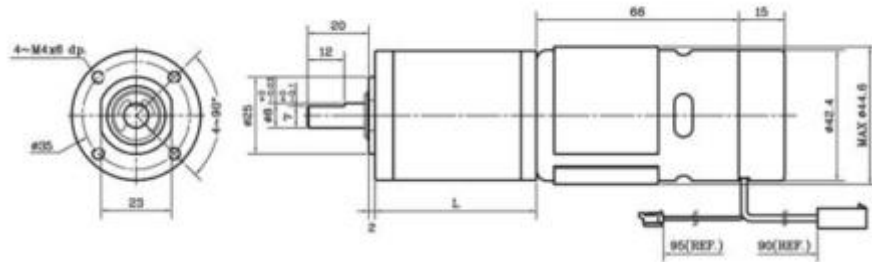


Fig. 23 DC모터 선정 및 스펙, 도면

또한 본 모터에는 감속기를 장착할 수 있으므로, 본 모빌리티의 RPM에 맞도록 49:1의 감속비를 가지는 감속기로 선정하였다. 엔코더가 달려있는 모터를 사용하여 속도 및 방향을 제어할 수 있다.

3.1.7 허브(HUB) 베어링 선정¹⁶⁾

내부 직경: 8mm, 외경 57mm, 순 무게: 52g

모터와 바퀴의 축 구멍을 연결하기 위해 허브 베어링이 필요하다. 앞에서 찾은 바퀴와 축 구멍은 8mm로, 8mm 허브 베어링으로 선정하였다.

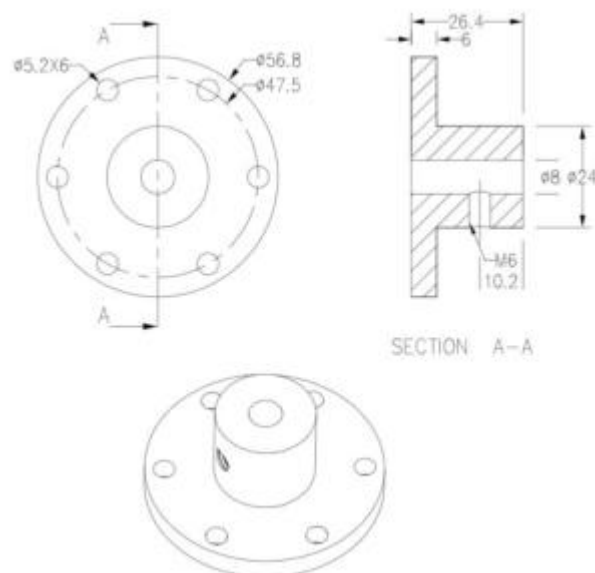


Fig. 24 허브(HUB) 베어링

16) <https://ko.aliexpress.com/i/32824702261.html>



Fig. 25 구동 장치 연결도¹⁷⁾

3.1.8 배터리 선정 ¹⁸⁾

시중에 나와있는 배터리 용량에 맞추면서 1회 사용한 후 충전하도록 하였다. 따라서 결정한 배터리 용량은 360Wh이다.

각 모터를 돌리는데 사용되는 전력: 40W > 모터 총 4개: 160W

1회 장을 보는 평균시간: 79분 > $160W \times 79\text{min} / 60 = 210.7[\text{Wh}]$

설계 변경 전	설계 변경 후
4채널 모터 드라이버	리튬 이온 배터리 팩
	
STX30L-BS (12V 30AH)	[MDD3A] 3Amp 4V-16V DC Motor

17) <https://eleparts.co.kr/goods/view?no=3191315>

18)

<https://smartstore.naver.com/rocketbat/products/5633461337?NaPm=ct%3Dlp80hme8%7Cci%3D0e947aab1d806ca1628446babe37798eab5128c%7Ctr%3DsIs%7Csn%3D747128%7Chk%3D46d734199f318dff9b460dd3abff02659dc0cc5>

	Driver (2 Channels)
설계 변경 내용	
<p>이전 설계 방식에서는 오토바이에 쓰이는 로켓트 배터리를 사용하기로 하였다. 하지만 무게가 무거워 모터 축 구동에 영향이 생길 것이라고 판단하였고 크기가 166*127*175로 큰 규격에 속하기 때문에 프로파일 및 배선 연결 시 방해가 될 것이라고 생각하였다. 따라서 무게가 0.45kg이고 크기가 51*72*73으로 비교적 이전 선정 배터리보다 양호한 편에 속하는 배터리로 변경하기로 하였다. 2채널 배터리팩이어서 배터리를 두 개를 사용해야 한다는 단점이 있었지만 보완 내용에는 지장이 없고, 무게와 규격 측면에서도 기회비용이 있다고 판단하였다.</p>	

장 볼수 있는 횟수 : $360/210.7 = 1.71$ 번

Capacity [Ah]	Voltage [V]	1회 주행 시 소모량		1회 완충 시 사용량	Weight [kg]
		Power [Wh]	Power [%]	Time	
30	12	210.7	58.5	135.09분	3.4

표. 4 STX30L-BS 배터리 제원 표

충전 시간 8~10시간		배터리 길이(mm)	배터리 폭(mm)	배터리 높이(mm)
STX30L	12V 30AH	166	127	175

표. 5 배터리의 규격

3.2 메카넘 휠을 이용한 전방향 주행형 AGV 구성원리 및 기구학적 모델링

3.2.1 전방향 주행형 AGV의 구성



Fig. 26 본 모빌리티의 전방향 주행형 AGV

다음과 같이 프로파일 및 모터, 감속기, 메카넘휠을 이용해 4륜 모빌리티를 구성하

였다.

3.3. 전방향 주행형 AGV의 모델링¹⁹⁾

3.3.1 가정

- ① AGV 의 4 개의 메카넘 휠은 AGV 의 기하학적 중심으로부터 동일한 거리에 위치하고 있다.
- ② 휠마다 하나의 모터가 장착되어 있으며, 각 모터는 독립적으로 구동된다.
- ③ AGV 의 몸체는 강체이다.
- ④ AGV 는 2 차원 평면에서만 이동한다.
- ⑤ 롤러는 그 회전축에 대해 수직방향으로 운동한다.
- ⑥ 외부 외란이 존재하지 않는다

3.3.2 전방향 주행형 AGV의 기구학적 모델링

Fig. 28은 전방향 AGV의 각 휠의 좌표를 나타낸다. 4개의 메카넘 휠의 롤러는 모든 방면에서 대칭을 이루며 각 휠의 회전 속도와 회전방향에 의해 차량의 진행방향 및 속도가 결정된다. 또한 표. 6은 Fig. 28의 매개변수에 대한 설명이다.

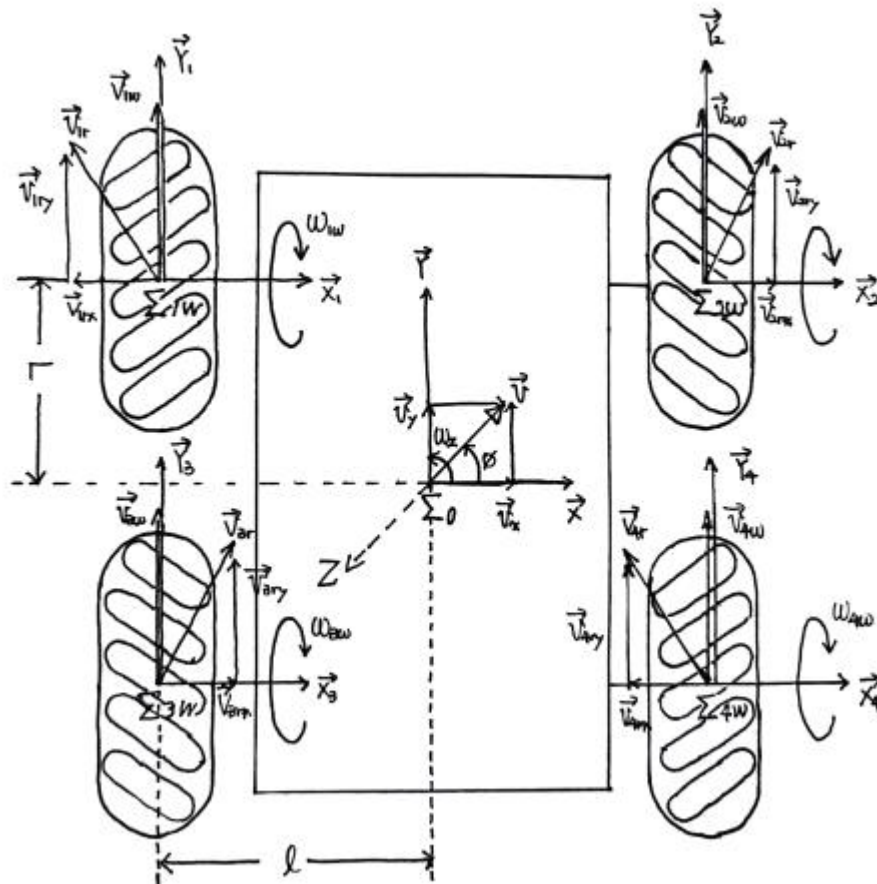


Fig. 27 메카넘휠의 회전좌표계를 이용한 진행방향, 속도 계산

19) 공학 석사 학위 논문: 메카넘 휠을 이용한 전방향 주행형 AGV의 평행 주차 제어, (2010. 2), 부경 대학교 대학원, 기계공학부 신종훈

parameter	Description
Σ_0	전방향 주행형 AGV의 좌표계
Σ_{iw}	i 번째 휠의 좌표($i=1,2,3,4$)
v	전방향 주행형 AGV의 선속도
v_x	전방향 주행형 AGV의 X방향의 선속도
v_y	전방향 주행형 AGV의 Y방향의 선속도
V_{ix}	i 번째 휠의 X축 방향 선속도
V_{iy}	i 번째 휠의 Y축 방향 선속도
V_{iw}	i 번째 휠의 선속도
V_{ir}	i 번째 휠의 롤러 회전축의 수직방향으로 발생하는 선속도
V_{irx}	i 번째 휠의 V_{ir} 의 X축 방향 성분
V_{iry}	i 번째 휠의 V_{ir} 의 Y축 방향 성분
l	좌·우 휠 사이 거리의 1/2
L	휠 중심에서 AGV중심 사이의 거리
w_z	전방향 주행형 AGV의 회전각속도
w_{iw}	i 번째 휠의 회전각속도
\emptyset	전방향 주행형 AGV의 전진방향각

표. 6 회전 좌표계 매개변수

바퀴의 선속도는 휠의 반경과 휠의 회전각속도를 곱한 값이므로

$$V_{iw} = R_w \times w_{iw} \quad (1.1)$$

이때 지면과 롤러의 접점에서 롤러의 회전축의 수직방향으로 선속도 V_{ir} ($i=1,2,3,4$)가 발생한다. 따라서 V_{ir} 을 V_{iw} 로 나타내면 다음과 같다.

$$V_1 = \begin{bmatrix} V_{1x} \\ V_{1y} \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{\sqrt{2}} \\ 1 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} V_{1w} \\ V_{1r} \end{bmatrix} \quad (2.1) \quad V_2 = \begin{bmatrix} V_{2x} \\ V_{2y} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} \\ 1 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} V_{2w} \\ V_{2r} \end{bmatrix} \quad (2.2)$$

$$V_3 = \begin{bmatrix} V_{3x} \\ V_{3y} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} \\ 1 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} V_{3w} \\ V_{3r} \end{bmatrix} \quad (2.3) \quad V_4 = \begin{bmatrix} V_{4x} \\ V_{4y} \end{bmatrix} = \begin{bmatrix} 0 & -\frac{1}{\sqrt{2}} \\ 1 & \frac{1}{\sqrt{2}} \end{bmatrix} \begin{bmatrix} V_{4w} \\ V_{4r} \end{bmatrix} \quad (2.4)$$

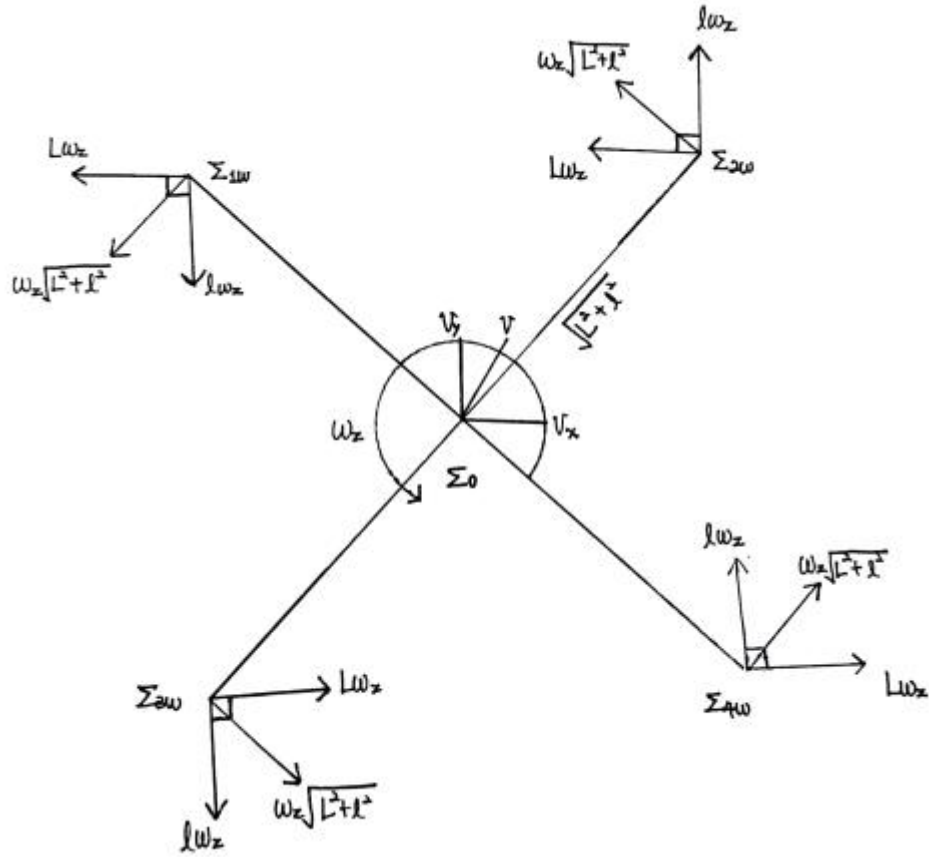


Fig. 28 모빌리티 중심에 대한 각 휠의 선속도 벡터도

Fig. 29에 의해 V 를 v 로 나타내면 다음과 같다.

$$V_1 = \begin{bmatrix} V_{1x} \\ V_{1y} \end{bmatrix} = \begin{bmatrix} v_x - L \\ v_y - l \end{bmatrix} \begin{bmatrix} 1 \\ w_z \end{bmatrix} \quad (3.1) \quad V_2 = \begin{bmatrix} V_{2x} \\ V_{2y} \end{bmatrix} = \begin{bmatrix} v_x - L \\ v_y \quad l \end{bmatrix} \begin{bmatrix} 1 \\ w_z \end{bmatrix} \quad (3.2)$$

$$V_3 = \begin{bmatrix} V_{3x} \\ V_{3y} \end{bmatrix} = \begin{bmatrix} v_x \quad L \\ v_y - l \end{bmatrix} \begin{bmatrix} 1 \\ w_z \end{bmatrix} \quad (3.3) \quad V_4 = \begin{bmatrix} V_{4x} \\ V_{4y} \end{bmatrix} = \begin{bmatrix} v_x \quad L \\ v_y \quad l \end{bmatrix} \begin{bmatrix} 1 \\ w_z \end{bmatrix} \quad (3.4)$$

(2.1)~(2.4)를 (3.1)~(3.4)에 대입하여 정리하면 다음과 같다.

$$V_w = J \cdot V_0 = R_w \cdot w_w \quad (4.1)$$

$$V_w = \begin{bmatrix} V_{1w} \\ V_{2w} \\ V_{3w} \\ V_{4w} \end{bmatrix}, \quad J = \begin{bmatrix} 1 & 2-(L+l) \\ -1 & 1 \quad (L+l) \\ -1 & 1-(L+l) \\ 1 & 1 \quad (L+l) \end{bmatrix}, \quad V_0 = \begin{bmatrix} v_x \\ v_y \\ w_z \end{bmatrix}$$

여기서 V_0 은 AGV의 선속도 벡터를 나타내는 OUTPUT 값이고
 V_w 는 휠의 선속도 벡터를 나타내는 INPUT 값이며
 J 는 변환 행렬로 상수값이다.

(4.1)은 정방행렬이 아니므로 역행렬이 아닌 유사 역행렬을 사용하여 V_0 에 대하여 정리 할 수있다. Pseudu Inverse Matrix를 이용하여 나타내면 다음과 같다.

$$V_0 = J^+ \cdot V_w + (I - J^+ \cdot J)w \quad (5.1) \quad (\text{단}, J^+ = (J^T \cdot J)J^T)$$

$w=0$ 이라고 하면

$$V_0 = \begin{bmatrix} v_x \\ v_y \\ w_z \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 \\ -\frac{1}{(L+l)} & \frac{1}{(L+l)} & \frac{1}{(L+l)} & -\frac{1}{(L+l)} \end{bmatrix} \begin{bmatrix} R_w \cdot \theta_{1w}' \\ R_w \cdot \theta_{2w}' \\ R_w \cdot \theta_{3w}' \\ R_w \cdot \theta_{4w}' \end{bmatrix} \quad (6.1)$$

$L = 229.26mm$, $l = 190mm$, $R_w = 76mm$ 이므로

$$V_0 = \begin{bmatrix} v_x \\ v_y \\ w_z \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 \\ -\frac{1}{419.26} & \frac{1}{419.26} & \frac{1}{419.26} & -\frac{1}{419.26} \end{bmatrix} \begin{bmatrix} 76 \cdot \theta_{1w}' \\ 76 \cdot \theta_{2w}' \\ 76 \cdot \theta_{3w}' \\ 76 \cdot \theta_{4w}' \end{bmatrix}$$

3.4 선정 기본 사양

모빌리티	무게(공차 중량)	25[kg]
	속도	0.97[m/s] (=3.5[km/h])
충전기	모델명	디포스전자 12V 2.5AH 충전기 ²⁰⁾
	충전용량	12V 18AH ~ 30AH
	입력 AC	100~240V 50/60HZ 1.0A
	출력 DC	14.5V 2.5AH
주행	1회 주행 시 소요 시간	79 [min]

표. 7 모빌리티 기본 사양/

20) <https://smartstore.naver.com/rocketbat/products/4245928770>

4. 프레임 설계 및 체결

4.1 프로파일 선정²¹⁾



재질 / 색상	A6063-T5 / 회색
표면처리	양극 처리
무게	1.073kg/m
단면적	395.8259mm ²
절단길이	35mm~6000mm
사용목적	기계구조물, 작업대, 기계커러, 기계창문 및 문, 센스 브라켓, 다용도 선반

Fig. 29 알루미늄 프로파일 선정 및 스펙

21) https://hitc.kr/shop/item.php?it_id=1589516197

4.2 프레임 체결 방식 선정

4.2.1 프로파일 2개 조립²²⁾

조인트 브라켓: SJC 30계열, 렌지볼트: M6,
무두 렌지 볼트: M5 2개를 이용해 체결

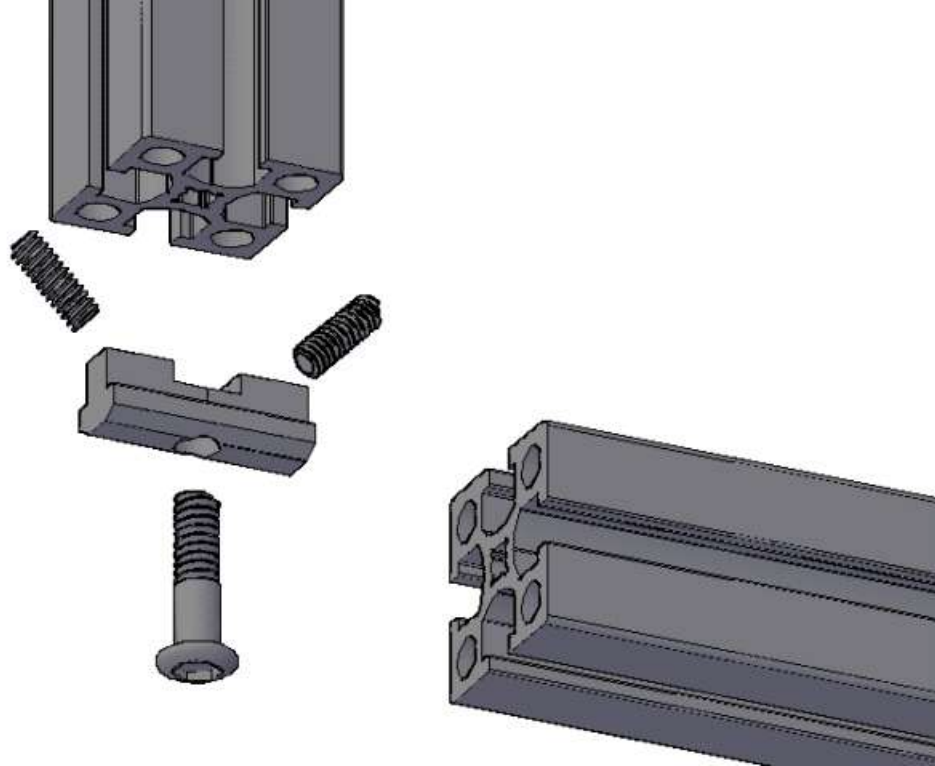


Fig. 30 프로파일 두 개 조립 시 필요부품

22) https://hitc.kr/shop/item.php?it_id=1589517140

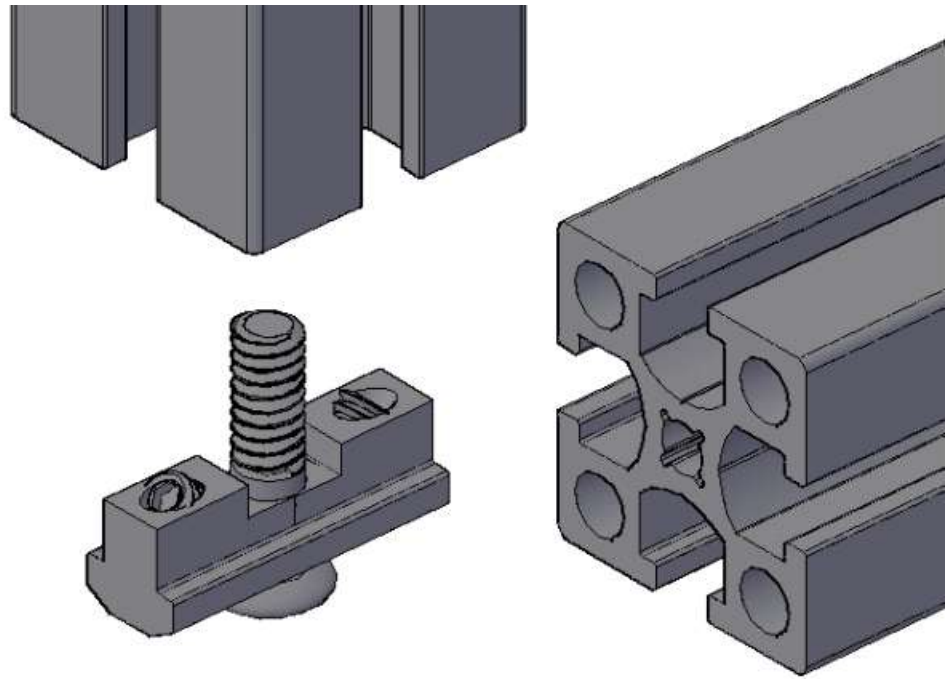


Fig. 31 조인트 브라켓 체결 방식

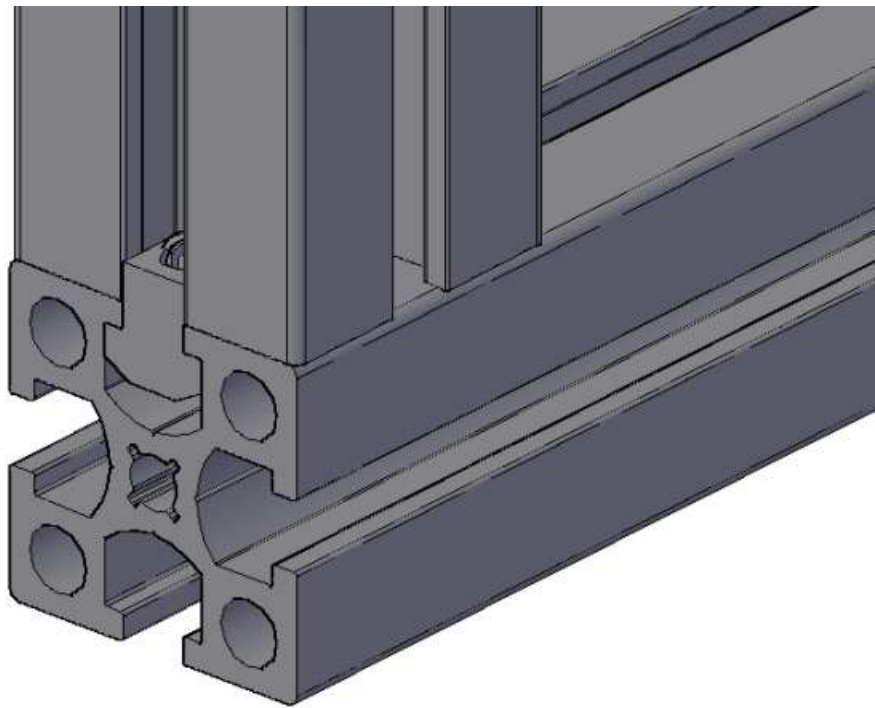


Fig. 32 프로파일 2개 체결한 모습

4.2.2 프로파일 3개 조립²³⁾

코너블럭: SCB 3030-2U, 렌지볼트: M6 3개,
볼트 커버, 코너블럭 커버를 이용해 조립한다.

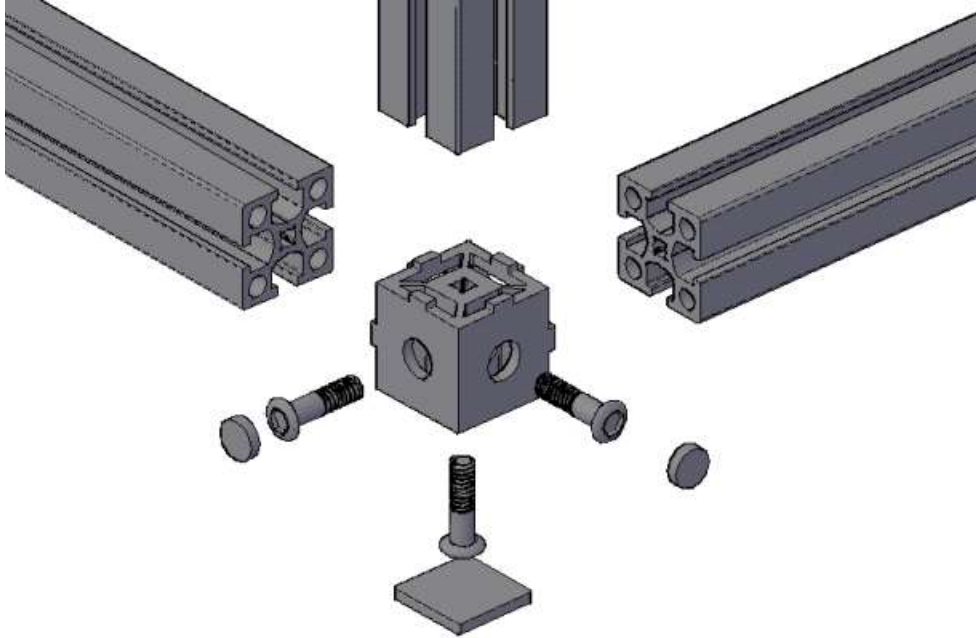


Fig. 33 프로파일 3개 조립시 필요부품

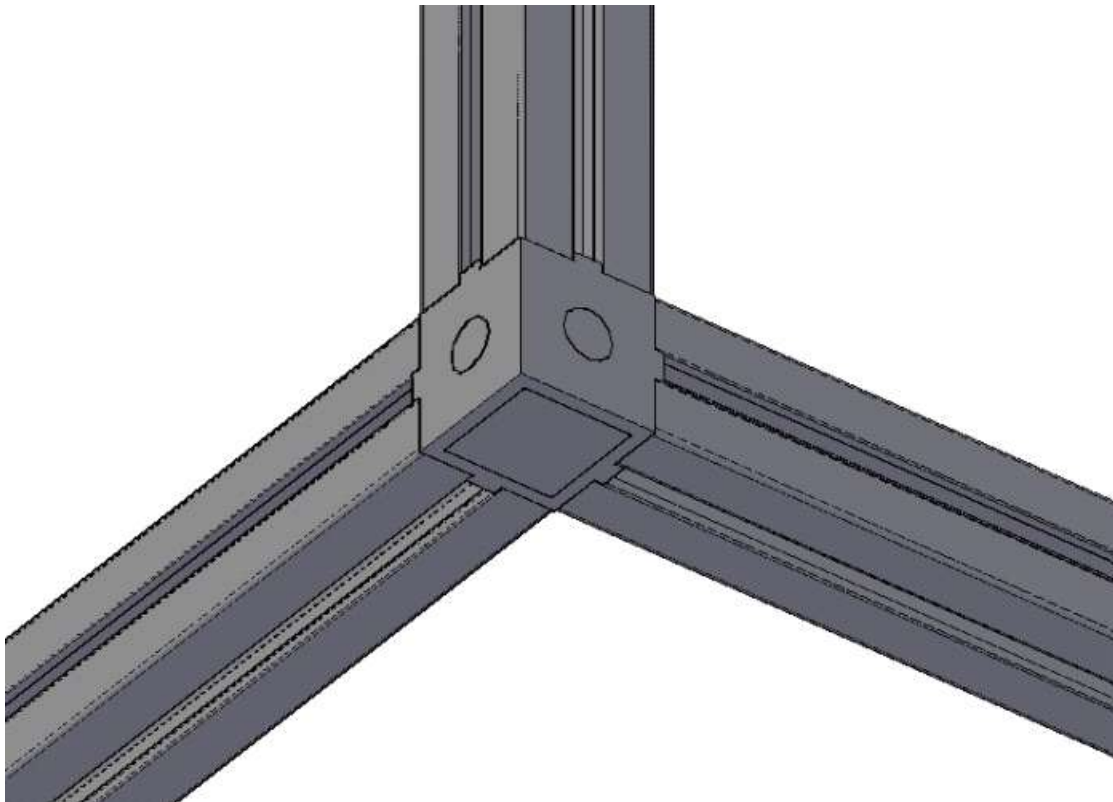


Fig. 34 프로파일 3개 체결한 모습

23) https://hitc.kr/shop/item.php?it_id=1589516913

4.3 프레임 안정성 계산

4.3.1 위층 프레임 처짐 계산

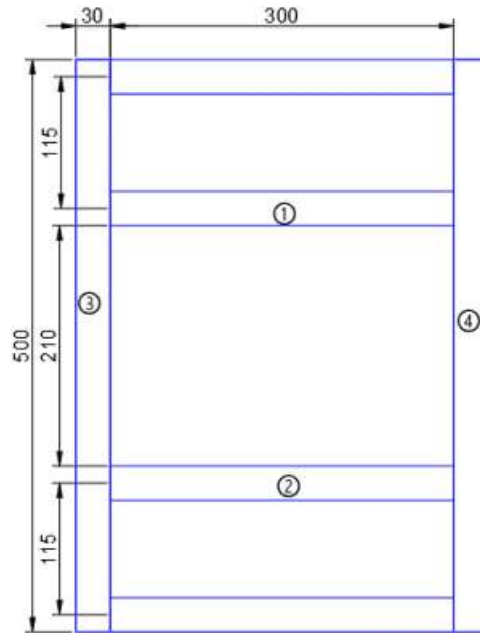


Fig. 35 모빌리티 가장 외쪽 프레임 평면도

1) 1번 프레임

1번 프레임이 받는 처짐은 2번 프레임이 받는 처짐과 같으므로 1번 프레임의 처짐값만 계산하고 2번 프레임에도 적용하기로 한다.

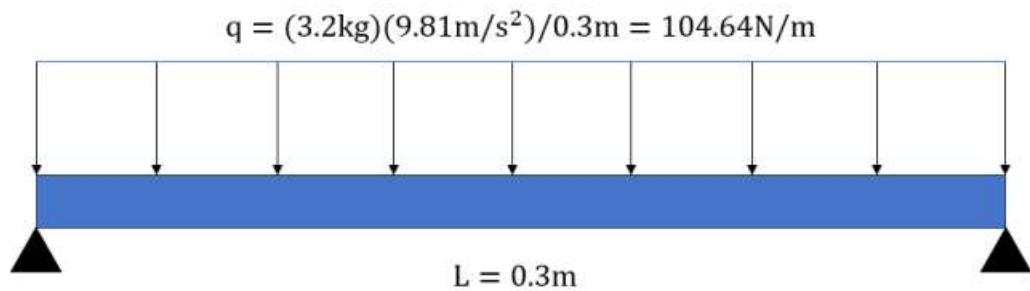


Fig. 36 1번 프레임의 Free-Body Diagram

(알루미늄 프로파일 $E = 70\text{ GPa}$)

(정사각형 단면 관성 모멘트 $I = \frac{b^4}{12}$ ($b = 0.03\text{m}$))

$$(\text{보의 굽힘모멘트}) M = \frac{qL}{2}(x) - qx\left(\frac{x}{2}\right) = \frac{qLx}{2} - \frac{qx^2}{2}$$

$$Elv'' = \frac{qLx}{2} - \frac{qx^2}{2}$$

$$EIv'' dx = \frac{qLx}{2} dx - \frac{qx^2}{2} dx$$

$$EI \int v'' dx = \int \frac{qLx}{2} dx - \int \frac{qx^2}{2} dx$$

$$EIv' = \frac{qLx^2}{4} - \frac{qx^3}{6} + C_1$$

이 때, 하중의 대칭성으로부터 $v'(\frac{L}{2}) = 0$ 이다.

$$0 = \frac{qL}{4}(\frac{L}{2})^2 - \frac{q}{6}(\frac{L}{2})^3 + C_1, \quad C_1 = -\frac{qL^3}{24}$$

$$EIv' = \frac{qLx^2}{4} - \frac{qx^3}{6} - \frac{qL^3}{24}$$

$$EIv = \frac{qLx^3}{12} - \frac{qx^4}{24} - \frac{qL^3x}{24} + C_2$$

이 때, 왼쪽 지지점에서 보의 처짐은 0이므로 $v(0) = 0$ 이다.

따라서 $C_2 = 0$ 이다.

$$\therefore v = -\frac{qx}{24EI}(L^3 - 2Lx^2 + x^3)$$

프레임의 최대 처짐은 하중의 대칭성으로부터 $\frac{L}{2}$ 에서 발생한다.

$$\delta_{\max} = v(\frac{L}{2}) = \frac{5qL^4}{384EI} = \frac{5(104.64N/m)(0.3m)^4}{384(70GPa)(0.000675m^4)} = 0.23nm$$

아주 작은 값의 처짐이므로 안정하다 할 수 있다.

2) 2번 프레임

1번 프레임과 같은 처짐을 받는다. 따라서 2번 프레임의 처짐은 안정하다.

3) 3번 프레임

3번 프레임이 받는 처짐은 4번 프레임이 받는 처짐과 같으므로 3번 프레임의 처짐값만 계산하고 4번 프레임에도 적용하기로 한다. Fig 1.을 참고하여 Free-Body Diagram를 그려보면 다음과 같다.

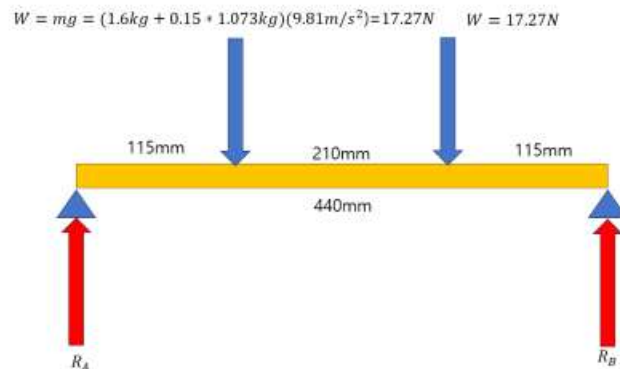


Fig. 37 4번 프레임의 FBD

프레임을 보라고 두고, 하중이 좌우 대칭이므로 정해져있는 공식을 사용하여 처짐을 계산해본다.

프레임을 보라고 두고, 하중이 좌우 대칭이므로 정해져있는 공식을 사용하여 처짐을 계산해본다.

$$\nu = -\frac{Px}{6EI}(3aL-3a^2-x^2), \nu' = -\frac{P}{2EI}(aL-a^2-x^2) \quad (0 \leq x \leq a) \quad (a = 0.115m)$$

$$\nu = -\frac{Pa}{6EI}(3Lx-3x^2-a^2), \nu' = -\frac{Pa}{2EI}(L-2x) \quad (a \leq x \leq L-a) \quad (a = 0.115m)$$

따라서, 보의 최대 처짐량은

$$\delta_C = \delta_{\max} = \frac{Pa}{24EI}(3L^2-4a^2)$$

(알루미늄 프로파일 $E = 70GPa$)

(정사각형 단면 관성 모멘트 $I = \frac{b^4}{12}$ ($b = 0.03m$))

$$(P = W = mg = ((0.3)(0.5)(1.073)kg)(9.81m/s^2) = 1.58N)$$

$$\therefore \delta_{\max} = \frac{Pa}{24EI}(3L^2-4a^2) = \frac{1.58N(0.115m)}{24(70 \times 10^9 N/m^2)(0.000675m^4)} ((3)(0.44m)^2 - (4)(0.115m)^2) = 0.08nm$$

최대 처짐량은 상당히 미미한 것으로 보이며 예측한 결과와 같다.

장바구니는 집중하중이 아닌 분포하중으로 분포하중으로 두고 계산을 중첩의 원리를 이용해보자. 이 때 장바구니는 하중의 대칭성을 적용할 수 있다. 또한 집중하중에 의한 처짐과 분포하중에 의한 처짐의 합을 프레임의 총 처짐이라고 할 수 있다. (중첩의 원리)

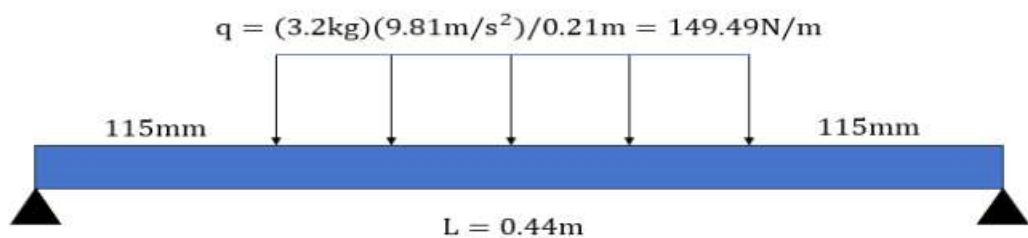


Fig. 38 장바구니 분포하중 처짐 계산

프레임의 최대 처짐은 하중의 대칭성으로부터 $\frac{L}{2}$ 에서 발생한다.

$$\delta_{\max} = v\left(\frac{L}{2}\right) = \frac{5qL^4}{384EI} = \frac{5(104.64N/m)(0.21m)^4}{384(70GPa)(0.000675m^4)} = 0.06nm$$

중첩의 원리를 적용하면,

δ_{\max} , 역시 처짐이 적어 안정하다 볼 수 있다.

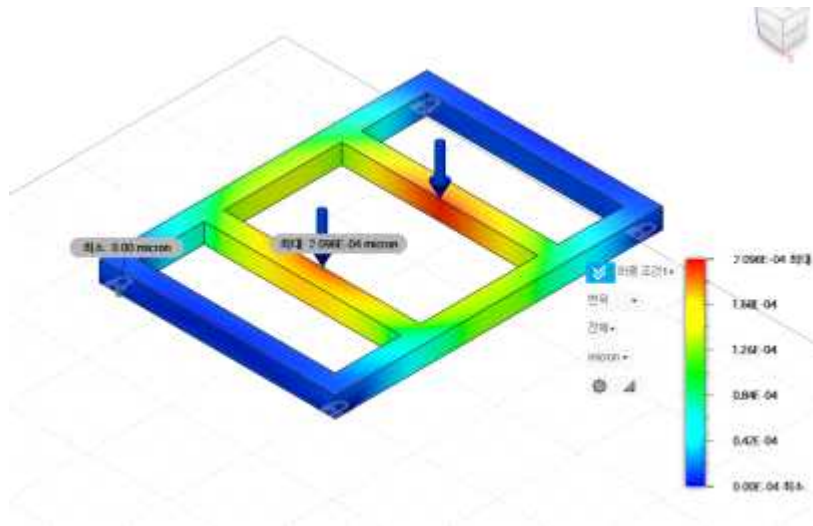


Fig. 39 가장 윗쪽 프레임 처짐 분포

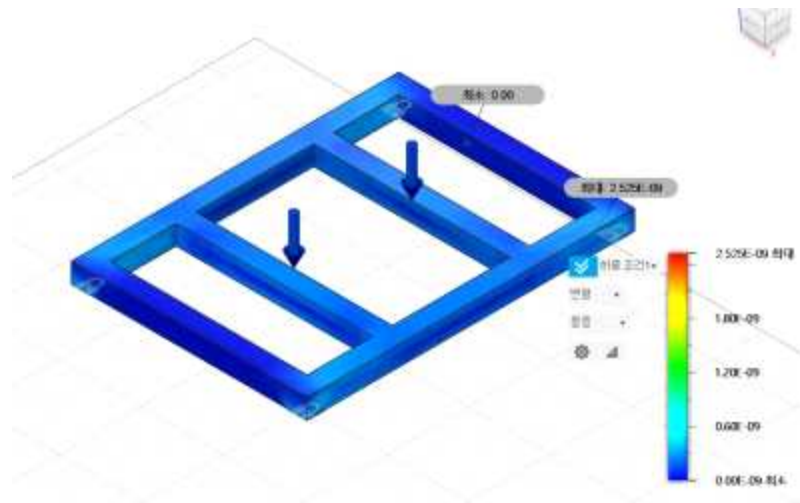


Fig. 40 가장 윗쪽 프레임 변형 분포

윗층 프로파일에 대한 처짐과 변형 분포를 Fusion 360을 이용해 확인하였을 때 계산값이 옳음을 알 수 있었다.

4.3.2 기둥 프레임 좌굴 해석

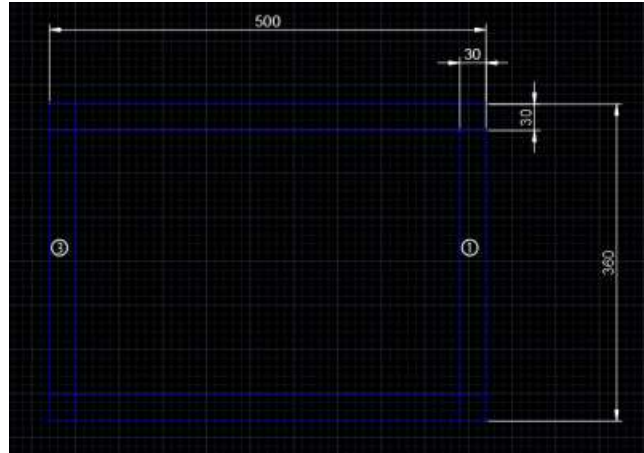


Fig. 41 기둥 프레임의 좌측면도

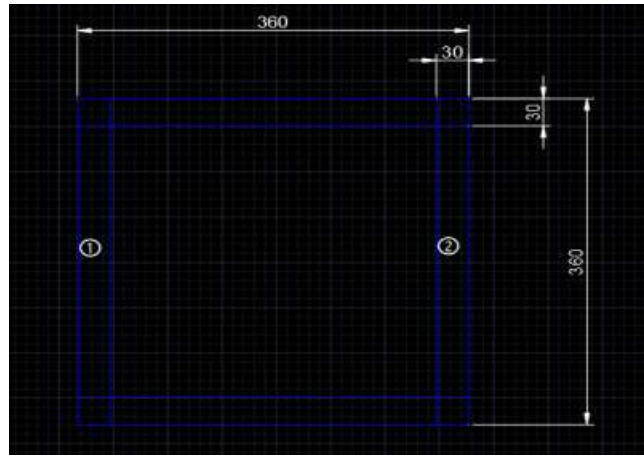


Fig. 42 기둥 프레임의 정면도

모빌리티에서 이용하고 있는 기둥을 회전에 대해 양단고정된 기둥이라고 볼 수 있으므로 사용가능한 임계하중 공식은 임계하중 $P_{cr} = \frac{4\pi^2 EI}{L^2}$ 이다.

(알루미늄 프로파일 $E = 70 GPa$)

(정사각형 단면 관성 모멘트 $I = \frac{b^4}{12}$ ($b = 0.03m$))

$$(\text{임계하중}) P_{cr} = \frac{4\pi^2 EI}{L^2} = \frac{4\pi^2 (70 GPa) ((0.03m)^4 / 12)}{(0.3m)^2} \approx 207 kN$$

알루미늄 프로파일의 탄성 계수는 $70 GPa$ 이고, 비례 한도로 $480 MPa$ 를 사용해보자.

위에서 사용한 임계하중 결과가 유효하려면 기둥의 임계 응력이 알루미늄의 비례한도보다 작아야한다.

$$(\text{임계응력}) \sigma_{cr} = \frac{P_{cr}}{A} = \frac{207 kN}{(0.3m)^2} = \frac{207000 N}{0.09 m^2} = 2.3 MPa$$

(임계응력) $\sigma_{cr} < (\text{알루미늄 프로파일의 비례한도})$

따라서 오일러 좌굴이론을 사용한 임계 하중의 계산은 만족된다.

임계하중이 모빌리티에 가해지는 하중보다 $207kN$ 으로 훨씬 크므로 모빌리티의 기둥(알루미늄 프로파일)은 좌굴이 없다고 볼 수 있다.

이는, 2, 3번 프레임에도 똑같이 적용 가능하므로 좌굴에 대해 안정적인 설계를 하였음을 알 수 있다.

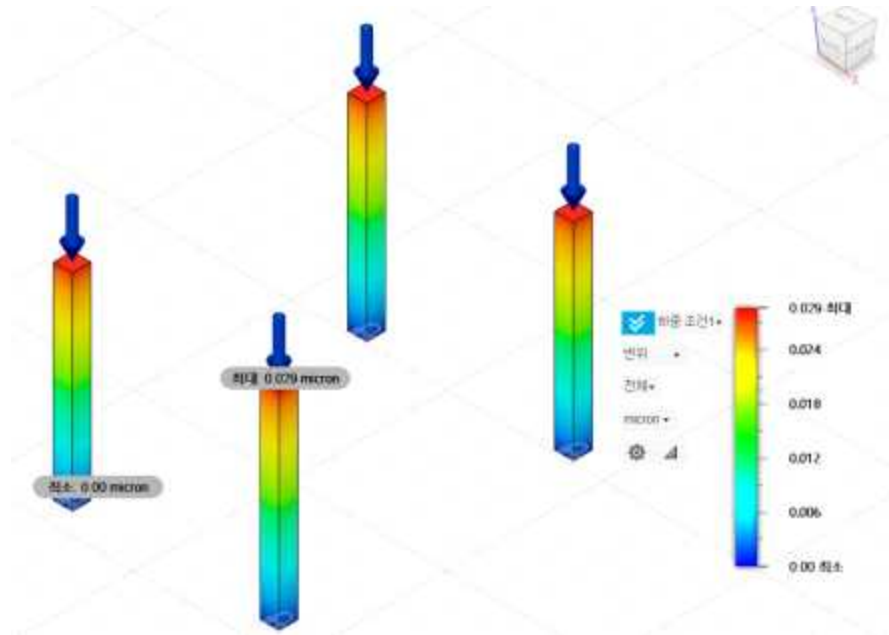


Fig. 43 프로파일 기둥 부분에 대한 처짐 분포

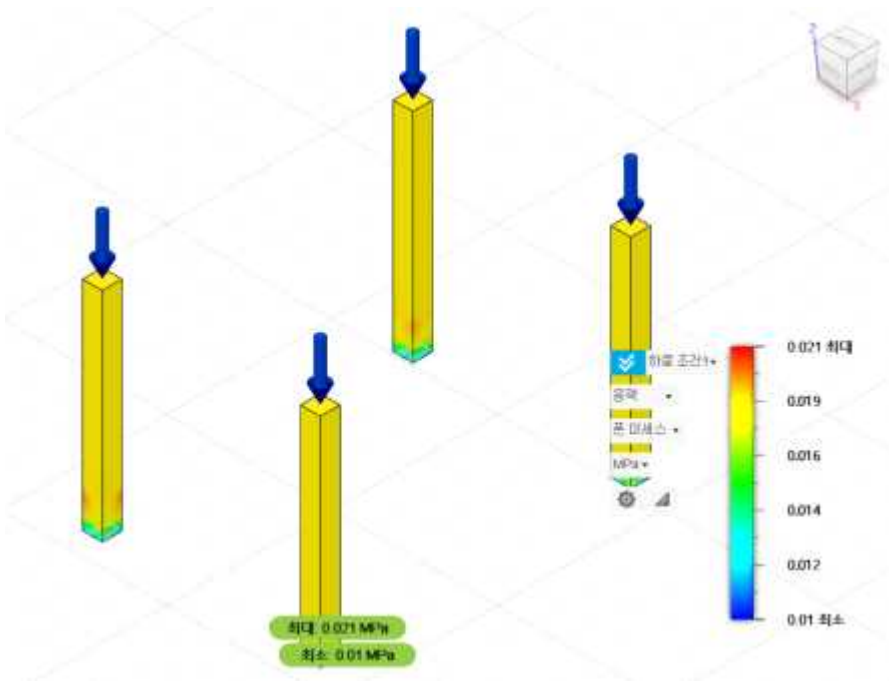


Fig. 44 프로파일 기둥 부분에 대한 응력 분포

기동에 대한 처짐과 변형 분포를 Fusion 360을 이용해 확인하였을 때 계산값이 옳음을 알 수 있었다.

4.3.3 아래층 프레임 처짐 계산

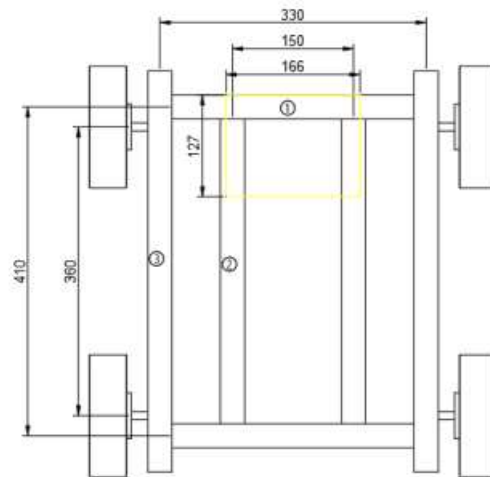


Fig. 45 모빌리티의 아래 프레임 평면도

1) 1번 프레임

1번 프레임의 자유물체도를 그리면 다음과 같다. 배터리는 프레임의 127mm 중에 30mm만을 차지하는 집중하중으로 계산하였다. (실제는 분포하중)

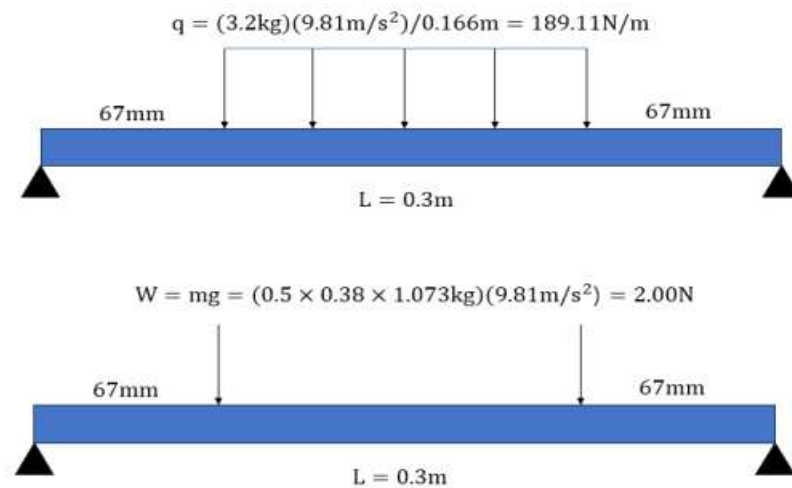


Fig. 46 1번 프레임의 Free-Body Diagram

배터리로 인한 처짐량과 위에서 계산으로 구한 처짐량을 더하여 중첩의 원리를 사용할 것이다.

(알루미늄 프로파일 $E = 70 GPa$)

(정사각형 단면 관성 모멘트 $I = \frac{b^4}{12}$ ($b = 0.03m$))

프레임의 최대 처짐은 하중의 대칭성으로부터 $\frac{L}{2}$ 에서 발생한다.

$$\delta_{\max} = v\left(\frac{L}{2}\right) = \frac{5qL^4}{384EI} = \frac{5(189.11N/m)(0.166m)^4}{384(70GPa)(0.000675m^4)} = 0.04nm$$

$$\nu = -\frac{Px}{6EI}(3aL - 3a^2 - x^2), \nu' = -\frac{P}{2EI}(aL - a^2 - x^2) \quad (0 \leq x \leq a) \quad (a = 0.075m)$$

$$\nu = -\frac{Pa}{6EI}(3Lx - 3x^2 - a^2), \nu' = -\frac{Pa}{2EI}(L - 2x) \quad (a \leq x \leq L - a) \quad (a = 0.075m)$$

따라서, 보의 최대 처짐량은

$$\delta_C = \delta_{\max} = \frac{Pa}{24EI}(3L^2 - 4a^2)$$

(알루미늄 프로파일 $E = 70 GPa$)

(정사각형 단면 관성 모멘트 $I = \frac{b^4}{12}$ ($b = 0.03m$))

$$(P = W = mg = (0.5 \times 0.38 \times 1.073kg)(9.81m/s^2) = 2.00N)$$

$$\therefore \delta_{\max} = \frac{Pa}{24EI}(3L^2 - 4a^2) = \frac{2.00N(0.075m)}{24(70 \times 10^9 N/m^2)(0.000675m^4)} ((3)(0.3m)^2 - (4)(0.075m)^2) = 0.03nm$$

따라서, 총 처짐량은 $0.04nm + 0.03nm = 0.04nm + 0.03nm = 0.07nm$ 으로 예상한 것과 같이 처짐량은 무시할 수 있다. 이 때, 편지지 부분의 기둥에 의한 하중은 직접적으로 하중을 주는 부분이 아니고 그 정도가 미미하기 때문에 무시하였다. 또한, 1번 프레임의 반대편 대칭인 프레임에는 배터리가 올려져 있지 않는데 배터리가 없기 때문에 1번 프레임의 처짐량보다 작을 것이라는 추측은 명확하다.

2) 2번 프레임

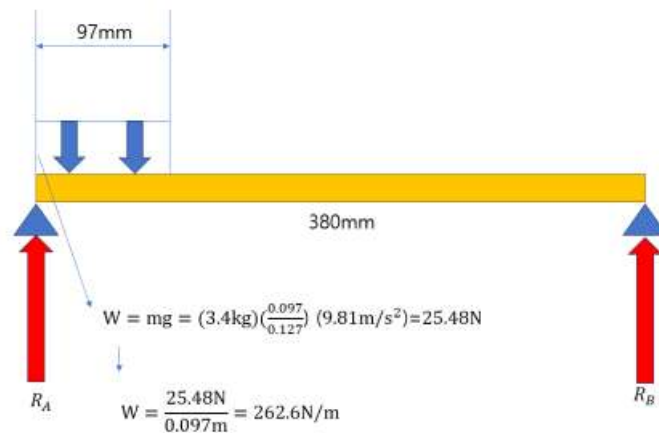


Fig. 47 2번 프레임의 Free-Body Diagram

프레임을 보의 형태라 가정하고, 본 상황의 Appendix를 사용하여 계산한다.

$$\nu = -\frac{qx}{24LEI}(a^4 - 4a^3L + 4a^2L^2 + 2a^2x^2 - 4aLx^2 + Lx^2) \quad (0 \leq x \leq a)$$

$$\nu = -\frac{qa^2}{24LEI}(-a^2L + 4L^2x + a^2x - 6Lx^2 + 2x^3) \quad (a \leq x \leq L)$$

알고 있는 수치들을 대입해보면,

$$\nu = -\frac{(262.6N/m)x}{24(0.38m)(70GPa)(0.000675m^4)}((0.097m)^4 - 4(0.097m)^3(0.38m) + 4(0.097m)^2(0.38m)^2 + 2(0.097m)^2x^2 - 4(0.097m)(0.38m)x^2 + (0.38m)x^2) \quad (0 \leq x \leq 0.097m)$$

$$\nu = -0.00000061x(0.00008853 - 0.00138726 + 0.00543464 + 0.018818x^2 - 0.14744x^2 + 0.38x^2) \quad (0 \leq x \leq 0.097m)$$

$$\nu = -\frac{(262.6N/m)(0.097m)^2}{24(0.38m)(70GPa)(0.000675m^4)}(-(0.097m)^2(0.38m) + 4(0.38m)^2x + (0.097m)^2x - 6(0.38m)x^2 + 2x^3) \quad (0.097m \leq x \leq L)$$

$$\nu = -0.000000088(-0.00003575 + 0.5776x + 0.00009409x - 2.28x^2 + 2x^3) \quad (0.097m \leq x \leq 0.38m)$$

두 구간의 처짐 그래프를 MATLAB을 통하여 그려보았다.

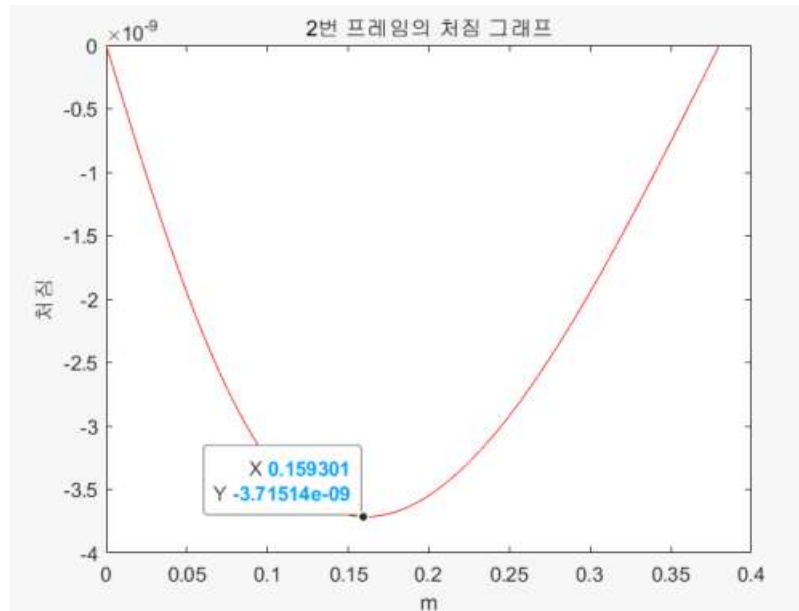


Fig. 48 2번 프레임의 처짐 그래프

2번 프레임의 최대 처짐량은 3.7nm이다.

처짐량은 미미하여 안정한 상태임을 확인 할 수 있다.

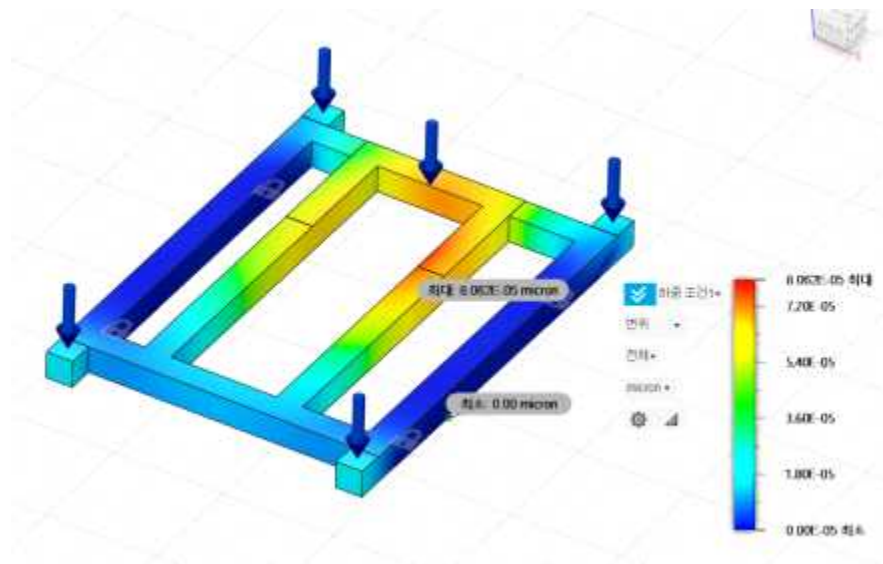


Fig. 49 아래층 프레임 처짐 분포

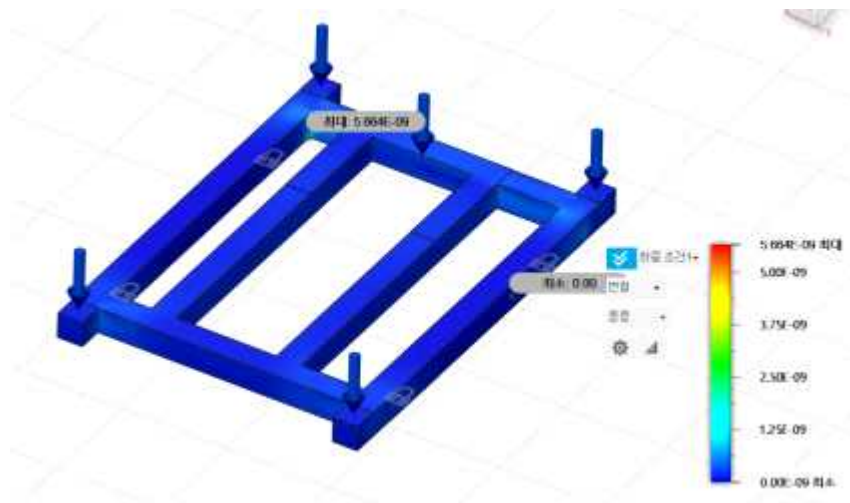


Fig. 50 아래층 프레임 변형 정도

아래층 프레임에 대한 처짐과 변형 분포를 Fusion 360을 이용해 확인하였을 때 계산값이 옳음을 알 수 있었다.

4.4 Fusion 360을 이용한 전체 프레임 해석

프레임의 전체적인 안정성을 시각적으로 파악하고자 Fusion 360 툴을 사용해 표현해보았다.

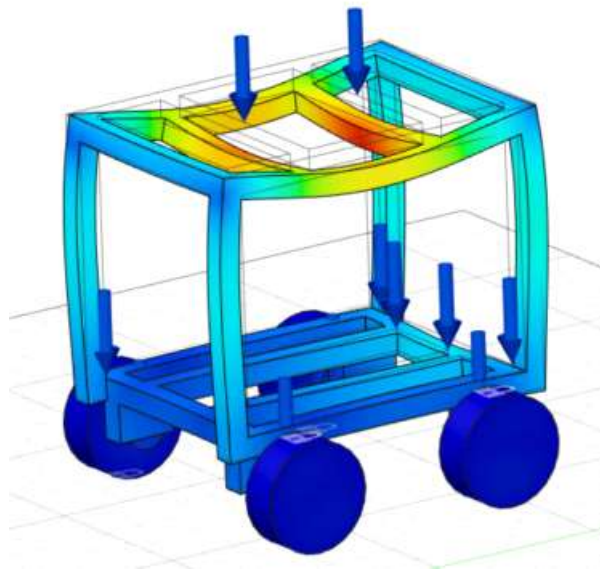


Fig. 51 프레임 처짐 분포(Fusion 360)

장바구니가 짊어지는 기준으로 프레임을 해석하였다. 역시 장바구니가 위쪽에 놓이게 되므로 그 부분의 처짐이 가장 크게 표현됨을 알 수 있다.

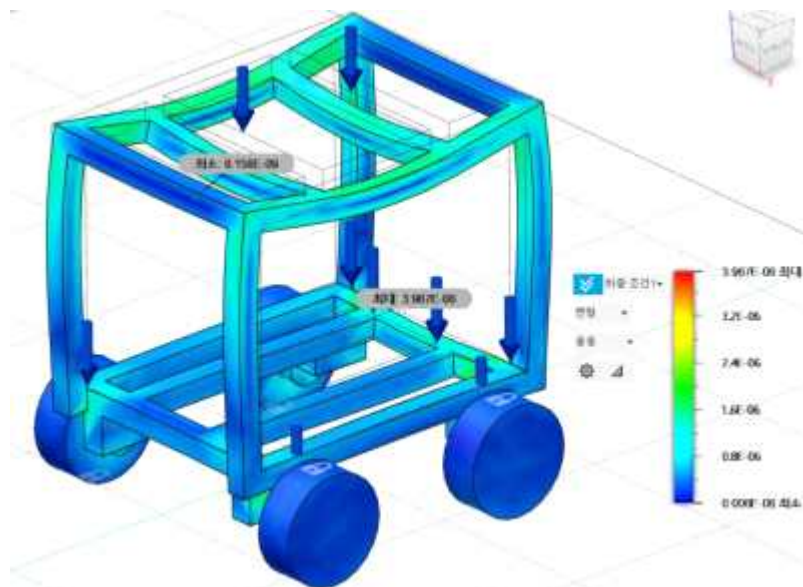


Fig. 52 프레임 변형 분포(Fusion 360)

프레임의 변형값을 보면 굉장히 작은 값(최대 $3.967 \times 10^{-6}m$)을 가짐을 알 수 있다. 따라서 변형에서 안정성을 가짐을 알 수 있다.

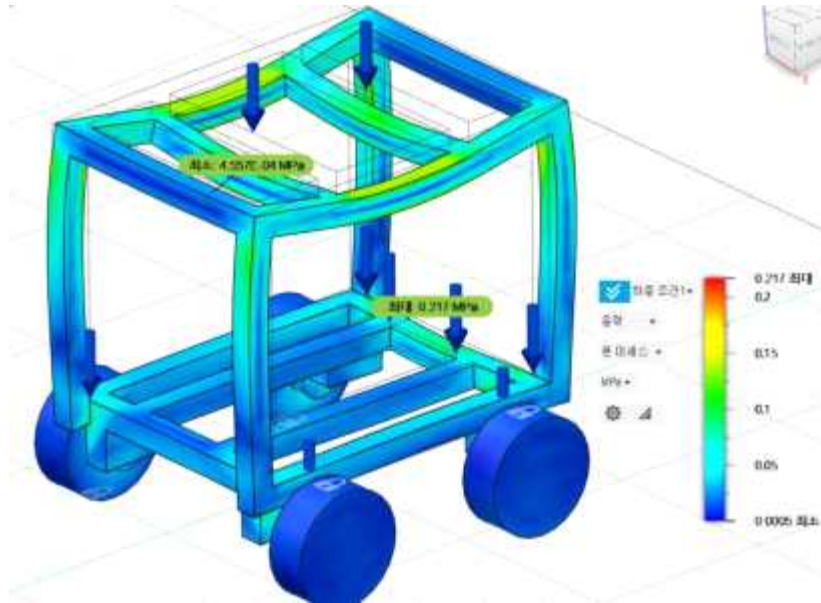


Fig. 53 프레임 응력 분포(Fusion 360)

프레임에서 응력집중이 일어나는 곳이 있는지 알고자 그리게 되었다. 전체적으로 균일한 색임을 보아 고르게 응력이 분포하여 안정함을 알 수 있다.

5. 제품 상세 구동 알고리즘 설계

5.1 알고리즘 순서도

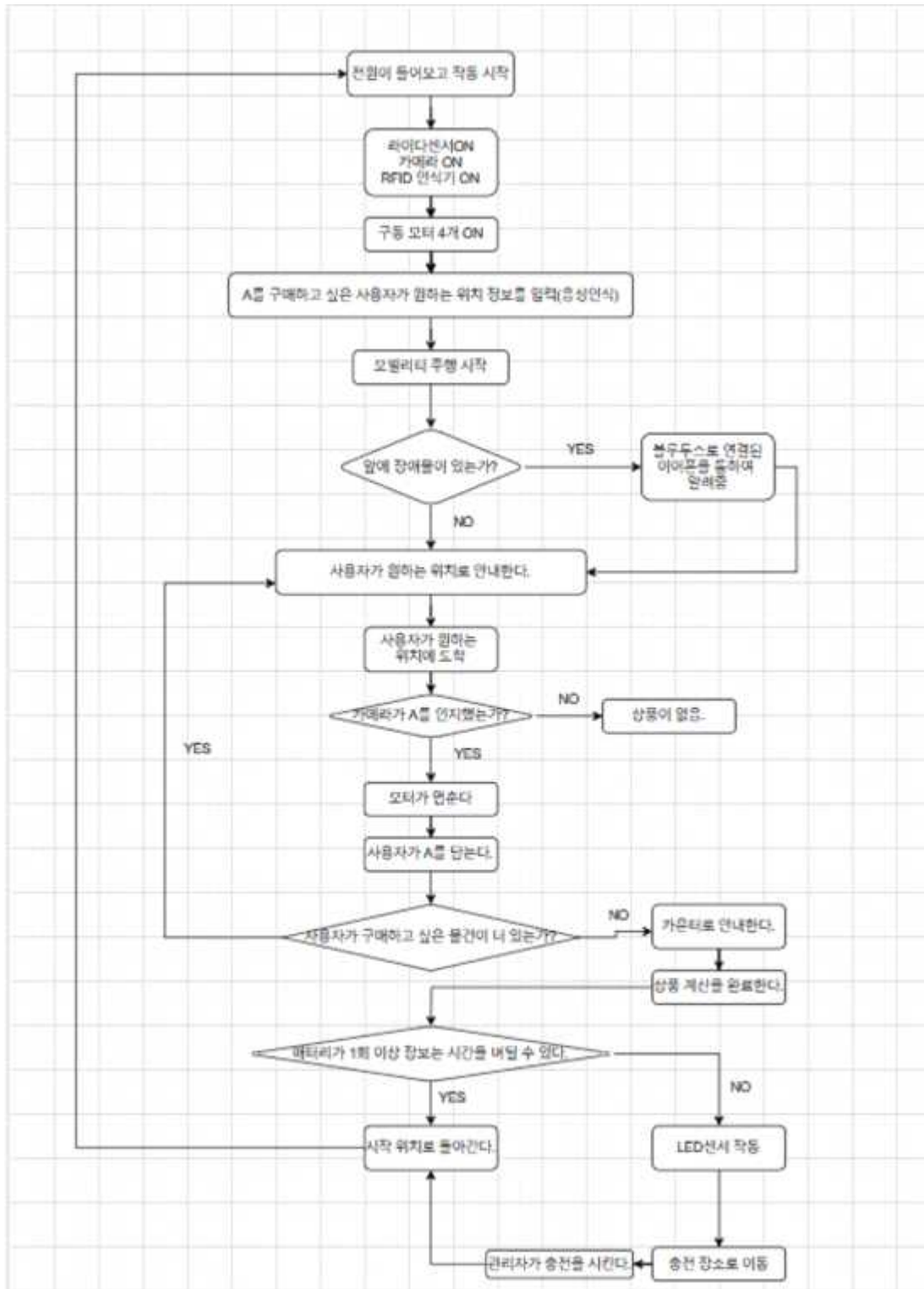


Fig. 54 알고리즘 순서도

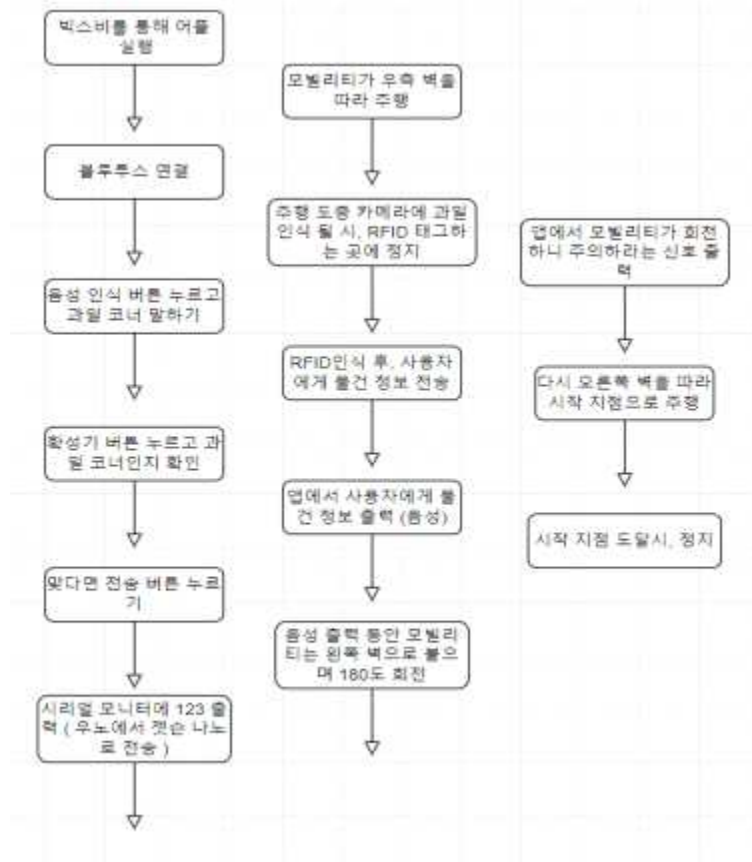


Fig. 51 Yolo V3 가능 시 모빌리티 구동 시나리오

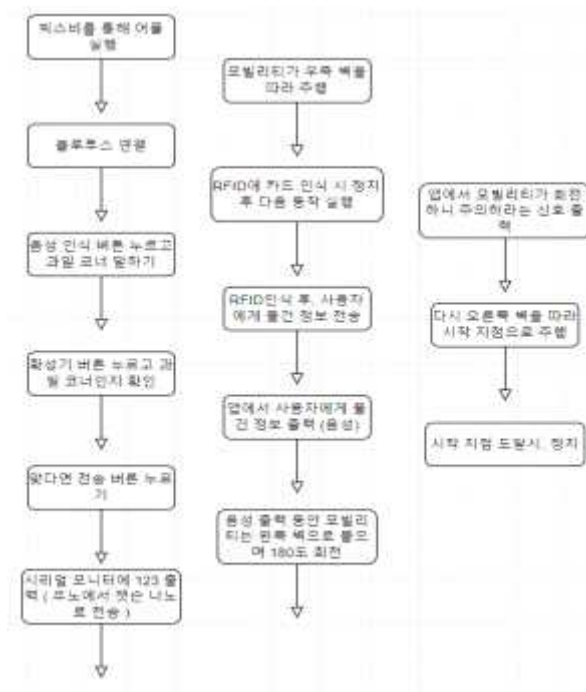


Fig. 52 Yolo V3 불가능 시 모빌리티 구동 시나리오

젯슨 나노의 메모리의 문제로 Yolo v3를 이용하여 카메라로 물건 인식이 불가능할 경우를 상정하여, 최종적인 작동 알고리즘 모델은 Yolo v3 사용 여부를 토대로 두 가지를 만들어 제작을 진행 하였다.

5.2 제어부

5.2.1 SLAM²⁴⁾

SLAM: 동시적 위치추정 및 지도작성

자율주행 차량에 사용되며 주변 환경 지도를 작성하는 동시에 차량의 위치를 작성된 지도 안에서 추정하는 방법이며, SLAM 알고리즘을 통해 차량은 미지의 환경에 대한 지도를 작성할 수 있다.

SLAM을 달성하는 데 사용할 수 있는 기술적 구성요소는 크게 두 가지 유형이 있다. 첫 번째는 프론트엔드 처리 등의 센서 신호 처리이며, 이는 사용하는 센서에 크게 의존하게 된다. 두 번째는 백엔드 처리 등의 자세 그래프 최적화로서 센서와는 무관하다.

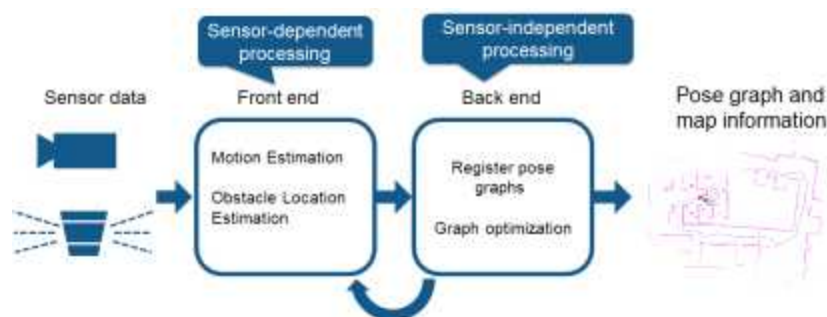


Fig. 57 SLAM 처리 흐름도

SLAM의 방식에는 카메라 SLAM 과 LIDAR SLAM 이 있는데, 그 중 LIDAR SLAM을 사용하고자 한다. LIDAR²⁵⁾ (Light Detection and Ranging)는 레이저 센서(또는 거리 센서)를 주로 사용하여 거리를 측정하는 방법이다.

레이저는 카메라와 ToF 등의 센서보다 훨씬 더 정밀하며 자율주행 차량과 같이 빠르게 이동하는 물체와 관련된 응용 사례에 사용된다. 레이저 센서에서 얻어지는 출력값을 통해 2차원(x, y) 지도를 작성할 수 있다.

24) <https://kr.mathworks.com/discovery/slam.html>

25) <https://kr.mathworks.com/discovery/lidar.html>

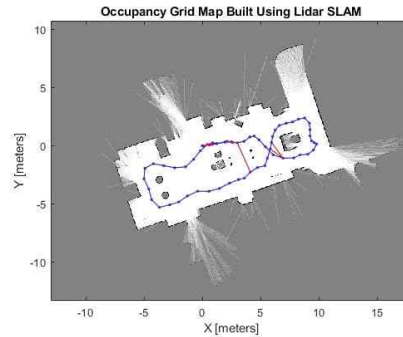


Fig. 64 SLAM을 이용한 지도 작성

GPS는 건물이나 나무 등의 자연물에 의해 위성에서 수신되는 위치 정보에 대한 정확도가 현저히 떨어지므로, LiDAR에 의해 취득되는 SLAM의 위치 정보 또한 인위적, 계절적 사물의 변화에 따라 정확도의 편차가 크다.

즉 실내에서 사용하는 자율주행 모빌리티인 경우 GPS를 사용할 수 없다.²⁶⁾ SLAM이 실내에서 위치를 추정하는데 있어 가장 적합한 방법이다.

5.2.2 SLAM 사용 코드

§ 2d LiDAR를 통해 SLAM을 구현하는 과정은 다음과 같다.

```
include "map_builder.lua"
include "trajectory_builder.lua"
options = {
  map_builder = MAP_BUILDER,
  trajectory_builder = TRAJECTORY_BUILDER,
  map_frame = "map",
  tracking_frame = "base link",
  published_frame = "base link",
  odom_frame = "odom",
  provide_odom_frame = true,
  publish_frame_projected_to_2d = true,
  --use_pose_extrapolator = true,
  use_odometry = true,
  use_nav_sat = false,
  use_landmarks = false,
  num_laser_scans = 1,
  num_multi_echo_laser_scans = 0,
  num_subdivisions_per_laser_scan = 10,
  num_point_clouds = 0,
```

26) 경로점 기반 자율주행 배달 로봇의 주행 안정성을 위한 GPS 모듈과 SLAM 모듈의 비교 연구

[대한전자공학회](#), [전자공학회논문지](#), 학술저널, [전자공학회논문지 제59권 제12호\(통권 제541호\)](#), 2022.12 저자정보: [김봉상](#) (홍익대학교) [남일진](#) (홍익대학교) [류제규](#) (이노시플레이션) [문희창](#) (홍익대학교)


```

(variable) pose_publish_period_sec: Any

pose_publish_period_sec = 5e-3,
trajectory_publish_period_sec = 30e-3,
rangefinder_sampling_ratio = 1.,
odometry_sampling_ratio = 1.,
fixed_frame_pose_sampling_ratio = 1.,
imu_sampling_ratio = 1.,
landmarks_sampling_ratio = 1.,
}MAP_BUILDER.use_trajectory_builder_2d = true
TRAJECTORY_BUILDER_2D.use_online_correlative_scan_matching = true
TRAJECTORY_BUILDER_2D.submaps.grid_options_2d.resolution = 0.3
TRAJECTORY_BUILDER_2D.use_imu_data = false
TRAJECTORY_BUILDER_2D.num_accumulated_range_data = 10
POSE_GRAPH.optimization_problem.odometry_translation_weight = 0.5
POSE_GRAPH.optimization_problem.odometry_rotation_weight = 0.5
return options

```

Fig. 58 SLAM 코드

코드에서 사용된 각 변수들의 의미와 관계는 다음과 같다.

- ① include "map_builder.lua" 및 include "trajectory_builder.lua" :
다른 Lua 파일인 map_builder.lua와 trajectory_builder.lua를 이 파일에 포함합니다.
- ② options = { ... } :
SLAM 시스템의 설정을 담고 있는 테이블을 정의합니다.
- ③ map_builder = MAP_BUILDER :
맵 빌더로 사용할 변수를 설정합니다.
- ④ trajectory_builder = TRAJECTORY_BUILDER :
궤적 빌더로 사용할 변수를 설정합니다.
- ⑤ map_frame = "map" :
맵 프레임의 이름을 설정합니다.
- ⑥ tracking_frame = "base_link" :
로봇의 위치를 추적하는 프레임의 이름을 설정합니다.
- ⑦ published_frame = "base_link" :
게시되는 프레임의 이름을 설정합니다.
- ⑧ odom_frame = "odom" :
오도메트리 프레임의 이름을 설정합니다.
- ⑨ provide_odom_frame = true :
오도메트리 프레임을 제공할 것인지 여부를 설정합니다.
- ⑩ publish_frame_projected_to_2d = true :
프레임을 2D로 투영하여 게시할 것인지 여부를 설정합니다.
- ⑪ use_odometry = true :
오도메트리 데이터를 사용할 것인지 여부를 설정합니다.

- ⑫ `use_nav_sat = false` :
GPS 데이터를 사용할 것인지 여부를 설정합니다.
- ⑬ `use_landmarks = false` :
랜드마크 데이터를 사용할 것인지 여부를 설정합니다.
- ⑭ `num_laser_scans = 1` :
레이저 스캔 데이터의 수를 설정합니다.
- ⑮ `num_multi_echo_laser_scans = 0` :
다중 에코 레이저 스캔 데이터의 수를 설정합니다.
- ⑯ `num_subdivisions_per_laser_scan = 10` :
레이저 스캔 당 하위 분할 수를 설정합니다.
- ⑰ `num_point_clouds = 0` :
포인트 클라우드 데이터의 수를 설정합니다.
- ⑱ `lookup_transform_timeout_sec = 0.2` :
변환 요청의 타임아웃을 설정합니다.
- ⑲ `submap_publish_period_sec = 0.3` :
서브맵 게시 주기를 설정합니다.
- ⑳ `pose_publish_period_sec = 5e-3` :
포즈 게시 주기를 설정합니다.
- ㉑ `trajectory_publish_period_sec = 30e-3` :
궤적 게시 주기를 설정합니다.
- ㉒ `rangefinder_sampling_ratio = 1.` :
레이저 거리 측정 데이터 샘플링 비율을 설정합니다.
- ㉓ `odometry_sampling_ratio = 1.` :
오도메트리 데이터 샘플링 비율을 설정합니다.
- ㉔ `fixed_frame_pose_sampling_ratio = 1.` :
고정된 프레임의 포즈 샘플링 비율을 설정합니다.
- ㉕ `imu_sampling_ratio = 1.` :
IMU 데이터 샘플링 비율을 설정합니다.
- ㉖ `landmarks_sampling_ratio = 1.` :
랜드마크 데이터 샘플링 비율을 설정합니다.
- ㉗ `MAP_BUILDER.use_trajectory_builder_2d = true` :
맵 빌더가 2D 궤적 빌더를 사용하도록 설정합니다.
- ㉘ `TRAJECTORY_BUILDER_2D.use_online_correlative_scan_matching = true` :
온라인 상관 스캔 매칭을 사용하도록 설정합니다.
- ㉙ `TRAJECTORY_BUILDER_2D.submaps.grid_options_2d.resolution = 0.3` :
서브맵의 격자 옵션 해상도를 설정합니다.
- ㉚ `TRAJECTORY_BUILDER_2D.use_imu_data = false` :
IMU 데이터 사용 여부를 설정합니다.

- ㉓ TRAJECTORY_BUILDER_2D.num_accumulated_range_data = 10 :
누적된 레이저 거리 데이터의 수를 설정합니다.
- ㉔ POSE_GRAPH.optimization_problem.odometry_translation_weight = 0.5 :
최적화 문제에서 오도메트리의 변환 가중치를 설정합니다.
- ㉕ POSE_GRAPH.optimization_problem.odometry_rotation_weight = 0.5 :
최적화 문제에서 오도메트리의 회전 가중치를 설정합니다.
- ㉖ return options :
설정된 옵션들을 반환합니다.

§ 구동하는데 필요한 파이썬 파일:

1. cd.py:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import rospy
import serial
import subprocess
from std_msgs.msg import Int32

def serial_listener():
    try:
        # 시리얼 포트 설정
        ser = serial.Serial('/dev/ttyACM1', 9600) # 포트명을 필요에 따라 변경하세요
        rospy.init_node('serial_listener', anonymous=True)
        rate = rospy.Rate(10) # 10Hz

        rospy.loginfo("Serial listener node started")
        print("Serial listener node started")

        while not rospy.is_shutdown():
            if ser.in_waiting > 0:
                data = ser.readline().decode('latin-1').strip()
                rospy.loginfo("Received data: {}".format(data))
                print("Received data: {}".format(data))
                try:
                    number = int(data)
                    print("Received number:", number)
                    if number > 0:
                        print("Launching ROS launch file...")
                        rospy.loginfo("Launching ROS launch file...")
                        # 절대 경로를 사용하여 런치 파일 실행
                        process = subprocess.Popen(
                            ["roslaunch", "/home/weed/catkin_ws/src/move/launch/move.launch"],
                            stdout=subprocess.PIPE, stderr=subprocess.PIPE
                        )
                    else:
                        print("Received number is not positive")
                except ValueError:
                    print("Received data is not a number")
            rate.sleep()
```

Fig. 66 cd.py 파이썬 코드

아두이노 우노로부터 신호가 들어왔을 때, move.launch파일을 실행하게 해준다. 즉 아두이노 우노에서 젯슨 나노로 통신 연결하는 것이다.

2. move.launch:

```
<launch>
  <node name="serial_node" pkg="roserial_python" type="serial_node.py" args="/dev/ttyACM0" />

  <!-- node -->
  <!--node name="publisher" pkg="test" type="publisher.py" output="screen"/-->

  <node name="publisher" pkg="test" type="publisher.py" output="screen"/>
</launch>
```

Fig. 60 move.launch 런치파일

본 launch 파일을 통해 publisher.py와 camera.py를 실행하도록 한다.

3. publisher.py:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import rospy
from std_msgs.msg import Int32

nodeName='publisher'
topicName1='input1'

rospy.init_node(nodeName,anonymous=True)

publisher1=rospy.Publisher(topicName1, Int32, queue_size=5)
ratePublisher=rospy.Rate(1)
intMessage=1

while not rospy.is_shutdown():
    rospy.loginfo(intMessage)
    publisher1.publish(intMessage)
    ratePublisher.sleep()
```

Fig. 61 publisher.py 파이썬 코드

본 파일은 노드 1로 작동하며, 실행될 때 1을 계속 출력하며 1이 출력되는 동안 오른쪽 벽을 따라가는 아두이노 메가 코드를 실행할 수 있도록 하였다.

4. camera.py:

```
#!/usr/bin/env python

import rospy
from std_msgs.msg import Int32

nodeName = 'camera'
topicName = 'camera_info'
stopTopicName = 'stop_signal'

def stop_callback(outputMessage):
    if outputMessage.data == 1:
        rospy.signal_shutdown('Received stop signal from Arduino.')

def main():
    rospy.init_node(nodeName, anonymous=True)
    cameraPub = rospy.Publisher(topicName, Int32, queue_size=5)
    rate_camera = rospy.Rate(1)

    while not rospy.is_shutdown():
        rospy.loginfo("Publishing camera info...")
        cameraPub.publish(0) # Assuming 0 is the camera info
        rospy.Subscriber(stopTopicName, Int32, stop_callback)
        rate_camera.sleep()

if __name__ == "__main__":
    main()
```

Fig. 62 camera.py 파이썬 코드

본 파일은 노드 2로 작동하며, 실행될 때 Received stop signal from Arduino.을 계속 출력한다. Received stop signal from Arduino.이 출력되는 동안 RFID를 읽고 180도 회전하여, 다시 목적지로 오른쪽 벽을 따라 돌아오는 아두이노 메가 코드를 실행한다.

slam

slam을 이용해 mapping을 하는데 있어 ydlidar x2를 사용하였고 사양은 다음과 같다.

모델	X2
전본 추출 주파수(HZ)	3000
범위 주파수(HZ)	5-8
범위 반경(m)	0.1-8
전송 속도	115200
크기(mm)	55*38*101.7

표. 8 ydlidar x2 slam

slam은 google에서 지원하는 google cartographer를 사용하였고 slam을 하기 위해 필요한 odometry는 laser scan matcher라는 laser-tools에 포함된 패키지를 이용하여 구현하였다.

먼저 2d-lidar slam은 google에서 만든(hector slam, google cartographer) 혹은 gmapping을 주로 사용한다. 하지만 google cartographer를 선택한 이유는 다음과 같다.

	Gmapping	Cartographer
개발사	open source	google
라이선스	CCL License	Apache License 2.0
특징	2D	2D,3D
	개발 x	개발 진행중
	static한 상황에 유리	Dynamic한 상황에서도 가능

표. 9 Gmapping, Cartographer 비교

cartographer는 꾸준히 개발이 되어오고 있는 slam이며 움직이며 실시간으로 위치를 파악하고 장애물을 회피해야하는 상황에 잘 맞는 slam이다. 또한 hector slam은 odometry가 없어도 작동하며 매우 가벼운 model인 장점이 있지만 전방에 아무것도 없는 상태라면 로봇이 정지상태로 인식하는 error가 있으며 odometry가 없는 탓에 loop-closer가 타 slam에 비해 잘 안되는 문제점이 있다. 또한 hector slam을 직접 사용해본 결과 실행중 알 수 없는 잡다한 오류가 많았다. 따라서 좀 더 많은 개발이 이루어져있는 google carto grapher를 사용하게 되었다.

google cartographer를 사용하기 위해서는 앞서 말했듯이 odometry가 선행 되어야한다. lidar 자체의 imu가 존재하지 않아 직접적인 계산과정을 통해 odometry를 구현이 필요했다. 따라서 laser scan matcher라는 패키지를 사용하였다. 이것의 간단한 서로 다른 위치에서 획득한 레이저 스캔을 등록하고 상대적인 위치 변환을 계산하는 것이다. 여러 방법 중 전체 공간에 대한 무차별 대입 검색을 기반으로 하는 Correlative Scan Matching (CSM) 알고리즘을 사용하였다. CSM의 순서는 다음과 같다.

참조 scan data를 가우시안 커널을 통해 확률적 grid map 생성 (각 셀은 해당 위치에서 laser point를 관찰할 확률)

1. 현재 scan data를 각 laser point에 반복 투영하여 처리
2. 여러 방면의 위치에서 반복한 data처리를 바탕으로 점수를 매기고 다음 공식을 이용해 높은 점수의 위치 찾기

$$T^* = \operatorname{argmax}_i \sum L(Tp'_i)$$

p' = 회전된 스캔점, $L()$ = 해당위치에서 포인트를 관찰할 확률 반환 함수

4. 단계별 검색으로 인해 계산시간 줄이기

이 기법을 이용하여 odometry를 구현 할 수 있었다. 하지만 odometry 자체에서 loop closer 현상이 발생했고 laser pointer 위치가 점점 돌아 좌표가 정확하게 뜨지 않는 현상이 발생했다. 하지만 더 나은 odometry를 구현할 수 없었기에 이를 바탕으로 slam을 실행하였더니 다음과 같은 결과가 나왔다.²⁷⁾

27) A Novel 2D Laser Scan Matching Algorithm For Mobile Robots Based on Hybrid Features

Jian Wen, Xuebo Zhang†

, Haiming Gao, Jing Yuan and Yongchun Fang

Institute of Robotics and Automatic Information System, Tianjin Key Laboratory of Intelligent Robotics

Nankai University, Tianjin 300350, P. R. China

Email: zhangxuebo@nankai.edu.cn, wenjian@mail.nankai.edu.cn

5.3.5 바퀴(PID) 제어

메카닉 휠의 특성 상, 바퀴의 정밀한 제어가 불가능하다면 방향 주행 뿐만 아니라, 단순 직선 주행에도 문제가 생길 수 있다. 따라서 정밀하고 정확한 바퀴의 제어를 위해 피드백 루프 제어 (PID(PD) 제어)를 사용하였다. PID 세 가지를 전부 제어하지 않고, PD제어를 사용한 이유는, 본 모빌리티는 펄스값을 받기 때문에 정상 상태 오차가 0에 수렴하기 때문에, 정상 상태 오차 제어를 위한 I값을 변수로 둘 필요성이 없기 때문이다. 제어에 사용된 코드는 다음과 같다.

<pre> void encoder_convert() { timeCurr = millis(); if (timeCurr - timePrev > 1000) { timePrev = timeCurr; noInterrupts(); // 인터럽트 발생 수치 계산 후 초기화 for (int i = 0; i < 4; i++) { if (direct[i] == 1) { // forward 방향의 경우 아무 것도 하지 않음 } else if (direct[i] == -1) { encoderA[i] = -encoderA[i]; // backward 방향의 경우 음수로 변환 } else { encoderA[i] = 0; // 정지한 경우 0으로 설정 } } interrupts(); } pd_control_encoder(); calcul(); // 변경한 연립다 값 출력 } </pre>	<pre> void pd_control_encoder() { float error[4]; float derivative[4]; float correction[4]; int target[4]; // 각 바퀴별 목표치 // 각 바퀴별 목표치 설정 for (int i = 0; i < 4; i++) { if (encoderA[i] > 0) { target[i] = 827; // 양수인 경우의 목표치 설정 } else if (encoderA[i] < 0) { target[i] = -827; // 음수인 경우의 목표치 설정 } else { target[i] = 0; // 0인 경우의 목표치 설정 } } // 각 바퀴마다 PD 제어 계산 for (int i = 0; i < 4; i++) { // 오차 계산 error[i] = target[i] - encoderA[i]; // 미분 계산 derivative[i] = error[i] - prevError[i]; // 보정 계산 (PD 제어) correction[i] = Kp * error[i] + Kd * derivative[i]; // 이전 오차 업데이트 prevError[i] = error[i]; // 각 바퀴의 속도 보정 encoderA[i] += correction[i]; } } </pre>
---	--

표 10. PD제어 코드

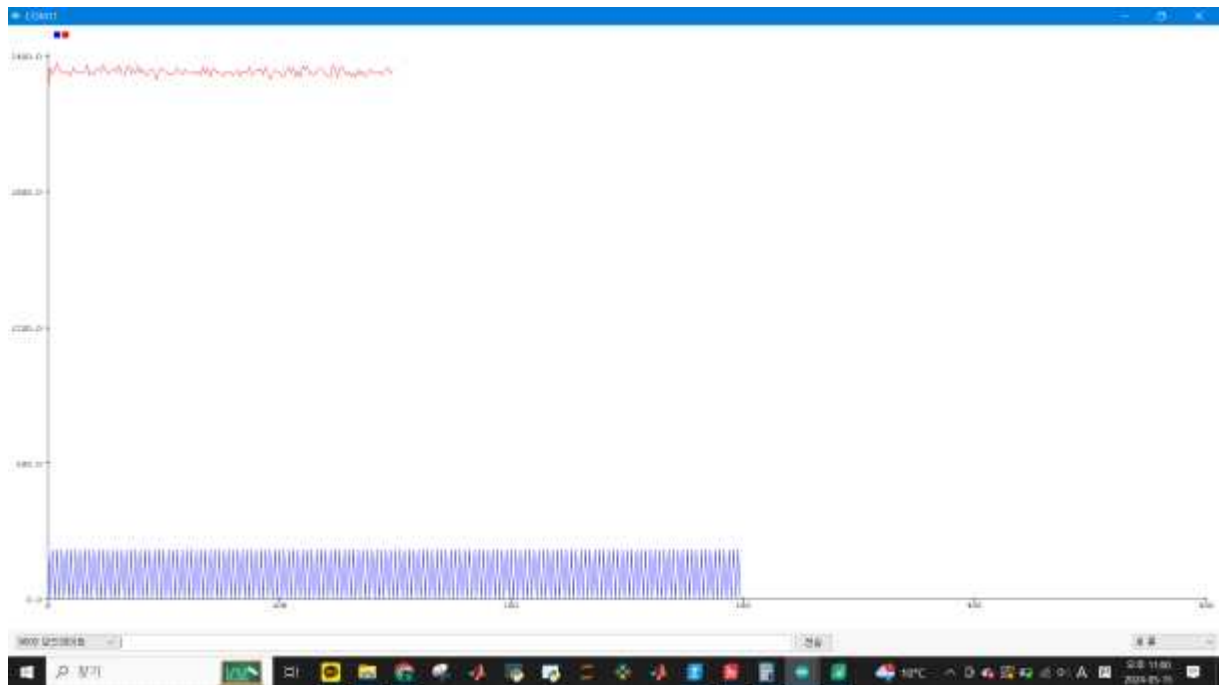


Fig. 64 선정값 ($K_p=0.5$ $K_d=1.4$) 에 대한 응답 선도



Fig. 65 PD 제어 응답 선도 (1)

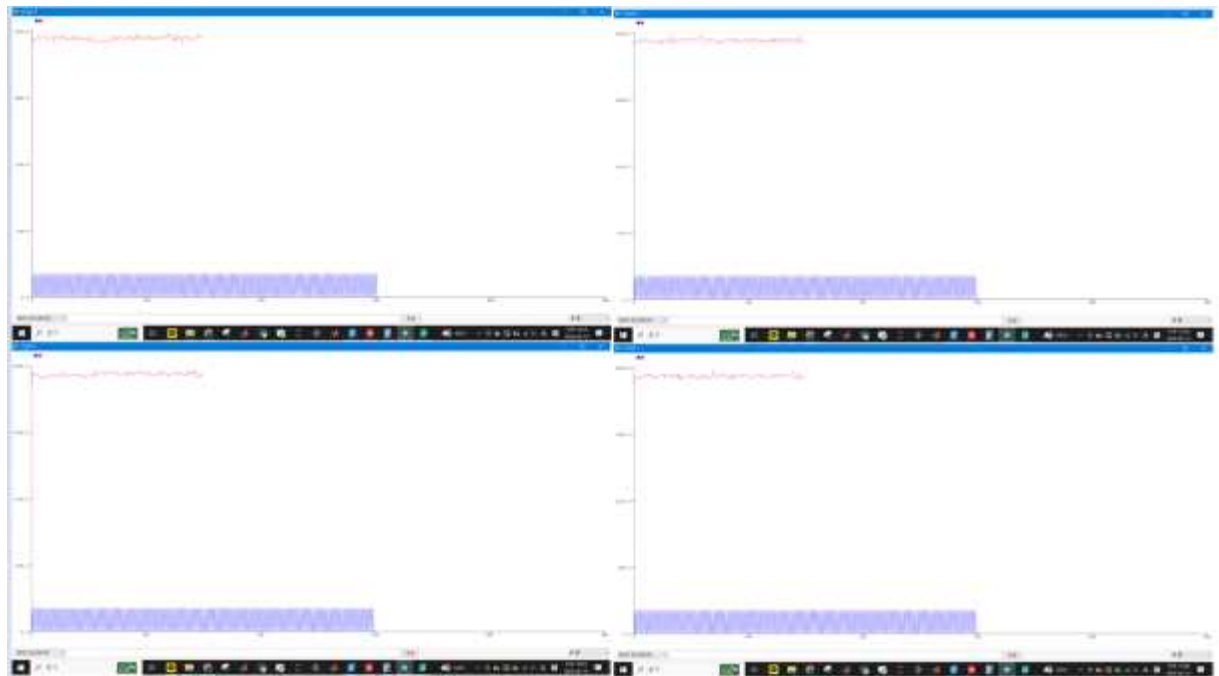


Fig. 66 Pd 제어 응답 선도 (2)

PD제어를 위해 $K_p = [1, 3, 5]$, $K_d = [10, 20, 30]$ 의 총 9가지 Case에 대한 응답을 관찰하였다. K_p 값이 증가 할수록 시스템이 빠르고 정확해지나, 너무 높으면 오버슈팅이 발생할 위험이 있고, K_d 값이 증가 할수록 시스템의 진동을 감소시켜 안정성을 향상시키나 너무 높다면 진동을 유발 할 수 있다. 이 점들과, 코드를 통해 관측한 응답을 기반으로 가장 안정한 형태라고 판단이 된 $K_p=0.5$, $K_d=1.4$ 를 PD제어를 위한 값으로 선정하였다. 선정된 값을 토대로 메카넘 휠 구동을 위해 모터의 속도를 PD제어를 통해 제어하였다. 모터 제어는 아두이노를 통해 모터 드라이버에 명령을 내림으로써 진행 되었다.

5.4 통신부

5.4.1 제품 알고리즘 개략도

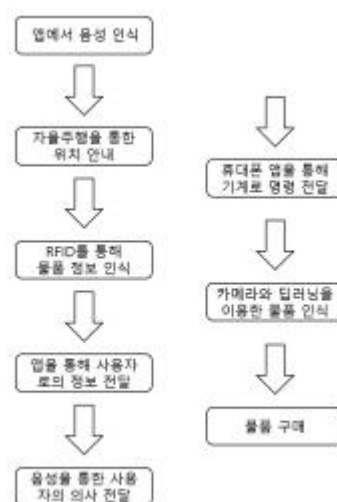


Fig. 67 제품 구동 과정을 간략하게 표현한 개략도

위 그림은 제품 구동 과정을 간략하게 나타낸 블록이다. 모든 음성 인식은 사용자의 휴대폰을 통해 진행되고, 이를 휴대폰에 설치된 어플리케이션을 이용하여 기계에 전달한다. 개략도 상 최초의 음성 인식을 통하여 기계는 사용자가 이용하기를 원하는 물품 코너를 인식한다. 기계는 인식한 물품 코너로 사용자를 안내한 후 각 물품 코너에 설치된 RFID를 통해 물품 코너에 판매중인 물품의 종류를 받아들여 사용자에게 전달한다. 기계로부터 전달받은 물품 정보를 토대로 구매를 희망하는 물품을 정하여 첫 번째와 동일하게 음성을 통해 기계로 사용자의 의사를 전달하고, 기계는 내장된 카메라와 딥러닝 프로그램을 통해 사용자가 구매를 희망하는 물품의 위치를 찾아준다.

5.4.2 앱 인벤터를 통한 아두이노와의 통신

사회적 약자를 위한 기계라는 설계 목적을 부합 시키기 위해, 본 기계는 사용함에 있어서 여러 편의성을 만족 시켜야 할 필요성이 있다. 구동 조건을 단순화시키기 위하여, 휴대폰을 통해 음성 등으로 기계에 직접적인 명령을 내릴 수 있도록 하였고, 그 과정에서 MIT App Inventor와 그로 개발한 어플리케이션을 사용 하였다. 휴대폰 어플리케이션은 기계에 설치 되어 있는 아두이노와 직접적으로 통신 함으로써 휴대폰으로 기계에 간편하게 명령을 내릴 수 있도록 설계되었고, 사용된 블록 코딩은 아래 그림과 같다.

번호	코드 역할	앱인벤터 코드
1	‘음성 인식 버튼’을 클릭하고 말한 텍스트를 인식한다.	
2	음성 인식 텍스트를 저장한다.	
3	‘보내기 버튼’을 클릭했을 때 ‘과일 코너’ 인식 되면 과일 코너로 이동한다 알리고, 인식 안되면 다시 음성인식이 다시 필요함을 알린다.	
4	음성 인식 텍스트를 앱에 출력하는 함수이다.	

5	블루투스 연결을 할 수 있도록 하는 함수이다.	
6	블루투스 연결을 끊도록 하는 함수이다.	
7	블루투스로 받은 텍스트를 앱으로 전송하는 함수이다.	
8	‘블루투스 연결 끊기 버튼’을 클릭했을 때 블루투스 연결을 끊고 앱에 알린다.	
9	‘전송! 버튼’을 클릭했을 때 ‘과일 코너’의 음성을 인식하여 123을 아두이노 메가로 전송하고 경로를 탐색한다 알린다.	
10	앱에서 주기적으로 블루투스를 통해 텍스트를 받는지 체크한다.	
11	블루투스 연결 기기의 주소, 이름을 받아와 블루투스를 활성화하도록 한다.	






12	‘상세정보 버튼’을 클릭했을 때 RFID를 통해 바나나와 토마토가 인식될 때 상세정보를 알리고, 만약 인식 되지 않았다면 아직 그 위치에 도달하지 않았음을 알린다.	
13	‘블루투스 연결 버튼’을 눌렀을 때, 블루투스 연결 된 후 연결됨을 알리고 음성인식 버튼을 누르도록 알린다.	
14	타이머 초기값을 0으로 지정한다.	
15	앱을 켤 때 타이머를 바로 활성화 시키지 않도록 한다.	
16	RFID가 인식된 후 타이머가 작동되며, 타이머가 18초 지나면 왼쪽으로 이동 후 회전한다 알린다.	

표 11. 아두이노 통신을 위한 블록 코딩

위와 같은 블록 코딩을 이용하여 실제 안드로이드 환경에서 실행이 가능한 어플을 개발하였고, 어플을 구동시킨 인터페이스 사진은 아래와 같다.

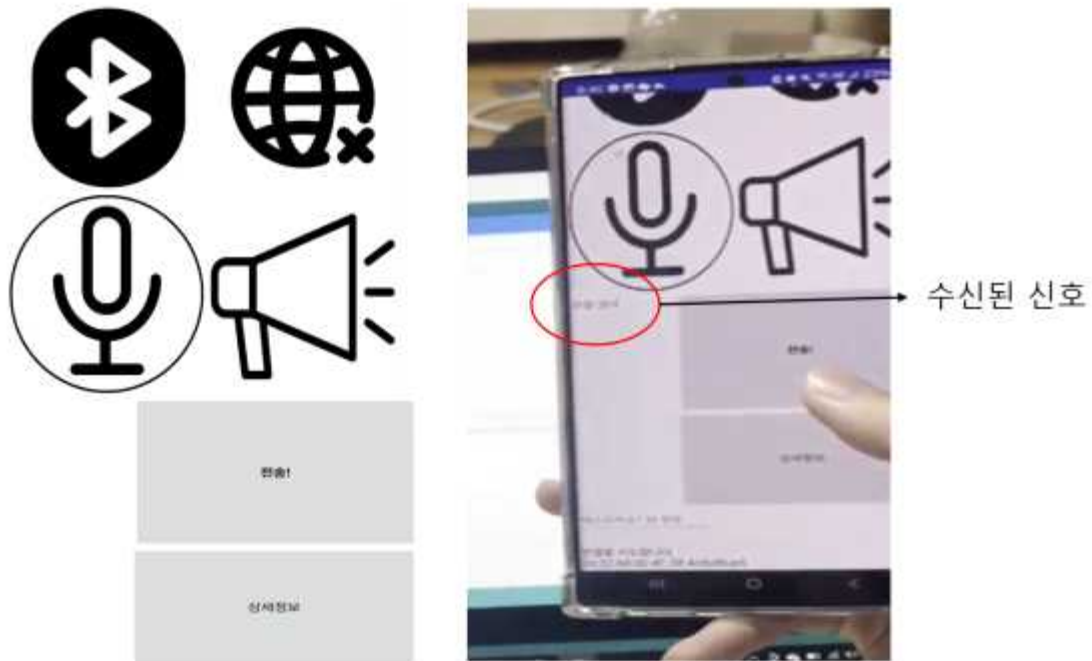


Fig. 68 어플 실행 화면(좌), 과일 코너 음성이 입력된 화면(우)




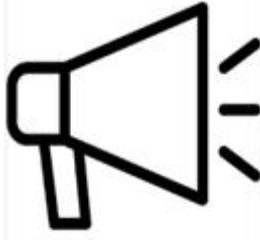


버튼 이름	버튼 모습
‘블루투스 연결 버튼’	
‘블루투스 연결 끊기 버튼’	
‘음성 인식 버튼’	
‘보내기 버튼’	
‘전송! 버튼’	
‘상세정보 버튼’	

표. 12 어플 실행 버튼

5.4.3 앱 사용법

1. '마트'앱을 실행시킨다. 이 과정은 타인이 앱을 터치하여 실행하거나 혹은 시각장애인 본인이 빅스비를 통해 앱을 실행하도록 할 수 있다.
2. '블루투스 연결 버튼'을 눌러 블루투스를 연결한다.
3. 블루투스 연결을 끊고싶으면 '블루투스 연결 끊기 버튼'을 누르면 된다.
4. 왼쪽 아래 '음성인식 버튼'을 누르고 과일 코너를 말한다.
5. '보내기 버튼'을 누르고 과일 코너가 잘 인식이 되었는지 확인하는 과정을 거친다. 잘 인식되었다면 오른쪽 밑의 '전송 버튼'을 눌러달라 하며, 잘 인식되지 않았다면 '음성인식 버튼'을 눌러 다시 인식해달라 알린다.
6. '전송!'버튼을 누르면 umm이 과일 코너를 향해 이동을 시작한다.
7. 과일 코너의 가판대의 과일에 RFID가 태그되었을 때 '상세정보'버튼을 누르면 그 과일의 상세정보(가격, 원산지, 등)을 들을 수 있다.
8. 상세정보를 앱이 읽는 동안 umm은 멈춰있어 구매자가 정보를 들을 시간을 주며, 이후 다시 움직임을 시작하므로 주의하라 앱에서 알린다.

5.5 센서부

5.5.1.1 RFID 작동 원리

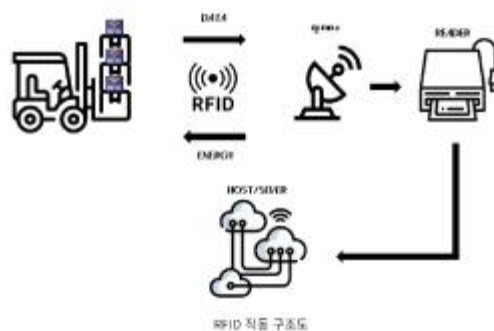


Fig. 69 RFID 작동 구조도

RFID는 안테나와 기판, 집적회로(IC)로 구성되어 있다. 회로가 외부 자기장을 받아들이면 안테나를 통해 외부와 통신을 하고 리더기를 통해 송신 받을 데이터를 해석하여 저장된 데이터를 불러오게 된다. 본 제품에서는 리더기로 송신받은 데이터를 CPU (Jatson Nano)로 받아들인 후 사용자에게 전송한다.

5.5.1.2 RFID 관련 코드

본 모빌리티는 목적 달성을 위해, 특정 위치에서 읽은 RFID 신호를 사용자에게 전달해야 할 필요가 있다. RFID 신호를 받아 들이고 블루투스 모듈을 통해 메시지를 전달하는 과정은 다음과 같은 코드를 통해 진행 된다.

번호	코드 역할	아두이노 우노 코드
1	SPI 통신 라이브러리 불러오기, MFRC522 라이브러리 불러오기, 소프트웨어 시리얼 라이브러리 불러오기	<pre>#include <SPI.h> #include <MFRC522.h> #include <SoftwareSerial.h></pre>
2	소프트웨어 시리얼 객체를 7(TX), 8(RX)으로 생성, reset핀은 9번으로 설정, SS핀은 10번으로 설정 (SS = Slave Selector)	<pre>SoftwareSerial BTSerial(7, 8); #define RST_PIN 9 #define SS_PIN 10</pre>
3	MFRC522를 이용하기 위해 mfrc 객체를 생성합니다.	<pre>MFRC522 mfrc(SS_PIN, RST_PIN);</pre>
4	시리얼 통신을 시작하고, 속도를 9600으로 설정합니다. 소프트웨어 시리얼 통신을 시작합니다. SPI 통신을 시작합니다. RFID 리더기를 초기화합니다.	<pre>void setup() { Serial.begin(9600); BTSerial.begin(9600); SPI.begin(); mfrc.PCD_Init(); }</pre>
8	"123"을 출력하는 상태를 제어하는 플래그입니다. RFID 태그 감지 후 123 출력을 다시 시작하기 위한 플래그입니다.	<pre>void loop() { static bool print123 = false; static bool resetPrint123 = false;</pre>
9	블루투스로부터 "123"을 수신하는지 확인합니다. 블루투스로부터 "123"을 수신하면 RFID 태그 감지 플래그를 재설정합니다.	<pre>if (BTSerial.available()) { String receivedData = BTSerial.readString(); if (receivedData == "123") { print123 = true; resetPrint123 = false; } }</pre>
10	"123"을 출력하는 상태입니다. "123"을 시리얼로 출력합니다. 1초의 딜레이를 줍니다.	<pre>if (print123) { Serial.println(123); delay(1000); }</pre>

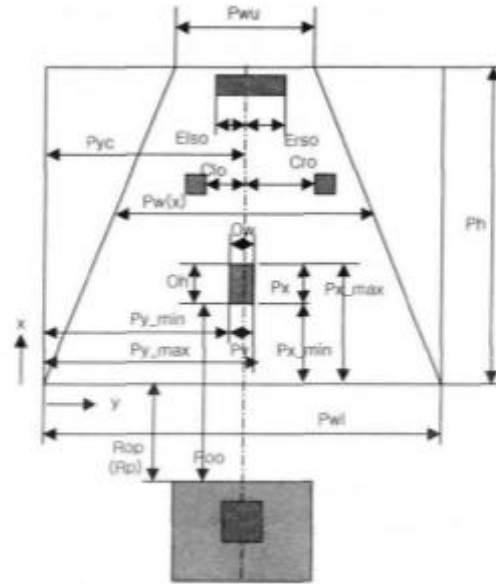
11	RFID 태그가 감지되면 "123" 출력 상태를 해제합니다. 27을 찾았는지 여부를 나타내는 플래그입니다. 16을 찾았는지 여부를 나타내는 플래그입니다.	<pre> if (mfrc.PICC_IsNewCardPresent()) && mfrc.PICC_ReadCardSerial() { print123 = false; Serial.print("Card UID: "); bool found27 = false; bool found16 = false; </pre>
12	UID 값을 출력하고 특정 값이 있는지 확인하고 플래그를 설정합니다.	<pre> for (byte i = 0; i < mfrc.uid.size; i++) { Serial.print(mfrc.uid.uidByte[i]); Serial.print(" "); </pre>
13	특정 값이 있는지 확인하고 플래그를 설정합니다.	<pre> if (mfrc.uid.uidByte[i] == 27) { found27 = true; } if (mfrc.uid.uidByte[i] == 16) { found16 = true; } } Serial.println(); </pre>
14	27을 찾았을 경우 블루투스로 데이터를 전송하고 시리얼 모니터에 메시지를 출력합니다.	<pre> if (found27) { BTSerial.write("토마토 상세정보\n"); Serial.println("토마토 상세정보"); } </pre>
15	16을 찾았을 경우 블루투스로 데이터를 전송하고 시리얼 모니터에 메시지를 출력합니다.	<pre> if (found16) { BTSerial.write("배나나 상세정보\n"); Serial.println("배나나 상세정보"); } </pre>
16	UID를 읽은 후 태그를 정지합니다. RFID 태그 감지 후 123 출력을 다시 시작하도록 설정합니다.	<pre> mfrc.PICC_HaltA(); resetPrint123 = true; } } </pre>

표. 13 RFID 및 BLUETOOTH 아두이노 코드

5.5.2 LIDAR 작동 원리

본 모빌리티에서는 회전형 라이다 센서를 사용할 것이다. 회전형 라이다 센서는 360도를 돌며 펄스 레이저를 송신 및 수신하여 주변의 객체를 인식하고 객체가 얼마나 멀리 있는지에 대한 거리를 측정한다. 펄스 레이저를 몇차원으로 송신하는지가 인식할 수 있는 차원을 결정한다. 차원이 높을수록 센서의 가격이 비싸지므로 본 모빌리티에서는 2D LIDAR 센서로 결정하였다. 이미 자율주행을 하기 위해서 LIDAR 센서는 필수인 부품이며, 인간의 눈 역할을 대신함을 알 수 있다.

전면 카메라를 이용해 정면에 있는 장애물의 좌표를 구하려고 한다. 장애물의 좌표는 모빌리티와 장애물 간의 상대 좌표를 이용하여 구할 수 있다. 한편, 모빌리티의 좌표는 SLAM을 이용하여 얻을 수 있다. 이는 모빌리티와 장애물 간 상대적 좌표를 알기 위해 참고한 논문의 파라미터이다.



Rw	x거리에 따른 y방향의 기준길이
Rp	로봇 앞면으로부터 사진의 밑면까지의 거리
Rho	x거에 따른 h방향의 기준길이
Rh	사진의 x방향(높이)의 기준길이
Pwl	사진의 아래 부분에서 y방향(폭)
Ph	사진의 x방향(높이)의 기준길이(Rh) 내에서의 전체 픽셀수
Pwu	사진의 위부분에서 y방향(폭)의 기준거리(Rw) 내에서의 전체 픽셀수
Phl	사진의 앞에 있는 기준실험물의 길이(Rho) 내의 전체 픽셀수
Phh	두 기준실험물 사이의 실거리 내의 전체 픽셀수
Phu	사진 뒤에 있는 기준실험물의 길이(Rho)내의 전체 픽셀수
Pyc	사진의 좌측으로부터 로봇의 중심선까지의 길이

Fig. 70 상대적 좌표를 알기 위해 측정해야 할 파라미터

28) 이동로봇을 위한 카메라 1대를 이용한 소형 장애물 인식방법에 관한 연구

지널정보: [Korean Society for Precision Engineering, Journal of the Korean Society for Precision Engineering](#), 학술지별: [한국정밀공학회지 제22권 제9호](#), 2005.9 85 - 92 (8page), 김갑순

다음은 Python에서 OpenCV를 사용하여 BMP 파일을 읽고, 2차원 영상처리를 통해 소형 장애물의 크기와 로봇의 위치를 추출하는 간단한 예제 코드이다. 예제 코드를 활용하여 위 파라미터와 식을 입력하고 거리를 추정할 것이다.

```
import cv2
import numpy as np

def find_obstacles_and_robot_position(image_path):
    # BMP 파일 읽기
    image = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE)

    # 이미지 전처리 (이진화 등)
    _, binary_image = cv2.threshold(image, 128, 255, cv2.THRESH_BINARY)

    # 소형 장애물과 로봇의 위치를 찾기 위한 연산 수행
    # 여기서는 예시로 윤곽선을 찾는 것으로 가정
    contours, _ = cv2.findContours(binary_image, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    obstacles = []
    robot_position = None

    for contour in contours:
        # 윤곽선의 경계 상자 계산
        x, y, w, h = cv2.boundingRect(contour)

        # 장애물의 크기와 위치를 리스트에 추가
        obstacles.append({'width': w, 'height': h, 'position': (x + w/2, y + h/2)})

    # 로봇의 위치를 중심으로 잡음 (로봇을 감지할 수 있는 다른 방법을 사용하려면 수정 필요)
    if len(contours) > 0:
        x, y, w, h = cv2.boundingRect(contours[0])
        robot_position = (x + w/2, y + h/2)

    return obstacles, robot_position

# BMP 파일 경로 설정
image_path = 'your_image.bmp'

# 함수 호출
obstacles, robot_position = find_obstacles_and_robot_position(image_path)

# 결과 출력
print('Detected Obstacles:')
for obstacle in obstacles:
    print(f'Width: {obstacle['width']}, Height: {obstacle['height']}, Position: {obstacle['position']}')

print('\nRobot Position:')
print(robot_position)
```

Fig. 72 소형 장애물의 크기와 로봇의 위치를 추출하는 코드

이를 이용해 장애물 회피 경로 설정 알고리즘을 작성 할 수 있다.

장애물 위치와 로봇의 위치가 Fig. 66과 같다고 가정하자. 전면 카메라를 이용하여 장애물의 좌표를 구해낼 수 있다.

따라서 다음과 같은 방식으로 경로를 설정할 수 있다. 이는 장애물이 인식이 되었을 때 작동하는 코드로 구성할 것이다.

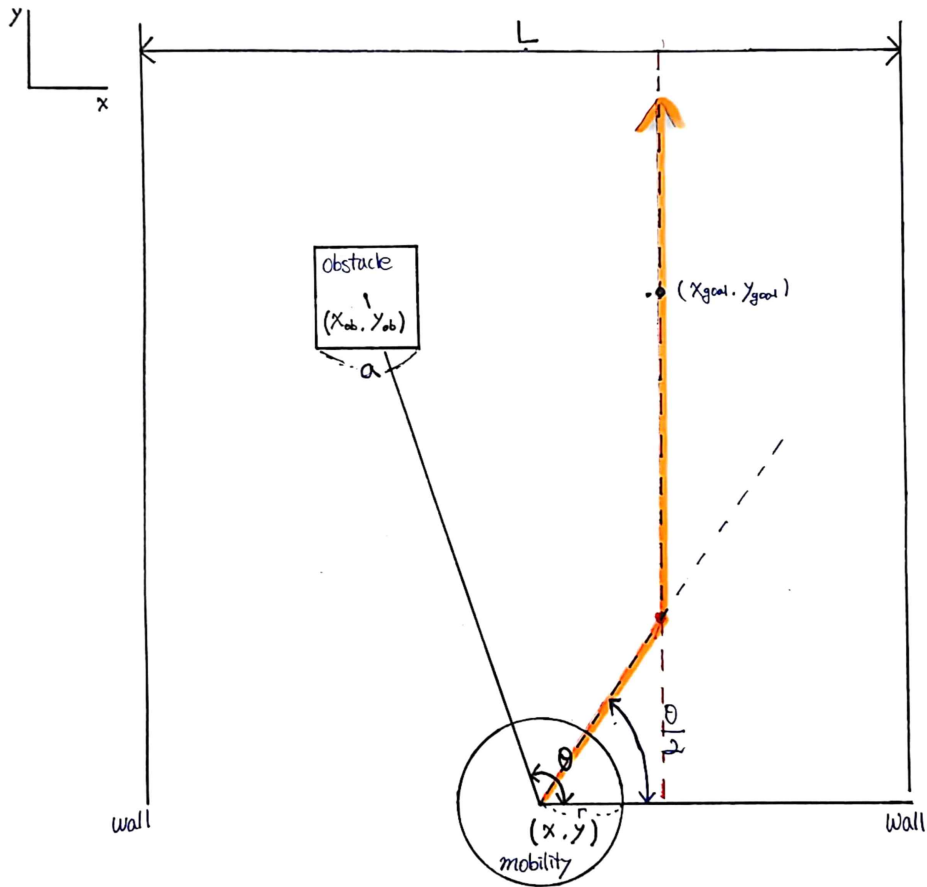


Fig. 73 모빌리티의 장애물을 피하는 알고리즘 좌표 설정

Fig. 69을 참고하여 변수를 정의해보면 다음과 같다:

$$(\text{상대 좌표 } \vec{u}) = (x - x_{ob}, y - y_{ob})$$

$$(\text{로봇의 진입각도}) \theta = 90^\circ + \tan^{-1} \left(\frac{x_{ob} - x}{y_{ob} - y} \right)$$

$$(\text{벽과 벽 사이의 거리}) = L$$

이 때, 장애물은 정사각형으로 설정하였으며 한 변의 길이는 a로 설정하였다.

또한, 모빌리티는 원으로 설정하였으며 한 변의 길이는 r로 설정하였다.

다음과 같이 동작원리를 생각해보았다 : 장애물이 인식이 되면 코드 구동 시작.

1) $L - x_{ob} < \frac{L}{2}$ 일 때

만약, $L - x_{ob} < \frac{L}{2}$ 이면, 오른쪽 경로가 공간이 더 많으므로 모빌리티는 오른쪽 경로를 선택한다.

2) $L - x_{ob} - 2r < 0$ 일 때

만약, $L - x_{ob} - 2r < 0$ 이면, 공간이 있음에도 모빌리티가 지나갈 공간이 없으므로 멈춘다.

3) $L - x_{ob} - 2r > 0$ 일 때

모빌리티가 장애물을 회피하여 지나갈 수 있다.

모빌리티는 $\frac{\theta}{2}$ 방향을 설정한 후, x 좌표가 $(L - x_{ob} - \frac{a}{2})\left(\frac{1}{2}\right)x$ 이 될 때까지 주행한 후 벽에 평행하게 회전하여 다음 목표지점까지 계속 직진한다. 목표지점 (x_{goal}, y_{goal}) 에 도달했을 때 위 알고리즘은 구동이 종료된다.

또 다른 목표지점 (x_{goal2}, y_{goal2}) 이 입력되면, 위와 같은 과정을 반복한다.

5.5.4 카메라 2 (측면) 작동 원리²⁹⁾

측면카메라를 이용하여 마트 물품을 인식을 하고자 한다. 다음 그림은 마트 물품을 인식하는 과정의 간략한 그림이다.

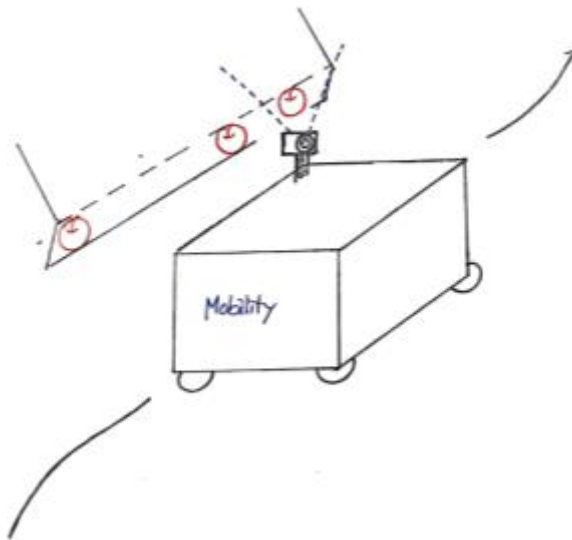


Fig. 74 측면 카메라를 이용한 마트 물품 인식

마트 물품 인식 및 분류를 위한 전처리:

29) <https://kr.mathworks.com/discovery/convolutional-neural-network-matlab.html>

컴퓨터는 이미지를 각 픽셀값을 가진 숫자배열로 인식하기 때문에 각 이미지 안의 마트 물품을 인식하고 분류하기 위해서는 전처리 과정이 필요하다. 다음은 일반적인 전처리 일련의 과정을 설명한 것이다.

1) 이미지 크기 조정 및 정규화

이미지의 크기를 일관성있게 조정하며 정규화를 진행한다. 이를 통해 이미지 차원을 줄이고 용량을 줄일 수 있으며 이미지 분류를 위한 필요한 픽셀정보만 담을 수 있도록 하였다. 정규화는 0과 1사이로 진행한다.

2) 레이블 인코딩

분류 문제에서 레이블을 모델이 이해할 수 있는 형식으로 변환한다. 일반적으로 원-핫 인코딩을 사용하며 이는 집합의 크기를 벡터차원으로 하며 표현하고 싶은 값의 인덱스에 1의 값을 부여하며 다른 인덱스에는 0을 부여하는 벡터 표현방식을 뜻한다.

3) 데이터 증강

데이터 증강이란 기존의 훈련데이터를 변형하여 새로운 데이터를 생성하는 과정을 뜻한다. 이는 학습 모델의 성능 향상 및 과적합 방지에 도움을 준다.

6. 제작개요



6.1 변경사항

설계 변경 전	설계 변경 후
4채널 모터 드라이버	리튬 이온 배터리 팩
	
STX30L-BS (12V 30AH)	[MDD3A] 3Amp 4V-16V DC Motor Driver (2 Channels)

설계 변경 내용

이전 설계 방식에서는 오토바이에 쓰이는 로켓트 배터리를 사용하기로 하였다. 하지만 무게가 무거워 모터 축 구동에 영향이 생길 것이라고 판단하였고 크기가 166*127*175로 큰 규격에 속하기 때문에 프로파일 및 배선 연결 시 방해가 될 것이라고 생각하였다. 따라서 무게가 0.45kg이고 크기가 51*72*73으로 비교적 이전 선정 배터리보다 양호한 편에 속하는 배터리로 변경하기로 하였다. 2채널 배터리팩이어서 배터리를 두 개를 사용해야 한다는 단점이 있었지만 보완 내용에는 지장이 없고, 무게와 규격 측면에서도 기회비용이 있다고 판단하였다.

표. 14 설계 변경 사항 (1)

설계 변경 전	설계 변경 후
4채널 모터 드라이버	2채널 모터드라이버
	
1.5Amp 7V-25V DC Motor Driver	[MDD3A] 3Amp 4V-16V DC Motor Driver (2 Channels)

설계 변경 내용

이전 설계 방식에서는 4채널 모터 드라이버를 사용하여 모터 4개를 하나의 모터드라이버에 연결하고자 하였다. 하지만 설계변경사항(1)에 따라 배터리를 2채널로 변경하였기 때문에 모터드라이버도 2채널로 변경하였다. 2채널로 변경함에 따라 앞바퀴 모터 2개에 2채널 모터드라이버 1개, 뒷바퀴 모터 2개에 2채널 모터드라이버 1개를 연결하기로 하였다. 변경에 따라 비용적인 측면에서 좋다고 판단하였다. 또한, 4채널 모터 드라이버가 더 많은 채널을 제어하기 때문에 더 많은 전력을 소비하는데 배터리 작동 시간이나 전원 공급에 대한 측면에서 2채널 모터드라이버가 더 좋다고 판단하였다.

표. 15 설계 변경 사항 (2)




설계 변경 전	설계 변경 후
초음파 센서 사용하지 않음	초음파 센서 사용함
-	
-	HC-SR04 / Arduino Ultrasonic
설계 변경 내용	
<p>이전 설계 방식에서는 초음파 센서를 사용하지 않았다. 하지만 자율주행에서는 센서가 많을수록 정확성이 증가한다. 초음파 센서는 아두이노용 초음파 센서를 사용하였고 그 코딩 과정이 간단하기 때문에 초음파 센서를 사용해도 시간적인 소모가 별로 없을 것이라고 판단하여 초음파센서를 추가하였다. 또한 비용도 1000원 내외로 비용적인 측면에서도 적절하다고 판단하였다. 초음파센서를 추가함으로써 전방, 우측면 장애물 인지 및 거리 탐지 성능이 향상될 것이라고 초음파센서를 추가하는 것으로 설계 내용을 수정하였다.</p>	


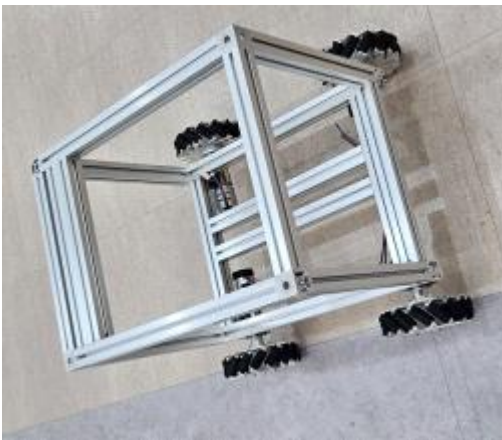
표. 16 설계 변경 사항 (3)


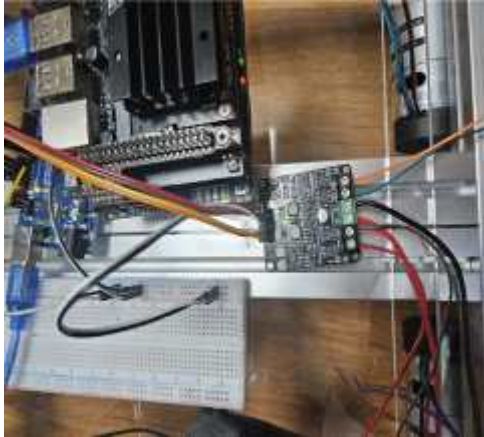
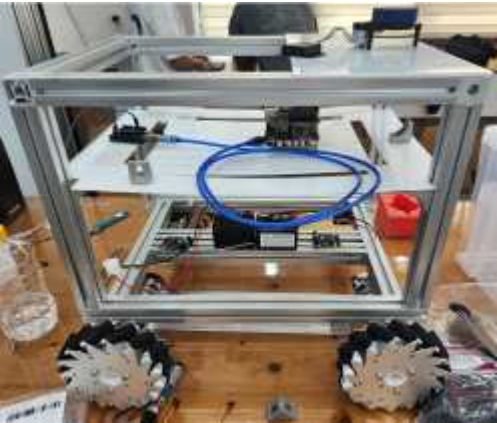
설계 변경 전	설계 변경 후
아두이노 우노	아두이노 메가 2560 + 아두이노 우노
	
아두이노 우노 Uno R3 호환 보드 CH340 ATMEGA328P	아두이노 메가 2560 R3 호환 보드 CH340 [SZH-EK375] + 아두이노 우노 Uno R3 호환 보드 CH340 ATMEGA328P
설계 변경 내용	

이전 설계 방식에서는 아두이노 우노를 사용하기로 하였는데 아두이노 메가를 사용하는 방식으로 설계 내용이 바뀌었다. 모빌리티 특성상 많은 갯수의 디지털 핀, 인터럽트 핀, PWM핀을 사용하여야 하는데, 4개의 모터, 2개의 모터드라이버, 초음파 센서를 사용하려면 아두이노 우노로는 핀이 부족하였다. 따라서 아두이노 메가와 아두이노 우노를 같이 쓰는 것으로 재선정하였다. 또한 모빌리티 구동방식에서 다중 통신이 사용되는데 아두이노 메가가 시리얼 포트 또한 더 많이 가지고 있기 때문에 다중 통신을 사용하는 모빌리티에 더 유용하다고 판단하였다. 결국 아두이노 메가에는 모터드라이버, 초음파 센서 등을 이용하고 아두이노 우노에는 블루투스 모듈과, RFID모듈을 사용하기로 하였다. 젯슨나노에 아두이노 우노와 아두이노 메가 둘 다 연결할 포트가 필요한데 젯슨나노의 포트는 총 3개이며, 라이다, 아두이노 우노, 아두이노 메가를 연결할 수 있었다.

표. 17 설계 변경 사항 (4)

6.2 제작과정

번호	내 용	사 진
1	상부 프레임 조립 및 모터, 브라켓 결합	
2	메카넘 휠 결합, 외장 틀 완성	

번호	내 용	사 진
3	엔코더 모터, 모터드라이버, 아두이노 메가, 브레드보드 연결	
4	모터드라이버의 배선 및 Jetson Nano 배선	
5	Jetson Nano, YDLidar X2, 위치 설정 및 결합	


번호	내 용	사 진
6	Fig. Jetson Nano, YD Lidar X2 연결	

표. 18 제작 과정

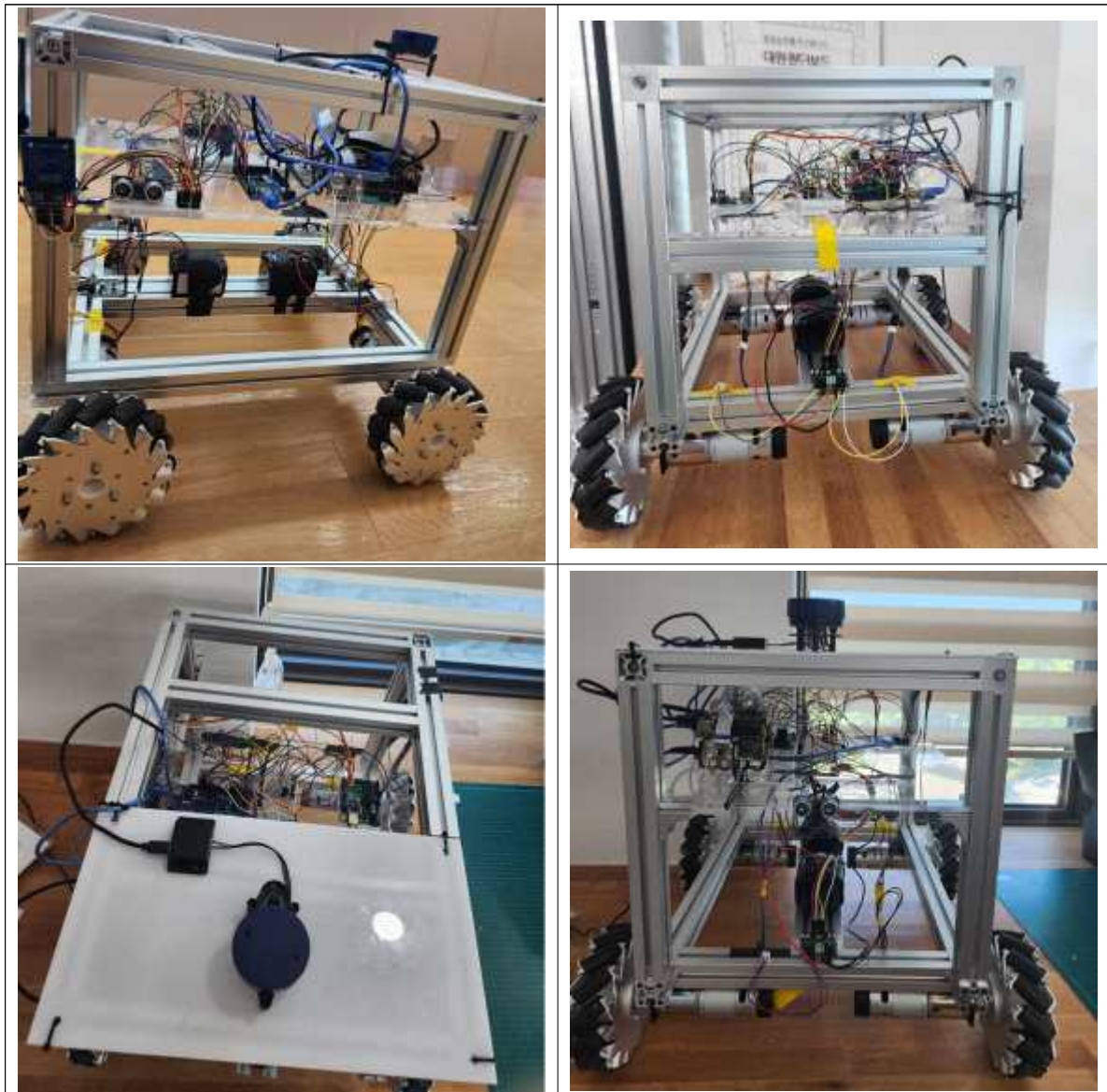


표. 19 최종 모빌리티 형상

7. 기타 설계

7.1 통신 전용 어플 설계

7.1.1 어플리케이션 제작 의도

시각 장애인을 보조하는 기계를 제작함에 있어 가장 중요 한 건 시각 외의 자극을 통해 시각 장애인에게 정보를 전달하는 것이다. 청각, 후각, 촉각, 통각 등의 후보들이 있었는데 그 중 청각에의 전송을 위해 어플리케이션 개발을 결정했다. 시각 장애인에게 스마트폰은 생활을 보조함에 있어 가장 큰 지분을 차지하는 도구이다. 그에 따라 시각 장애인에게 가장 친숙하고 몸에서 떨어지지 않는 것들 중 하나이다. 따라서 현실적인 방안인 청각과 촉각 중, 사용자에게 있어 익숙한 스마트폰 앱과 무선 이어폰을 활용하여 청각을 통한 정보 전달을 하고자 어플리케이션 개발을 하게 되었다. 또한 이 어플은 사용해 사용자에게 정보를 전달하는 것 뿐만 아니라, 사용자의 의지를 기계에 전달하는 용도로도 사용 될 예정이다.

7.1.2 어플리케이션의 역할

어플리케이션은 제작 목적에 따라 아래의 역할을 수행 하게 된다.

- ① 최초 기계와 접속 시 배터리 잔량을 음성으로 사용자에게 전달
- ② 사용자가 이용을 원하는 구역(ex. 육류코너, 야채코너)를 음성을 통한 입력
- ③ 도착한 구역의 물품 정보를 사용자에게 음성을 통한 전달
- ④ 구입을 희망하는 상세 물품 품목을 음성을 통해 기계로 입력

7.1.3 어플리케이션 개발 과정

본 어플리케이션은 사용자(휴대폰)과 기계(Jatson NaNo)간의 간단한 양방향 소통의 역할의 수행을 목적으로. MIT App Inventor를 이용하여 개발 할 예정이다. MIT App Inventor는 복잡한 프로세스를 가진 어플의 개발에는 한계가 있지만, 위와 같은 간단한 통신의 역할은 초보자도 충분히 할 수 있는 플랫폼이기 때문에 개발 툴로써 선정하게 되었다. 이 과정은 5.4.2에서 설명한 바가 있다.

7.2.3. 기능의 상세 설계

배터리 잔량은 사용자에게 %단위로 환산되어 안내 될 예정이고, 상세 과정은 다음과 같다.

- ① 전압계를 이용하여 배터리의 개방회로 전압을 측정한다.
- ② 배터리의 SoC에 따른 개방회로 전압을 참고하여, 현재 전압을 기반으로 SoC를 계산한다. 계산에는 아래와 같은 식이 사용 된다.

$$\frac{12.65 - OCV_R}{12.65 - 11.89} \times 100(\%)$$

- ③ 계산된 SoC를 6.1의 어플리케이션을 통해 사용자(핸드폰)에게 전송한다.
- ④ 핸드폰과 연동되어 있는 유.무선 이어폰을 통해 사용자에게 현재 배터리의 상황(SoC)를 음성을 통해 전달한다.

8. 모빌리티 구동 S/W 설계

8.1 회로도(연결방식) 설계

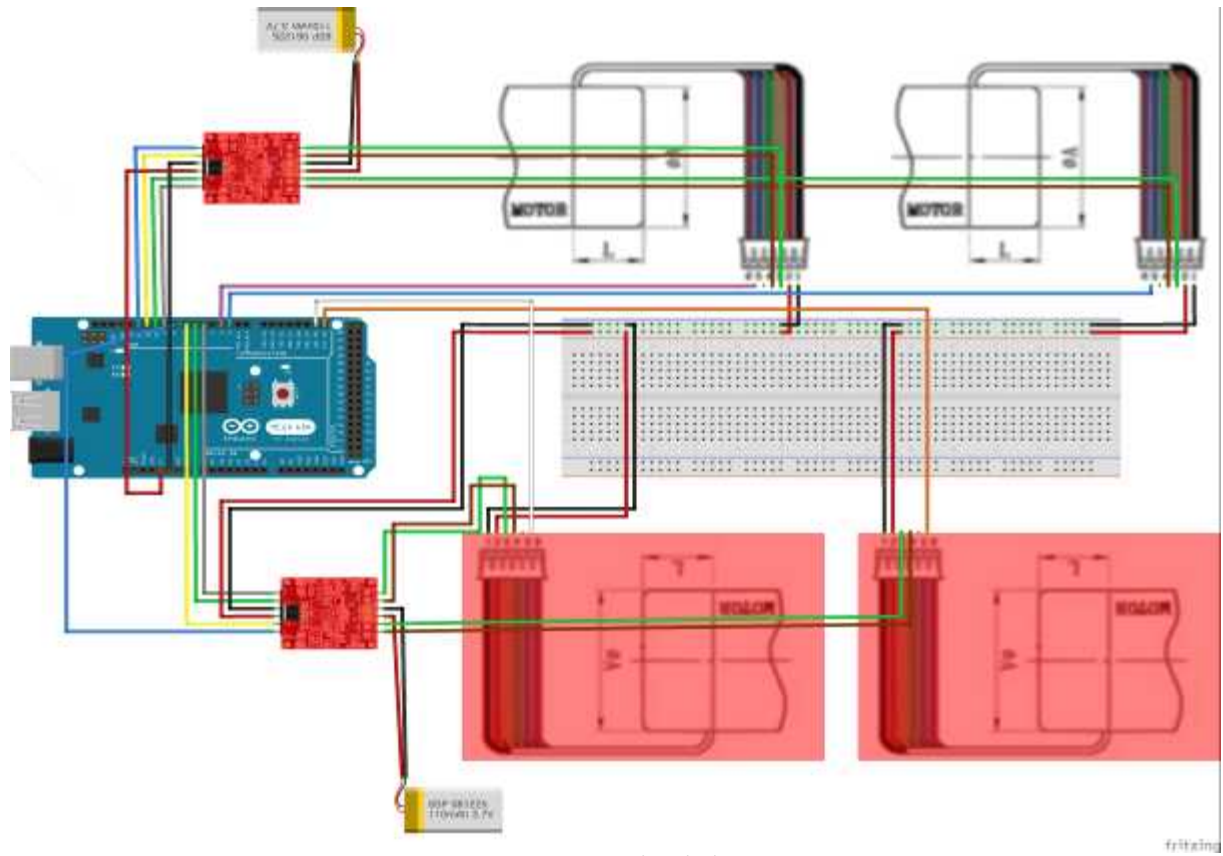


Fig. 75 모터 배선도

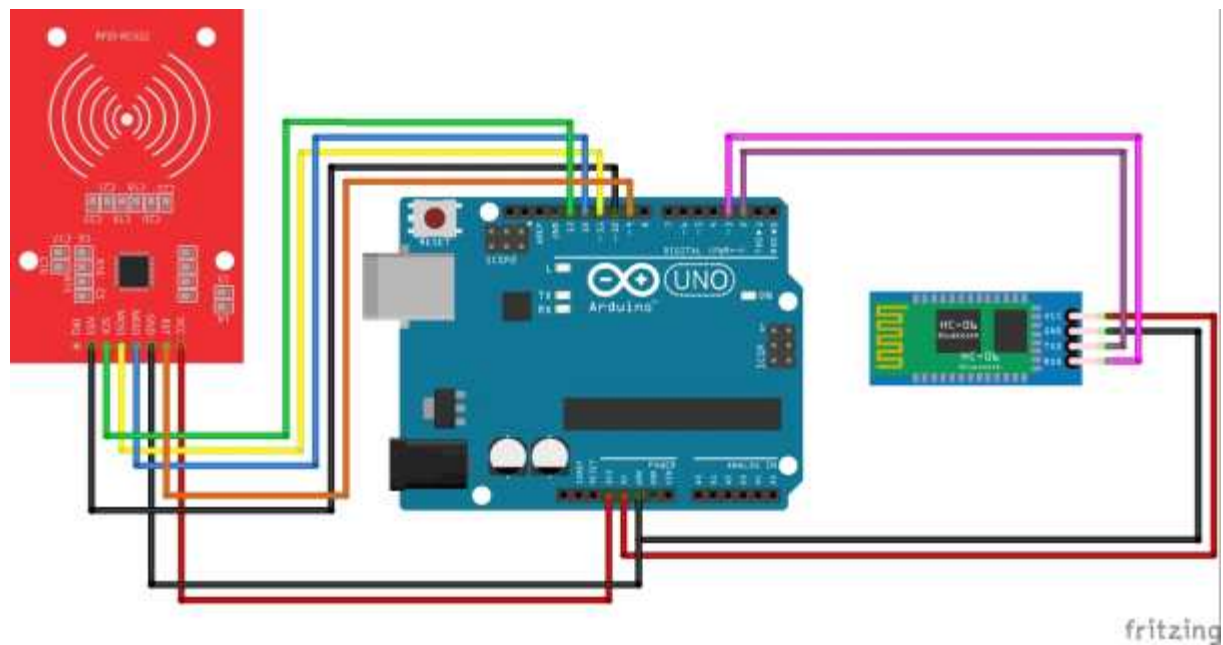


Fig 76 RFID와 블루투스 연결 선도

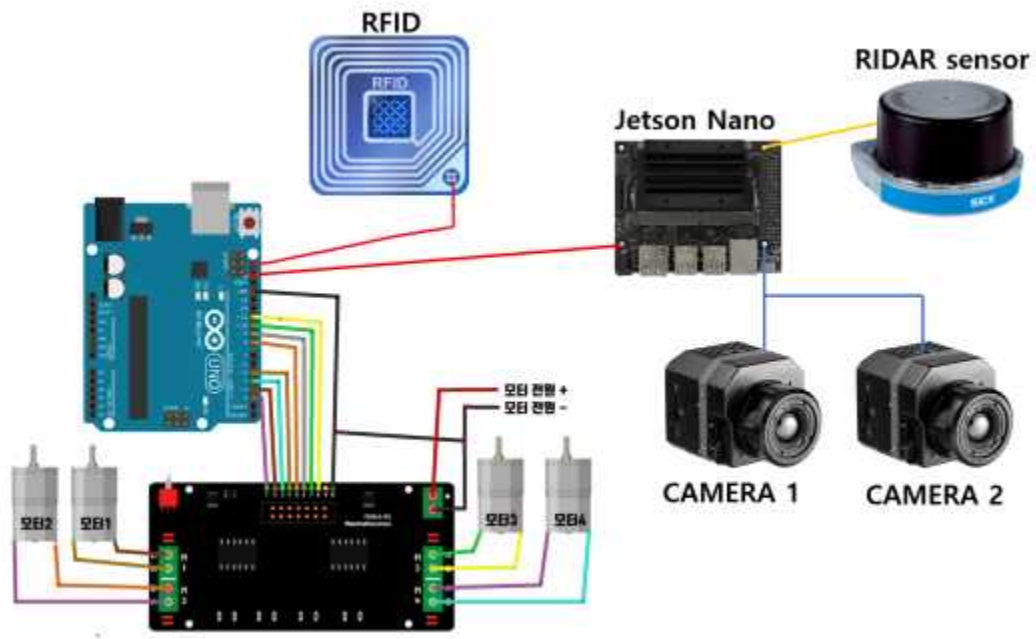


Fig. 77 주요 부품 간 연결 방식 정리

8-2-1 모빌리티 구동 코딩

번호	명령 설명	아두이노 코드
1	구동과 통신에 필요한 라이브러리를 불러온다.	<pre>#include <ros.h> #include <std_msgs/Int32.h> #include <CytronMotorDriver.h></pre>
2	PID제어에 쓰이는 모터의 엔코더를 아두이노와 연결한다.	<pre>// PID 제어 관련 상수 및 변수 #define ENCODER_A1 20 #define ENCODER_A2 21 #define ENCODER_A3 2 #define ENCODER_A4 3</pre>
3	초음파센서 2개를 아두이노와 연결한다.	<pre>#define TRIG 18 // 초음파 센서의 트리거 핀 #define ECHO 19 // 초음파 센서의 에코핀 #define TRIG1 16 // 초음파 센서의 트리거 핀 #define ECHO1 17 // 초음파 센서의 에코핀</pre>
4	비례 이득과 미분 이득을 지정하고, 오차 값을 저장하는 배열을 만든다.	<pre>// pd_control value float kp = 0.1; float kd = 0.05; float prevError[4];</pre>

5	DC모터와 연결된 모터드라이버를 아두이노와 연결한다.	<pre>// 모터 객체 배열 생성 CytronMD motors[] = { CytronMD(PWM_PWM, 5, 6), // 왼쪽 앞 바퀴 모터 CytronMD(PWM_PWM, 7, 8), // 오른쪽 앞 바퀴 모터 CytronMD(PWM_PWM, 9, 10), // 왼쪽 뒷 바퀴 모터 CytronMD(PWM_PWM, 11, 12) // 오른쪽 뒷 바퀴 모터 };</pre>
6	모터의 기본 속도를 설정한다. (-255부터 +255)	<pre>// 기본 속도 설정 const float w = 34; float speeds[] = {w, w, w, w}; // 모터 속도 배열</pre>
7	엔코더와 관련된 변수들을 선언하고 초기화한다.	<pre>// encoder zero const volatile int encoderA[4] = {0, 0, 0, 0}; float direct[4] = {0, 0, 0, 0}; unsigned long timePrev = 0; unsigned long timeCurr = 0; float x = 0; int move_default = 0;</pre>
8	초음파 센서로 측정한 거리를 저장한다.	<pre>// 초음파 value volatile unsigned long distanceright; volatile unsigned long distancefront;</pre>
9	ROS노드에서 통신을 설정하고, ROS메시지를 퍼블리시하기 위한 준비를 한다.	<pre>// ros node 통신 ros::NodeHandle nh; // ros구동 std_msgs::Int32 outputMessage; ros::Publisher pub("output", &outputMessage);</pre>

10	<p>모빌리티가 벽을 따라 이동할 때 사용할 로직을 포함한다. (초음파 센서 사용)</p>	<pre>// 첫번째 과정(벽따라 이동) void callBackFunction1(const std_msgs::Int32 &inputMessage) { if (inputMessage.data == 1 && move_default==0) { scanright(); scanfront(); // 앞쪽 거리 조건을 우선적으로 처리 else if (distancefront > 0 && distancefront <= 30) { sd(); } else { // 오른쪽 거리 조건을 처리 if (distanceright <= 25) { cross1(); } else if (distanceright >= 35) { right(); } else if (distanceright >= 25 && distanceright < 35) { straight(); } } } }</pre>
11	<p>특정 조건에 따라 로봇의 이동을 제어하는 로직을 포함하는 콜백함수를 설정한다.</p>	<pre>if (inputMessage.data == 1 && move_default==1) { scanright(); scanfront(); // 앞쪽 거리 조건을 우선적으로 처리 if (x<0){ speeds[4]={0,0,0,0}; } else if (distancefront > 0 && distancefront <= 30) { sd(); } else { // 오른쪽 거리 조건을 처리 if (distanceright <= 25) { cross1(); } else if (distanceright >= 35) { right(); } else if (distanceright >= 25 && distanceright < 35) { straight(); } } } }</pre>
12	<p>특정 조건에 따라 로봇의 회전 및 이동을 제어하는 로직을 포함한다.</p>	<pre>// 두번째 과정(회전) void callBackFunction2(const std_msgs::Int32 &inputMessage) { if (inputMessage.data == 2) { scanright(); scanfront(); if (distanceright > 8) { right(); } else { left(); sd(); rotate2(); delay(9300); move_default=1; outputMessage.data=1; pub.publish(&outputMesaage); } } }</pre>

13	ROS환경에서 메시지를 수신하는 Subscriber를 설정한다.	<pre> ros::Subscriber<std_msgs::Int32> sub1("input1", &callbackFunction1); ros::Subscriber<std_msgs::Int32> sub2("input2", &callbackFunction2); ros::Subscriber<std_msgs::Int32> sub3("input3", &callbackFunction3); </pre>
14	초음파 센서와 엔코더를 설정하고, ROS를 사용하여 통신한다.	<pre> void setup() { pinMode(TRIG, OUTPUT); pinMode(ECHO, INPUT); pinMode(TRIG1, OUTPUT); pinMode(ECHO1, INPUT); // encode setup pinMode(ENCODER_A1, INPUT_PULLUP); pinMode(ENCODER_A2, INPUT_PULLUP); pinMode(ENCODER_A3, INPUT_PULLUP); pinMode(ENCODER_A4, INPUT_PULLUP); // interrupt attachInterrupt(digitalPinToInterrupt(ENCODER_A1), ISR_encoderA0, FALLING); attachInterrupt(digitalPinToInterrupt(ENCODER_A2), ISR_encoderA1, FALLING); attachInterrupt(digitalPinToInterrupt(ENCODER_A3), ISR_encoderA2, FALLING); attachInterrupt(digitalPinToInterrupt(ENCODER_A4), ISR_encoderA3, FALLING); nh.initNode(); nh.advertise(pub); nh.subscribe(sub1); nh.subscribe(sub2); nh.subscribe(sub3); } </pre>
15	엔코더의 A채널에서 발생하는 인터럽트를 처리하는 함수를 정의한다.	<pre> // encoder input void ISR_encoderA0() { encoderA[0]++; } void ISR_encoderA1() { encoderA[1]++; } void ISR_encoderA2() { encoderA[2]++; } void ISR_encoderA3() { encoderA[3]++; } </pre>
16	각 모터의 속도를 설정하는 함수를 정의한다.	<pre> // 모터 속도 설정 함수 void setMotorSpeeds() { for (int i = 0; i < 4; i++) { motors[i].setSpeed(speeds[i] * direct[i]); } } </pre>

17	모빌리티의 모터를 제어하는 세 가지 함수를 정의한다.(직진, 우방향, 좌방향, 평행직진, 좌방향, 평행직진)	<pre>// 모터 제어 함수들 void straight() { float temp[] = {1, 1, 1, 1}; memcpy(direct, temp, sizeof(direct)); setMotorSpeeds(); delay(1100); } void right() { float temp[] = {1, -1, -1, 1}; memcpy(direct, temp, sizeof(direct)); setMotorSpeeds(); delay(1800); sd(); } void left() { float temp[] = {-1, 1, 1, -1}; memcpy(direct, temp, sizeof(direct)); setMotorSpeeds(); delay(5000); sd(); }</pre>
18	모빌리티의 모터를 제어하는 세 가지 함수를 정의한다.(직좌, 대각선 방향, 직우 대각선 방향, 반시계방향회전, 정지)	<pre>void cross1() { float temp[] = {0, 1, 1, 0}; memcpy(direct, temp, sizeof(direct)); setMotorSpeeds(); delay(1000); sd(); } void cross2() { float temp[] = {1, 0, 0, 1}; memcpy(direct, temp, sizeof(direct)); setMotorSpeeds(); delay(100); sd(); } void rotate2() { float temp[] = {-1, 1, -1, 1}; memcpy(direct, temp, sizeof(direct)); setMotorSpeeds(); delay(1100); } void stop() { float temp[] = {0, 0, 0, 0}; memcpy(direct, temp, sizeof(direct)); setMotorSpeeds(); }</pre>
19	초음파센서를 이용하여 전방 및 우측의 거리를 측정하는 함수들을 정의한다.	<pre>void scanright() { digitalWrite(TRIG, LOW); delayMicroseconds(2); digitalWrite(TRIG, HIGH); delayMicroseconds(10); digitalWrite(TRIG, LOW); distanceRight = pulseIn(ECHO, HIGH) / 58.2; } void scanfront() { digitalWrite(TRIG1, LOW); delayMicroseconds(2); digitalWrite(TRIG1, HIGH); delayMicroseconds(10); digitalWrite(TRIG1, LOW); distancefront = pulseIn(ECHO1, HIGH) / 58.2; }</pre>
20	엔코더 값을 PD제어에 용이하게 변환하고 PD제어 알고리즘을 선언한다.	<pre>void encoder_convert() { timeCurr = millis(); if (timeCurr - timePrev > 1000) { timePrev = timeCurr; noInterrupts(); // 엔코더의 회전 수와 회전 방향 기록 for (int i = 0; i < 4; i++) { if (direct[i] == 1) { // forward 방향의 경우 마우트 방향 표시 값 } else if (direct[i] == -1) { encoderA[i] = -encoderA[i]; // backward 방향의 경우 마우트 방향 표시 } else { encoderA[i] = 0; // 정지한 경우 마우트 방향 표시 } } interrupts(); } sd_control_encoder(); calcul(); // 센서값 저장과 표시 }</pre>

21-1	<p>변환된 엔코더 값을 사용하여 PD제어 알고리즘에 적용한다.</p>	<pre>void pd_control_encoder() { float error[4]; float derivative[4]; float correction[4]; int target[4]; // 각 모터별 목표치 }</pre>
21-2	<p>PD제어 알고리즘을 사용하여 각 모터의 속도를 조절한다.</p>	<pre>// 각 모터별 목표치 설정 for (int i = 0; i < 4; i++) { if (encoderA[i] > 0) { target[i] = 827; // 0도일 경우 목표치 설정 } else if (encoderA[i] < 0) { target[i] = -827; // 360도일 경우 목표치 설정 } else { target[i] = 0; // 90도일 경우 목표치 설정 } } // 각 모터별 초기 속도 설정 for (int i = 0; i < 4; i++) { // 초기 속도 derivative[i] = error[i] - preerror[i]; // 보정 속도 (PD 제어) correction[i] = kp * error[i] + kd * derivative[i]; // 이전 속도 값 저장 preerror[i] = error[i]; // 각 모터별 속도 보정 encoderA[i] += correction[i]; }</pre>
22	<p>엔코더 값을 이용하여 모빌리티의 속도를 계산하고, 모빌리티의 위치를 업데이트하는 함수를 정의한다.</p>	<pre>// 속도 계산 함수 void calcul() { float w_rad[4]; for (int i = 0; i < 4; i++) { w_rad[i] = encoderA[i] / 14.63; } for (int i = 0; i < 4; i++) { speeds[i] = abs(w_rad[i]) * 2.19; } for (int i = 0; i < 4; i++) { encoderA[i] = 0; } float Vx = 0.076 / 4 * (w_rad[0] - w_rad[1] - w_rad[2] + w_rad[3]); float Vy = 0.076 / 4 * (w_rad[0] + w_rad[1] + w_rad[2] + w_rad[3]); if (move_default == 0) { x += Vx * 1.2; } else if (move_default == 1) { x -= Vx * 1.2; } }</pre>
23	<p>최종적으로 모빌리티의 동작을 제어하는 함수 및 메인 루프를 정의한다.</p>	<pre>void sd() { stop(); delay(100); } void loop() { nh.spinOnce(); delay(1); }</pre>

표. 20 모터 구동 아두이노 코드

8.2.2. 코드 부가 설명

<과정2>는 엔코더 모터의 핀을 설정하여 PD제어를 위한 엔코더 값을 받아오기 위해 설정하였다. 앞바퀴의 왼쪽, 오른쪽, 뒷바퀴의 왼쪽, 오른쪽 모터가 각각 1, 2, 3, 4이고 그에 해당하는 핀 번호를 20번, 21번, 2번, 3번으로 설정하였다.

<과정3>에서는 초음파 센서 2개에 대한 핀설정을 하였다. TRIG핀과 ECHO핀은 각각 통신에 필요한 TX, RX핀에 연결해야 하므로 1번 2번 초음파 센서를 각각 18, 19, 16, 17번 핀으로 설정하였다.

<과정 4>에서 PD 제어를 위해 비례 이득과 미분 이득을 설정하고 오차값을 저장하는 prevError라는 배열을 만들었다.

<과정5>에서 모터 구동을 위해 엔코더 모터와 모터 드라이버를 연결한 뒤 각각 아두이노 메가에서 구동을 할 수 있게 해주는 PWM핀인 5, 6번, 7, 8번, 9, 10번, 11, 12번 핀에 각각 연결하였다.

<과정6>에서, 사용 중인 IG-42GM 모터가 나누어 놓은 -255~255에 해당하는 속도를 34로 설정하였다. 이 때, 해당 속도를 34로 설정한 이유는 구동의 편리함과 이용자가 시각장애인인 점을 고려하였다.

<과정7>에서 모터에 결합되어 있는 엔코더에서 나오는 펄스값은 일정하지 않다. 따라서 엔코더 값을 시리얼 모니터에 뜨게 하는 주기를 delay(1100) 즉, 1.1초로 설정하였고, PD제어를 위해 누적되는 엔코더 펄스 값이 아닌 매번 초기화되는 펄스 값을 받기 위해서 초기화하는 과정 추가하였다.

<과정8>에서 초음파 센서에 측정된 거리 값을 저장하기 위한 변수들을 선언하는데 초음파 센서에 측정도니 거리 값은 고정된 값이 아니라 실시간으로 변화하는 값이다. 따라서 'volatile' 키워드로 변수를 선언하였다. 'volatile' 키워드는 변수가 프로그램의 여러 부분에서 변경될 수 있음을 나타내는 키워드이다.

<과정10>은 콜백함수를 정의하고 콜백함수로 하여금 입력 메시지를 받아 주어진 조건에 따라 모빌리티를 동작시킨다. 연속된 총 3개의 if절을 종합하여 설명하면 다음과 같다. 앞쪽의 거리가 0보다 크고 30보다 작거나 같으면, 앞쪽에 장애물이 있음을 감지한 것으로 간주하고 'sd()'함수를 호출하여 정지하게 된다. 또한, 오른쪽 방향에 장애물이 없는 경우 오른쪽 벽에 붙어 갈 수 있게 우직진 방향, 우측 평행 직진, 직진 방향을 순서대로 반복하게 나타냄으로써 주변환경에 방해를 최소화하기로 하였다.

<과정11>은 마찬가지로 콜백함수를 정의하고 콜백함수로 하여금 입력 메시지를 받아 주어진 조건에 따라 모빌리티를 동작시킨다. 만약 모빌리티가 원하는 코너에 도착하게 된다면 시각장애인이 물건을 탐지할 시간을 줘야한다. 따라서 먼저 정지시킨 후, 시각장애인이 물건을 고를 시간을 준다. 그 시간은 이용자가 설정할 수 있다. 만약 물건을 다 고르면 rotate2():를 실행시켜 계산할 카운터로 갈 준비를 한다.

<과정12>도 콜백함수를 정의하고 콜백함수로 하여금 입력 메시지를 받아 조건에 따라 모빌리티를 동작시킨다. 이 과정은 <과정10>과 유사하다. <과정11>에서 물건을 고르고 카운터로 가는 과정이므로 <과정10>의 역이라고 보면 된다.

<과정14>는 초음파 센서에 연결된 핀들과 관련된 모드를 설정한다. `pinMode(TRIG, OUTPUT);`, `pinMode(ECHO, INPUT);`, `pinMode(TRIG1, OUTPUT);`, `pinMode(ECHO1, INPUT);` 부분에서 TRIG 핀은 출력으로 ECHO핀은 입력으로 설정된다. 또한, 'attachInterrupt'를 사용하여 엔코더 입력에 대한 인터럽트를 설정하였다. 각 엔코더 핀에서 하강하는 엣지가 감지될 때마다 해당하는 인터럽트 서비스 루틴(ISR)을 호출한다.

<과정16>은 코드를 최대한 간단화하기 위하여 'for'를 사용하여 속도 설정 루프를 구성하였다. `motors[i].setSpeed(speeds[i] * direct[i]);` 는 각 모터의 속도를 설정한다. 바퀴 특성상 본 프로젝트의 모빌리티는 매카넘휠을 사용하기 때문에 모터의 각속도는 값 1개로 고정시켜도 무방하다.

<과정17~18>은 모터제어를 위한 함수를 설정하는 것이다. 불필요한 방향의 함수는 따로 설정하지 않았다. 모빌리티에 설정한 함수는 직진, 우측 평행 이동, 좌측 평행 이동, 우방향 대각선 직진, 좌방향 대각선 직진, 반시계 방향 회전, 정지 함수를 설정하였다. <과정16> 부가 설명에서 모터의 각속도는 값 1개로 고정시킨다고 언급한 바 있다. 따라서 함수의 속도 설정을 보면 0, -1, 1의 값으로만 제어하는 것을 확인할 수 있다.

<과정19>에서 `distanceright = pulseIn(ECHO, HIGH) / 58.2;` 부분에서 58.2로 나누는 부분은 초음파의 속도와 시간 간의 관계를 이용하여 거리를 계산하기 위함이다. 일반적으로 초음파는 공기나 다른 매질을 통해 전파되는데 공기에서 초음파의 속도는 약 343m/s 이다. 초음파가 발신되어 대상물에 반사되고 다시 수신되는데 걸리는 시간은 초음파가 발신되고 되돌아오는 시간이다. 이 시간을 측정하여 거리를 계산하고 계산하는 과정에서 초음파 속도로 나누어야 하는데 실제로는 속도가 다르게 측정될 수 있기 때문에 조정값으로 58.2를 사용한다.

<과정20~22>에서 각 모터의 엔코더의 펄스 목표치를 전진할 때 827, 후진할 때 -827로 설정하였다. 이는 1회전당 펄스값을 모터의 제원에서 얻고 이를 토대로 모터 각속도에 대입했을 때 실제로 계산한 이상적인 값이다. `encoderA[i] / 14.63`에서 '14.63'은 엔코더의 카운트 값을 각속도로 변환하기 위한 상수이다. 이는 엔코더 해상도를 고려한 값이다. `speeds[i] = abs(w_rad[i] * 2.19)`에서 '2.19'는 각속도를 선형속도로 변환하기 위한 상수이다. 이는 매카넘휠의 반지름과 모터의 감속비를 고려한 값이다.

Ⅲ. 결론

9. 제품 요약

9.1 제품 사양

제품명	(Useful Market Mobility)[UMM]
크기	$478.1[mm] \times 500[mm] \times 545[mm]$
무게	$25kg$
구성	
프레임	30×30 알루미늄 프로파일
구동부	엔코더 장착형 감속 기어 모터: IG42-GM+ Encoder 01 type 메카넘 휠: NEX-14165 리튬이온배터리: 12V 10AH 배터리팩 (그린에너지 GR50E-0302)
조작부	아두이노 우노, 아두이노 메가, 2채널 모터 드라이버 (MDD-3A), Jetson Nano
센서	2D Lidar-X2 센서, RFID 센서: RC-522, 초음파 센서: HC-SR04, 블루투스 센서: HC-06

표. 21 최종 제품 사양서

9.2 제품 성능

이동 속도	$0.56[m/s](= 2[km/h])$
구동 동력	$40[W]-12[V]$
소비 전력	$360[W]$
가동 시간	$2.25[hour]$
최대 소비 에너지	$360.30[Wh]$
최대 이동 거리	$7875[m]$
예상 가동 장소	마트 내부

표. 22 제품 성능표

9.3 제품 기능

분류	개요	상세 설명
주기능	PD제어, 메카닉 휠을 이용한 전방향 모빌리티 구동	엔코더 모터 4개를 이용해 펄스 값에 따른 PD제어를 할 수 있도록 하였으며, 바퀴는 메카닉휠을 채택하여 제자리 회전을 포함한 전방향 주행이 가능하도록 하였다.
	마트의 맵 만들기	2D Lidar를 이용, 마트의 맵을 나타낼 수 있도록 하였다.
	우측 통행	우방의 초음파센서를 이용하여 오른쪽 벽과 일정한 거리를 유지하며 주행하도록 하였다. 이를 통해 우측 통행이 일상화된 사람들과 최대한 부딪히지 않도록 한다.
	마트 물건 인식	제품의 정보를 가지고 있는 RFID 카드와 센서를 통해 원하는 물건 위치에 도달할 시 해당 물건의 정보를 음성으로 알리도록 하였다.
	장애물 인식	전방의 초음파 센서를 이용해 전방에 물체가 있을 시 모빌리티의 구동을 멈추며, 장애물이 사라지면 다시 구동하도록 하였다.
부가 기능	앱과 연동 및 통신	누구나 가지고있으며 음성인식 기능을 기본적으로 제공하는 스마트폰을 모빌리티와 통신하여 제어할 수 있도록 하였다.
	음성을 통한 앱 제어	시각장애인을 위한 기능이므로, 음성으로 제어하도록 하여 제어에 불편함이 없도록 하였다.
	친환경 모빌리티	리튬이온배터리를 사용하여 모빌리티를 구동시켜, 공해 및 소음이 최대한 줄도록 하는 친환경 모빌리티가 되도록 하였다.

표. 23 제품 기능표

위의 표를 참고하면, 모빌리티의 의 부가 기능 중 ‘휴대폰 애플리케이션 연동’은 주기능이라고 할 수 있을만큼 중요하다. 시장 조사를 참고하면 시각장애인이 장을 볼 때 또는 집에서 혼자 시장앱을 이용하여 시장을 볼 때 느끼는 불편한 점은 음성인식이나 음성을 이용한 서비스 제공이 부족하다는 점이다. 이를 보완하고자 애플리케이션을 통하여 음성 서비스를 제공하기로 하였다.

‘UMM’은 시각장애인들의 편리한 장보기를 위하여 설계된 모빌리티이지만, 마트에 대해서 정보가 없는 노약자층이나, 기타 사회적 약자가 써도 편리할 것이다.

9.4 단계별 달성 목표

1. 사용자의 의사를 3초 이내로 받아 들인다.
2. 기계가 받아들인 정보를 사용자에게 전달한다.
3. 사용자를 원하는 위치까지 데려다 준다.
4. 30cm이내의 물건 중 사용자가 원하는 물건을 판별한다.
5. 주행 간에 5m 이내의 장애물을 인식, 회피 할 수 있다.

달성 목표	평가 방법
1. 사용자의 의사를 모빌리티가 받아 들인다.	1. 스마트폰에서 앱을 실행한다
	2. 사용자가 음성을 입력한다
	3. 모빌리티가 작동을 시작한다
2. 모빌리티가 받아들인 정보를 사용자에게 전달한다.	1. 모빌리티에 RFID를 인식시킨다
	2. 사용자의 휴대폰에 물품 정보가 전송된다
	3. 전송된 정보가 음성신호로 사용자에게 전달된다
3. 사용자를 원하는 위치까지 데려다 준다.	1. SLAM을 이용한 Map을 완성한다
	2. 사용자의 음성 신호를 받아들인다
	3. 사용자가 사고자 하는 물품의 위치로 데려다 준다
4. 앱에서의 기능이 모두 작동한다.	1. 스마트폰에서 앱을 실행한다
	2. 앱의 각각의 버튼을 조건에 맞게 실행한다.
	3. 각각의 조건에 맞게 음성출력 혹은 모빌리티의 이동이 가능한지 판단한다.
5. 주행 간에 우측 보행이 가능하다.	1. 모빌리티를 우측 벽이 있는 환경에 둔다. 이때 우측 벽과 임의로 설정하여 둔다
	2. 스마트폰에서 앱을 실행하여 모빌리티를 구동시킨다
	3. 모빌리티가 우측벽과 멀다면 우측벽을 향해 이동할 것이고, 가깝다면 좌측으로 이동할 것이며, 적당한 거리라면 전방주행할 것이다. 각각의 경우에 맞는지 판단한다.

표. 24 달성 목표 작성표

9.5 목표 달성 여부

본 제품의 목표는 다음과 같다.

1. 사용자의 의사를 모빌리티가 받아 들인다.
2. 모빌리티가 받아들인 정보를 사용자에게 전달한다.
3. 사용자를 원하는 위치까지 데려다 준다.
4. 앱에서의 기능이 모두 작동한다.
5. 주행 간에 우측 보행이 가능하다.

목표를 달성했는지 여부를 판단하기 위해 주행영상을 통해 확인하고자 한다.



Fig. 78 [목표 1] 달성과정

다음과 같이 음성인식 버튼을 누르고 '과일 코너'라 말한 후 텍스트로 인식되는 것을 알 수 있다. 과일 코너와 비슷한 단어인 발코니라고 인식 되었을 경우 오른쪽의 '보내기 버튼'을 눌렀을 때 다시 '음성 인식 버튼'을 누르고 말해달라고 안내함을 알 수 있으며, '전송! 버튼'을 누르면 과일 코너로 이동한다. 따라서 사용자의 이동 의사를 모빌리티가 받아들이고 그에 따른 행동을 할 수 있다.

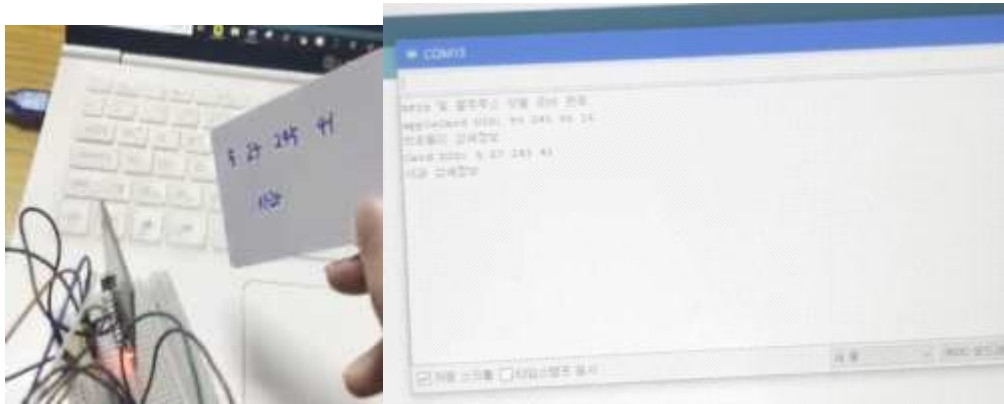
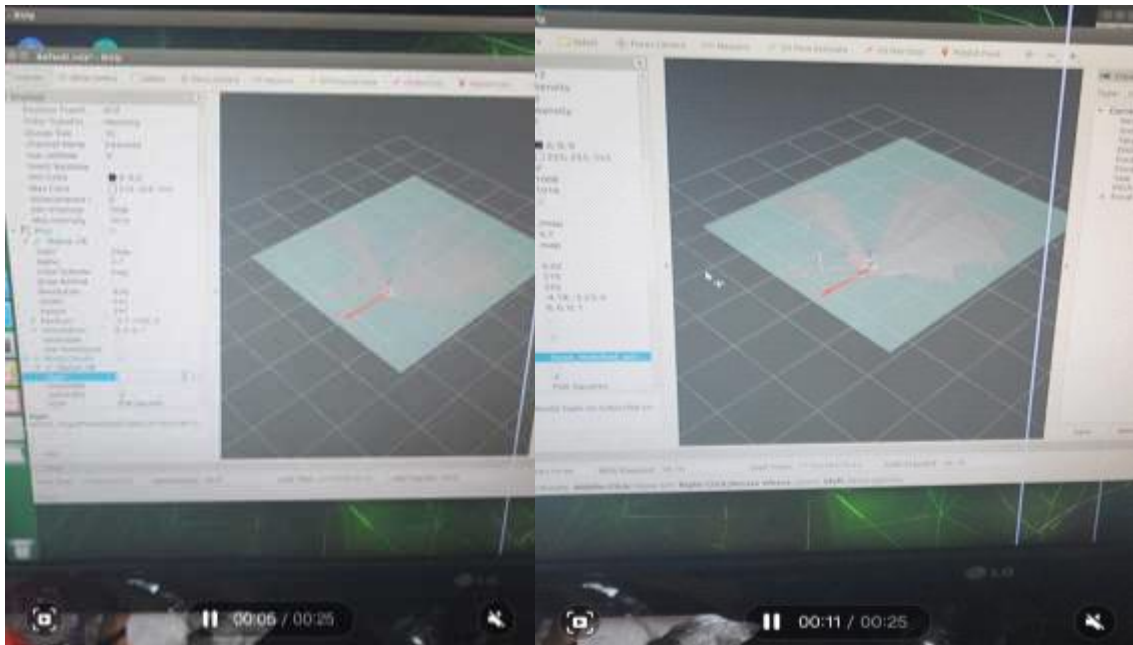
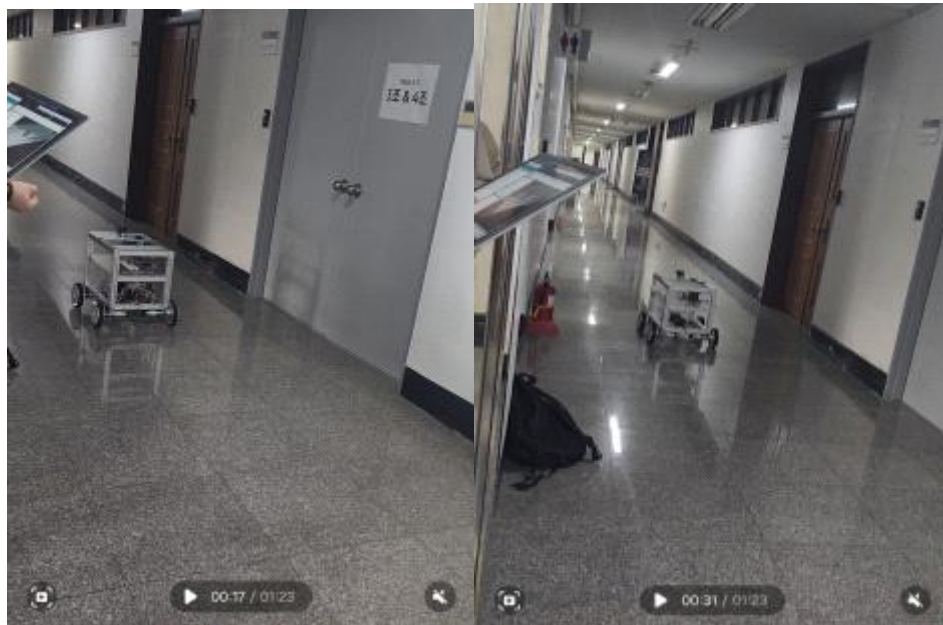
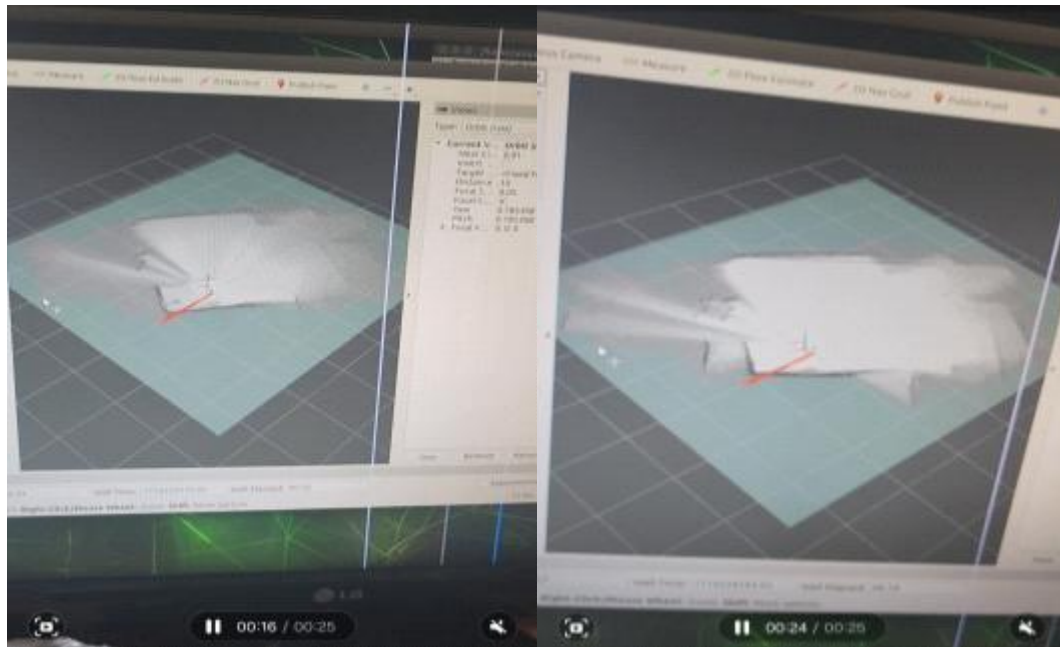


Fig. 79 [목표 2] 달성과정

UID가 99 245 88 16인 카드를 RFID에 인식시켰을 때 브로콜리의 상세정보를 아두이노 시리얼 모니터에 출력하고, 브로콜리의 상세정보를 앱에서 읽어준다. 마찬가지로 UID가 5 27 245 41인 카드를 RFID에 인식시켰을 때 사과와 상세정보를 아두이노 시리얼 모니터에 출력하고, 사과의 상세정보를 앱에서 읽어준다. UID는 고유값이므로 각각에 물품의 상세정보를 넣어주게 된다면 어떠한 물품의 정보이든 넣을 수 있어, 사과 대신 토마토를, 브로콜리 대신 바나나를 넣어도 동일하게 작동함을 알 수 있다. 사용자는 따라서 모빌리티가 RFID를 인식한 정보를 앱으로 받을 수 있다.





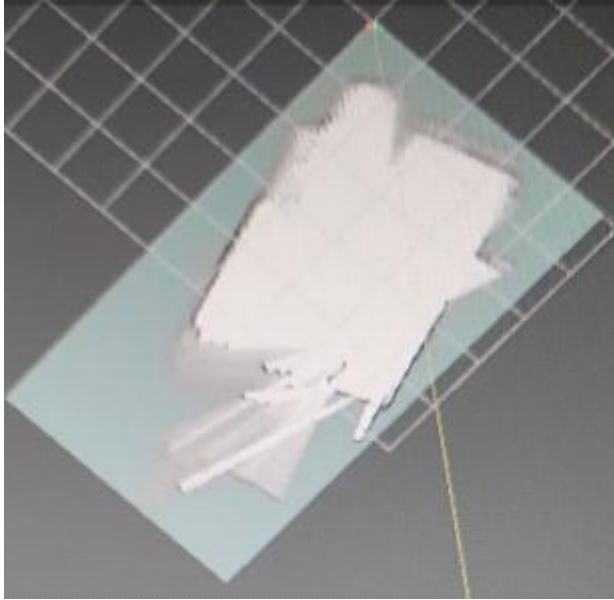


Fig. 80 [목표 3] 달성과정

rviz를 통해 맵을 2D로 표현할 수 있는 공간을 열고, SLAM을 실행하면 2D Lidar를 통해 맵이 실시간으로 만들어 지는 것을 알 수 있다. 이는 시간이 경과함에 따라 맵이 점점 선명하게 나타남을 통해 알 수 있다. 또한 음성신호를 보내고 전송을 누르면 주행을 시작하며, 노트북 모니터 화면에 맵을 실시간으로 생성하였음을 알 수 있다. 마트의 맵은 한번 작성하면 마트 내에서 변하는 것이 없기 때문에 반영구적 사용이 가능하다. 위와 같은 맵을 만들 수 있었는데, 위에서 흐리게 표시된 것은 유리창 쪽으로, 2D Lidar의 특성상 펄스 레이저를 사방으로 발사할 때 유리창을 투과하기 때문에 흐리게 나옴을 알 수 있다. 따라서 맵의 정보를 제공받을 수 있으며, '전송! 버튼'을 클릭할 시 우측보행하며 과일코너까지 이동할 수 있다. 이후 카운터로 다시 돌아오는 과정 또한 스스로 가능하다.



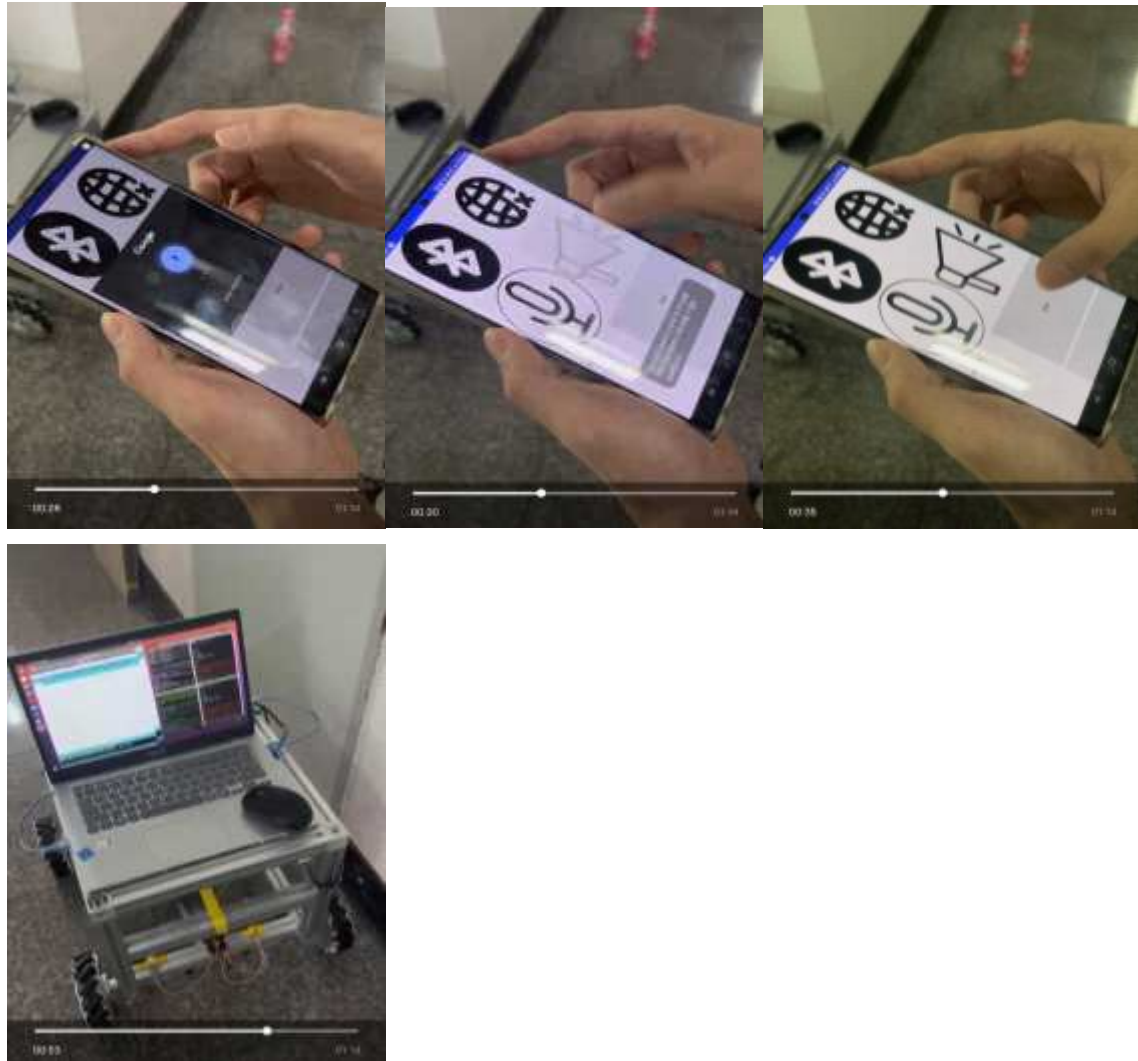


Fig. 81 [목표 4] 달성과정

구동하는데 필요한 버튼을 순서대로 누르며 음성 입력과 출력, 그에 따른 알림을 음성을 통해 알리도록 하고, 모빌리티가 원하는 곳으로 주행하게 함을 알 수 있다. 따라서 각각의 버튼은 모두 모빌리티를 주행시키기 위해 필요함을 알 수 있으며, 앱의 기능은 모두 정상 작동함을 알 수 있다.

```

// 첫번째 과정 (벽면 이동)
void callbackFunction1(const std_msgs::Int32 inputMessage) {
  if (inputMessage.data == 1) {
    scanRight();
    scanFront();
    // 앞쪽 거리 조건을 우선적으로 처리
    if (distancefront > 0 && distancefront <= 30) {
      ad();
    } else {
      // 오른쪽 거리 조건을 처리
      if (distanceright > 0) {
        if (distanceright <= 10) {
          crossel();
        } else if (distanceright >= 25) {
          right();
        } else if (distanceright >= 0 && distanceright < 25) {
          straight();
        }
      }
    }
  }
}

```



Fig. 82 [목표 5] 달성과정

아두이노 메가 코드에서 오른쪽 벽과 일정한 거리를 유지하며 직선주행하도록 하였다. 이를 확인하기 위해 앱을 통해 모빌리티를 구동시켰을 때, 처음에 우측 벽과 멀다 판단하여 41초의 두 그림에서 우측으로 이동하는 것을 알 수 있다. 이후 적당한 거리를 가졌다 판단하여 직선 주행을 한 모습이다. 우측 벽에 가까이 붙여놓았을 때에는 적당한 거리를 가지기 위해 좌측으로 이동하므로, 이를 통해 우측보행이 잘 이루어지고 있음을 알 수 있다.

10. 경제성 평가

10.1 제작비용

번호	부품명	모델명	수량	비용
H/W 부품				
1	배터리	그린에너지 3S2P 10.8V 10Ah 리튬이온 21700 12V 배터리팩	2EA	140,800
2	바퀴	메카넘휠	4EA	193,600
3	브라켓	다이캐스팅 브라켓 3025	9EA	4,000
4	브라켓	60mm 60각 BRACKET	4EA	49,440
5	프레임	프로파일 DF3030(30*30*80)	2EA	85,310
6	프레임	프로파일 DF3030(30*30*300)	1EA	
7	프레임	프로파일 DF3030(30*30*380)	2EA	
8	프레임	프로파일 DF3030(30*30*440)	2EA	
9	프레임	프로파일 DF3030(30*30*500)	2EA	
10	볼트	조인트 브라켓 30계열	22EA	31,900
11	허브	NEX-18008 8mm hub	2EA	70,400
12	모터	IG-42GM+Encoder 01 TYPE	4EA	482,880
13	블록	코너 블럭 3030용	4EA	27,200
S/W 부품				
1	컨버터	아두이노 DC-DC 가변형 스텝다운(전압강하) 모듈	1EA	6,600
2	CPU	아두이노 메가 2560 R3 호환 보드 CH340 [SZH-EK375]	1EA	18,000
3	CPU	엔비디아 젯슨 나노 개발 키트	1EA	283,800
4	드라이브	[MDD3A] 3Amp 4V-16V DC Motor Driver (2 Channels)	2EA	18,200
5	센서	X2 거리 측정 LiDAR Sensor	1EA	86,900
6	센서	엔비디아 Jetson Nano AI HD 카메라 모듈 800만화소	2EA	55,000
7	모듈	RFID-RC522	1EA	2,750
8	SD카드	젯슨 나노 micro SD	1EA	19,800
9	어댑터	젯슨나노 어댑터	1EA	14,490
10	모듈	Jetson Nano 18650×6 UPS 모듈	2EA	93,500
11	배터리	보호회로 리튬이온 충전지(배터리) 2000mA	6EA	40,590
12	모듈	아두이노 블루투스 모듈(HC-06)	1EA	5,500
			총 금액	
			1,712,840(원)	

표. 25 제작비용 정리

10.2 제품의 가격경쟁력 재확인

본 제품의 본질적인 목표는 시각장애인의 편한 마트 이용을 위해 편리함을 제공하는 것이다. 하지만 모빌리티특성상 거동이 불편한 분들의 불필요한 동선 낭비등을 위해 쓰일 수도 있다. KOSIS(국가 통계 포털)에 따르면, 우리나라의 등록장애인은 매해 늘어나는 추세이며, 시각장애인 또한 늘어나고 있는 추세라고 한다. 따라서 본 모빌리티의 수요가 늘어날 것으로 예상되며 사용을 원하는 장애인들도 많아질 것이다.

다양한 회사가 시각장애인을 위한 모빌리티를 개발하고 있다. 오스트리아에 본사를 둔 테크이노베이션은 시각장애인의 편리를 위해 장애물 감지 지능형 신발 ‘이노메이크(Innomake)’를 개발하였다. 이노메이크의 경고 시스템은 거리 센서, 발 움직임 감지 센서 등 많은 센서들을 이용하여 시각장애인 거동에 편리함을 주지만 신발가격은 약 3,200유로(한화 약 435만원)이라고 한다. 따라서 상용화 자체가 어렵고 상용화가 된다고 하더라도 사용하고자 하는 시각장애인분들이 경제적 부담을 느낄 것이다. 하지만 UMM은 개인적으로 구매하는 제품이 아니고 마트에서 공용으로 쓰는 모빌리티이다. 제품은 170만원 내외이므로 전에 언급한 신발보다 경제성과 실용성이 좋다고 할 수 있다.



Fig. 83 시각장애인 증가 추세³⁰⁾



Fig. 84 테크이노베이션 사의 ‘Innomake’³¹⁾

30) fig1. <https://wish.welfare.seoul.kr/swflmsfront/board/boardr.do?bmno=10021&opno=10017&bno=93870>

31) fig2. <https://www.donga.com/news/It/article/all/20220623/114071666/1>

11. 결론 및 추후 계획

최종 제품 제작을 완료하고 실제 구동 시, 설계 목표에 따라 사용자가 앱(APP)을 이용하여 원하는 마트 내 코너에 모빌리티가 이동하는 것을 확인하였다.

본 프로젝트에서 제작한 'UMM'을 통해 시각장애인분들이 마트를 이용할 시, 원하는 물건을 구매하고 싶을 때 어디로 가야할 지 몰라 겪게 되는 불편함을 개선할 수 있을 것이다. 더 나아가, 장애물을 피할 때, 매키넴혈을 이용하여 유연한 회피 동작을 구현할 수 있으며, APP과의 연동이 용이하여 이용자가 원할 때마다 원하는 코너를 갈 수 있어 시각장애인의 유연한 장보기를 도와준다.

추후계획은 다음과 같다. 대한기계학회에서 주관하는 '제14회 전국학생설계경진대회'에 참가하기로 하였으며 현재 설계제안서까지 제출 완료하였다. 앞으로 대회 일정에 맞추어 중간보고서와 최종보고서를 작성하여 제출할 예정이다.

The figure shows a design proposal form for the 14th National Student Design Competition. The form is divided into six panels. The top-left panel is the 'Design Proposal' (설계제안서) with fields for title, author, and institution. The top-middle panel is the 'Design Proposal Requirements' (설계제안서 요율본) with a table for requirements. The top-right panel is the 'Design Proposal' (설계제안서) with a table for requirements. The bottom-left panel is the 'Design Proposal' (설계제안서) with a table for requirements. The bottom-middle panel is the 'Design Proposal' (설계제안서) with a table for requirements. The bottom-right panel is the 'Design Proposal' (설계제안서) with a table for requirements.

Fig. 85 ‘전국학생설계경진대회’ 설계제안서

12. CAD 도면

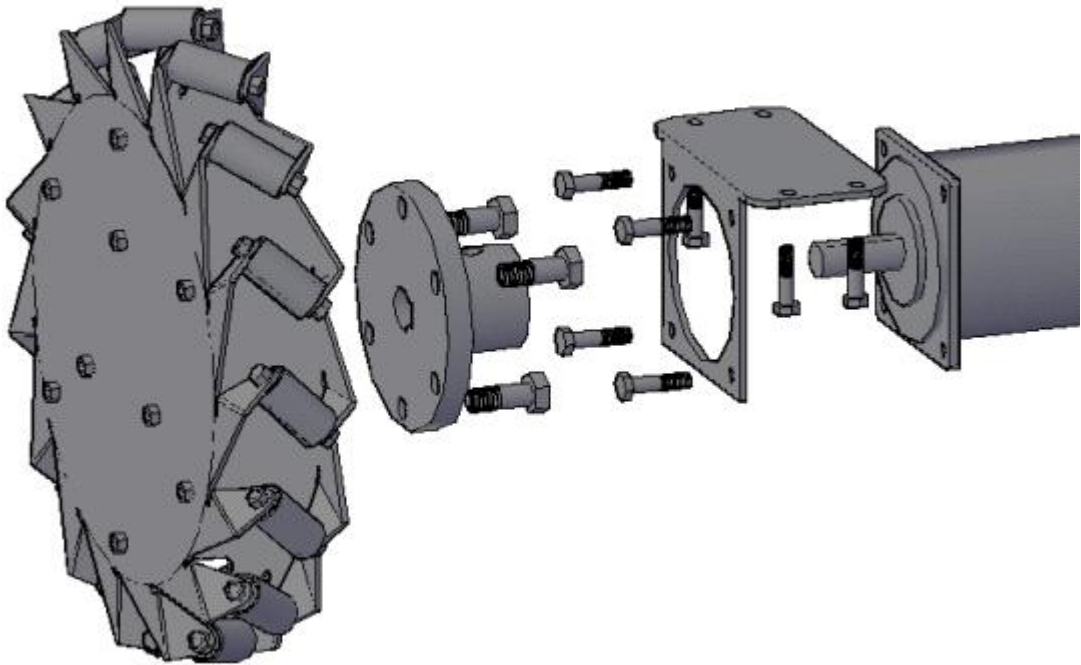


Fig. 86 구동부분 조립도 1

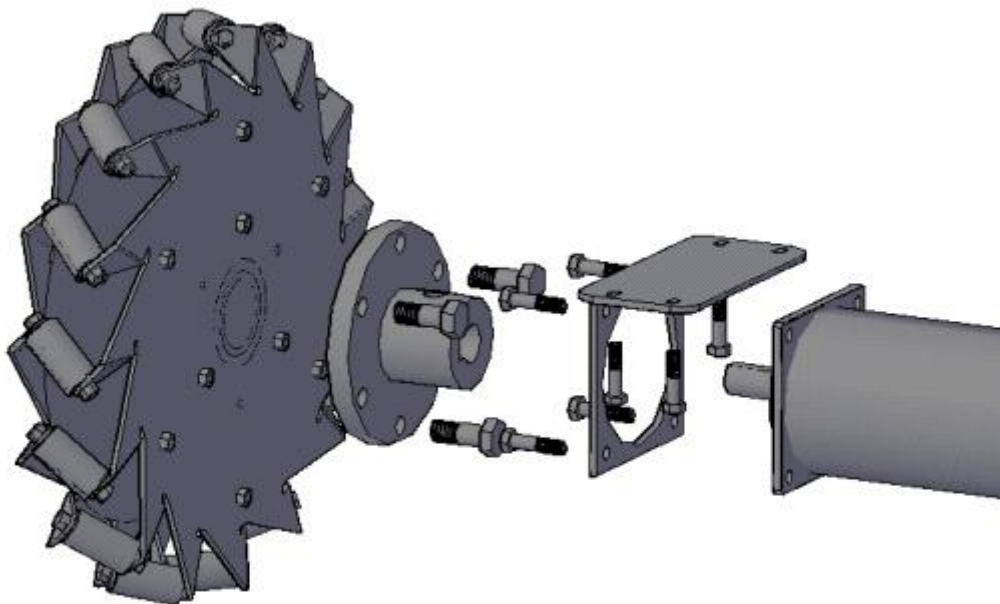


Fig. 87 구동부분 조립도 2

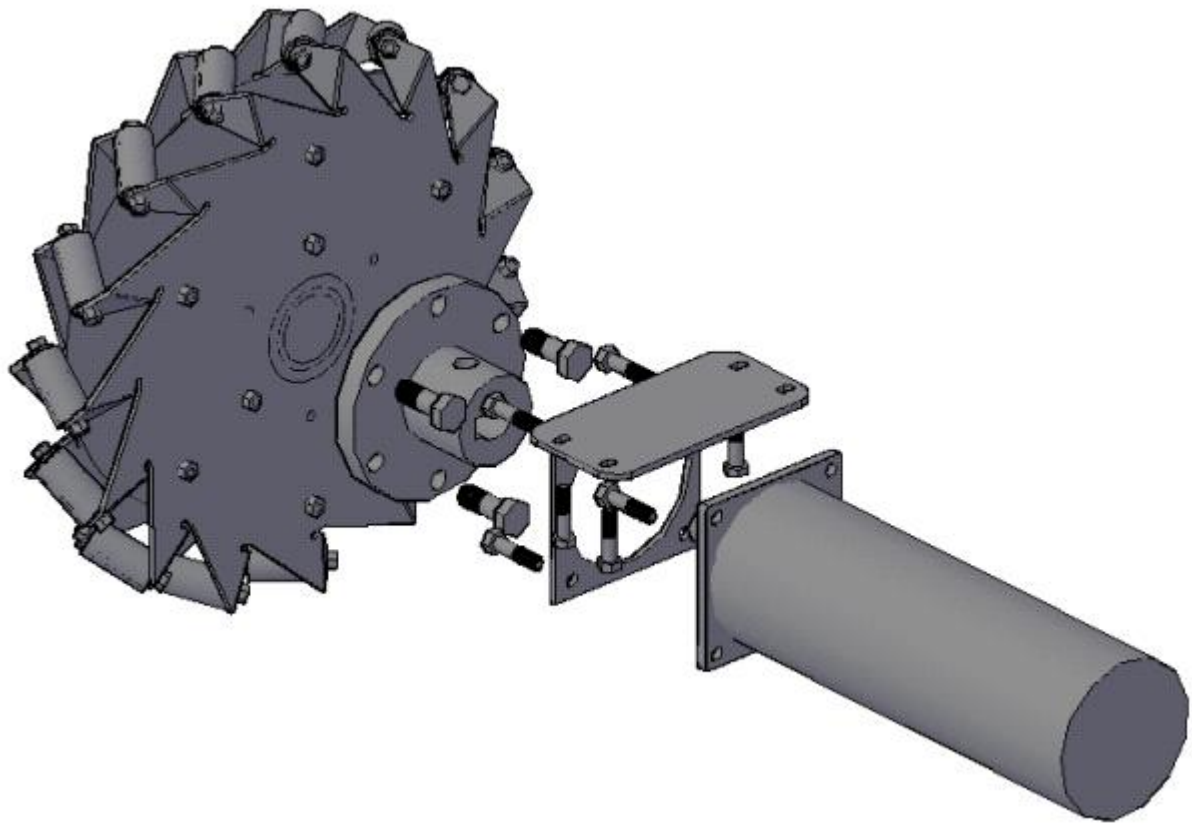


Fig. 88 구동부분 조립도 3

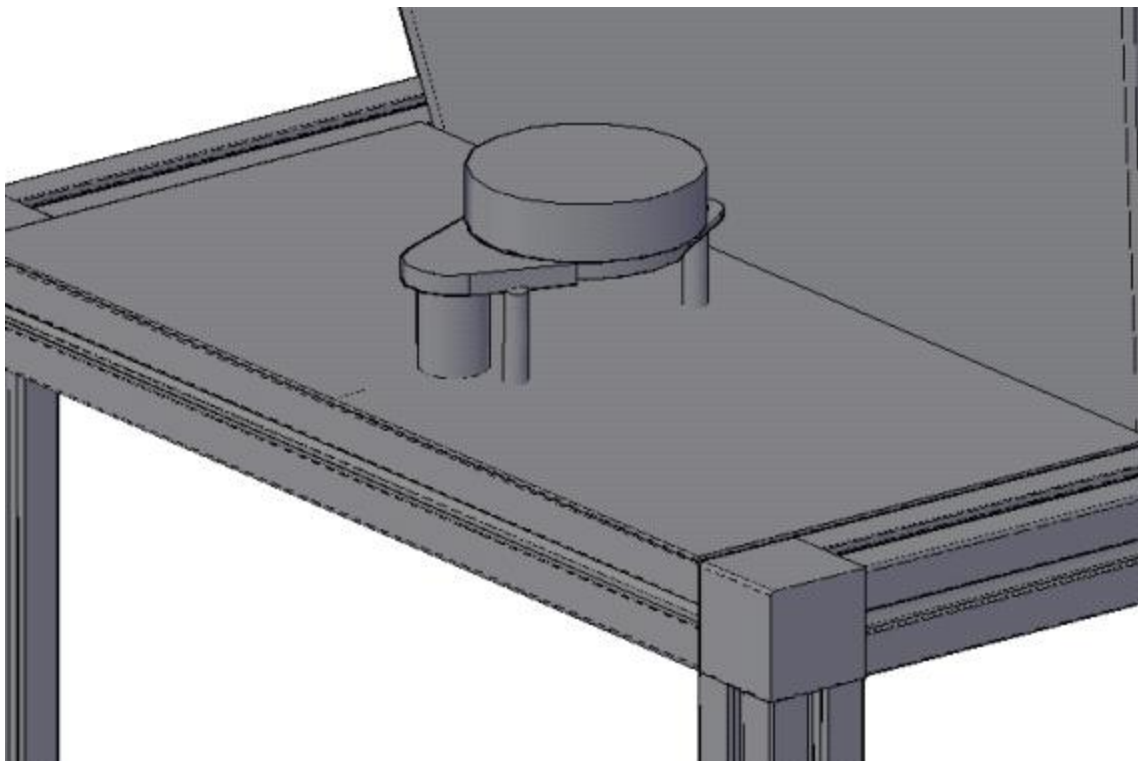


Fig. 89 LIDAR 센서 위치 및 모습

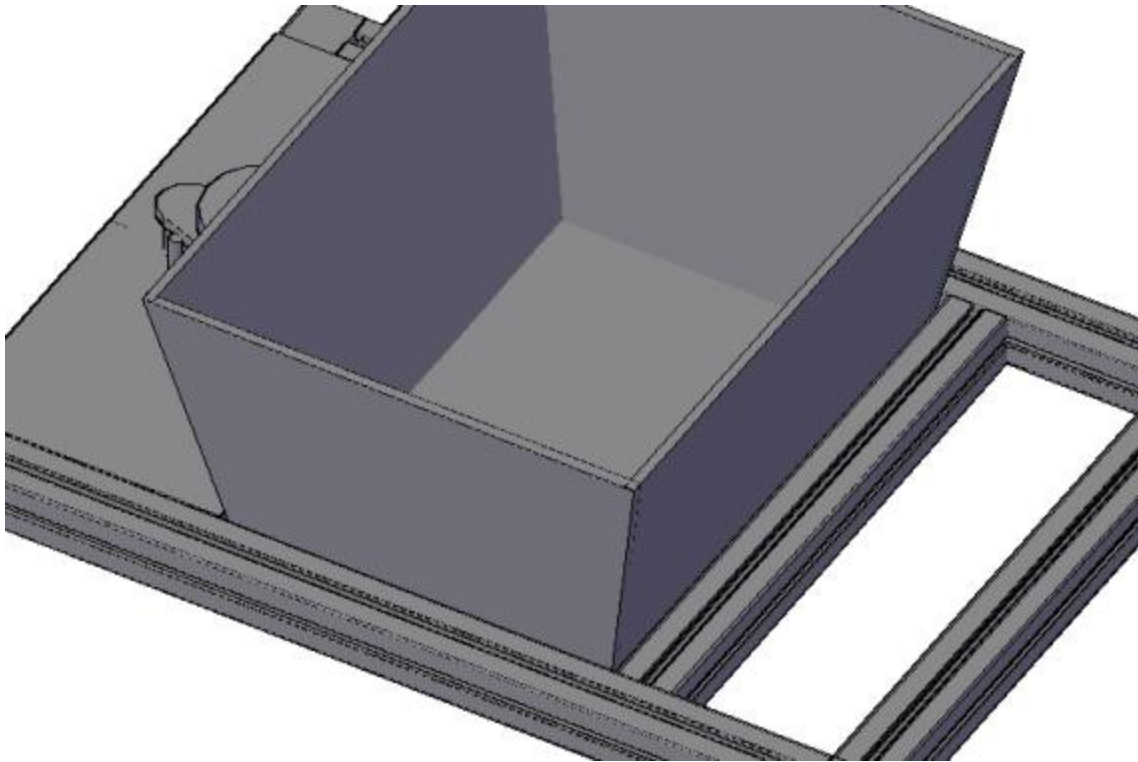


Fig. 90 바구니 장착 시 모습 1

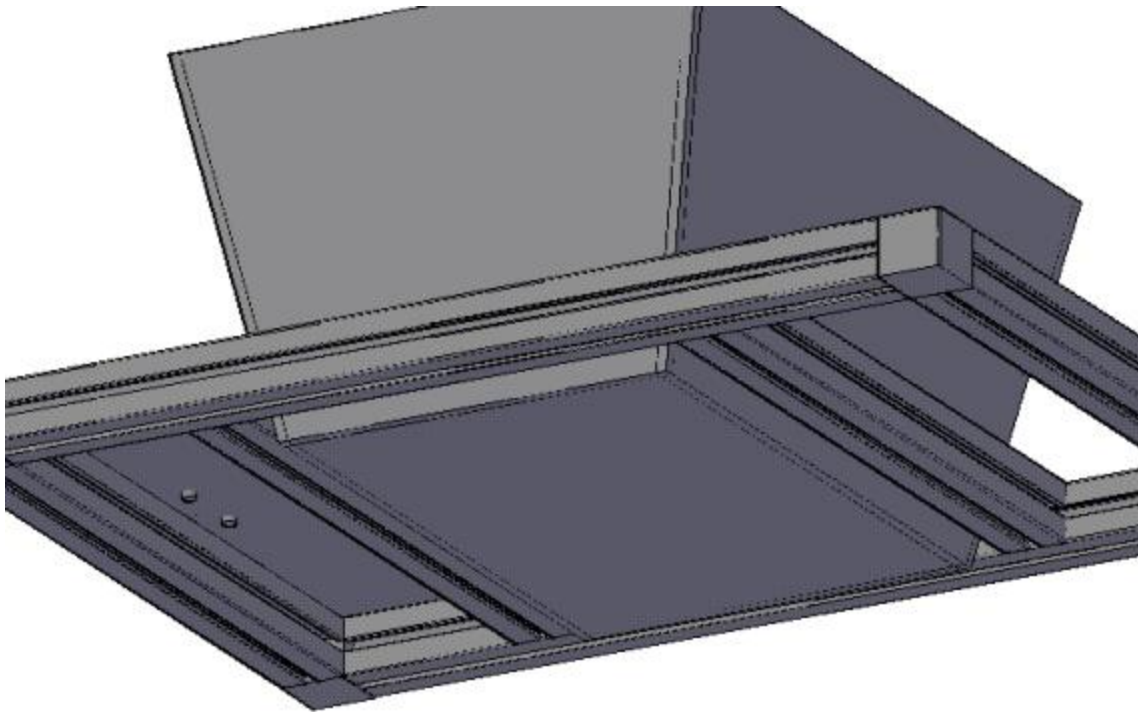


Fig. 91 바구니 장착 시 모습 2

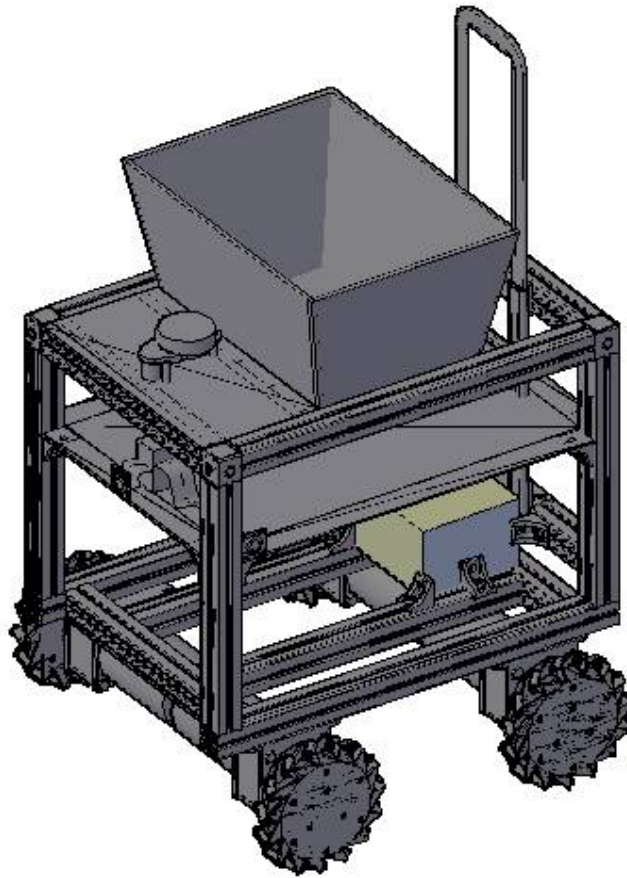


Fig. 92 모빌리티 전체 모습 1

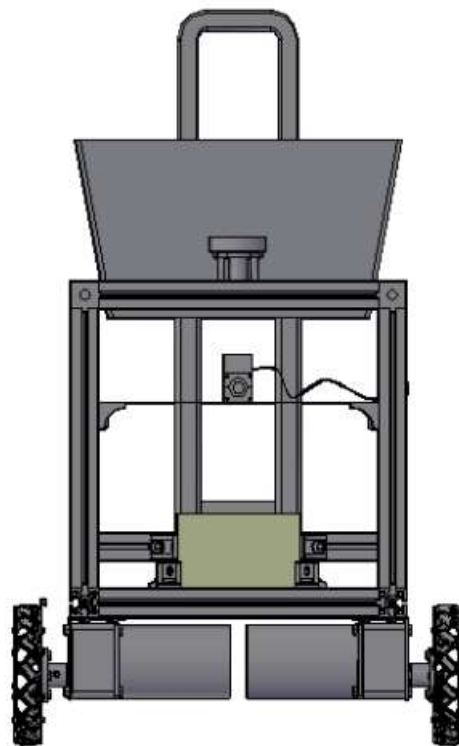


Fig. 93 모빌리티 전체 모습 2: 정면도

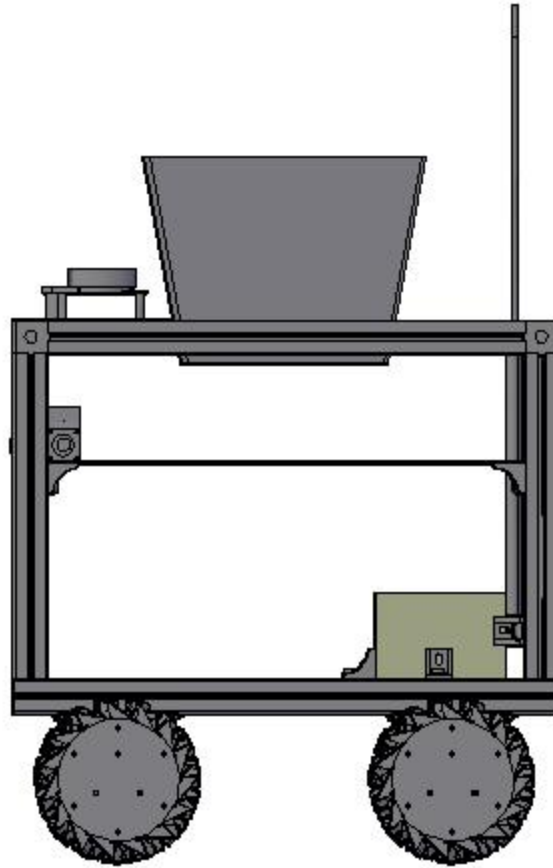


Fig. 94 모빌리티 전체 모습 3: 우측면도

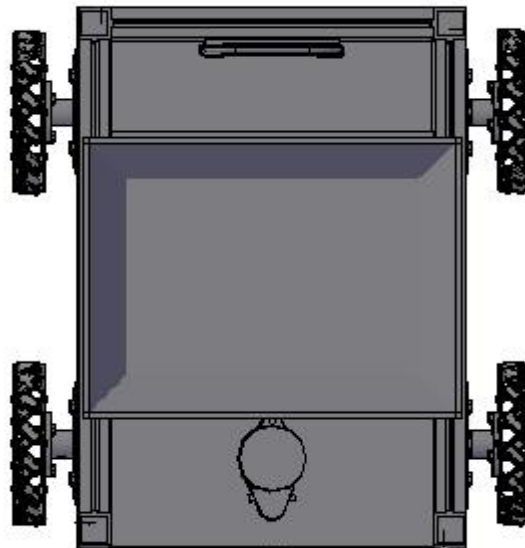


Fig. 95 모빌리티 전체 모습 4: 평면도

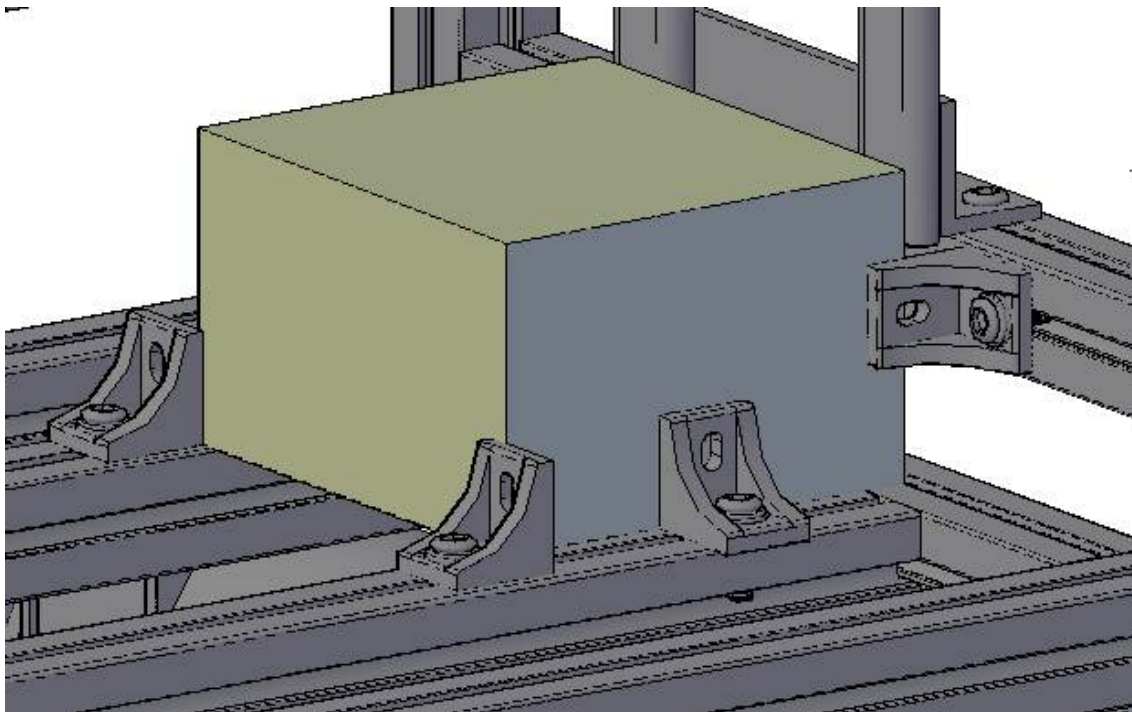


Fig. 96 배터리 체결방식 및 위치 1

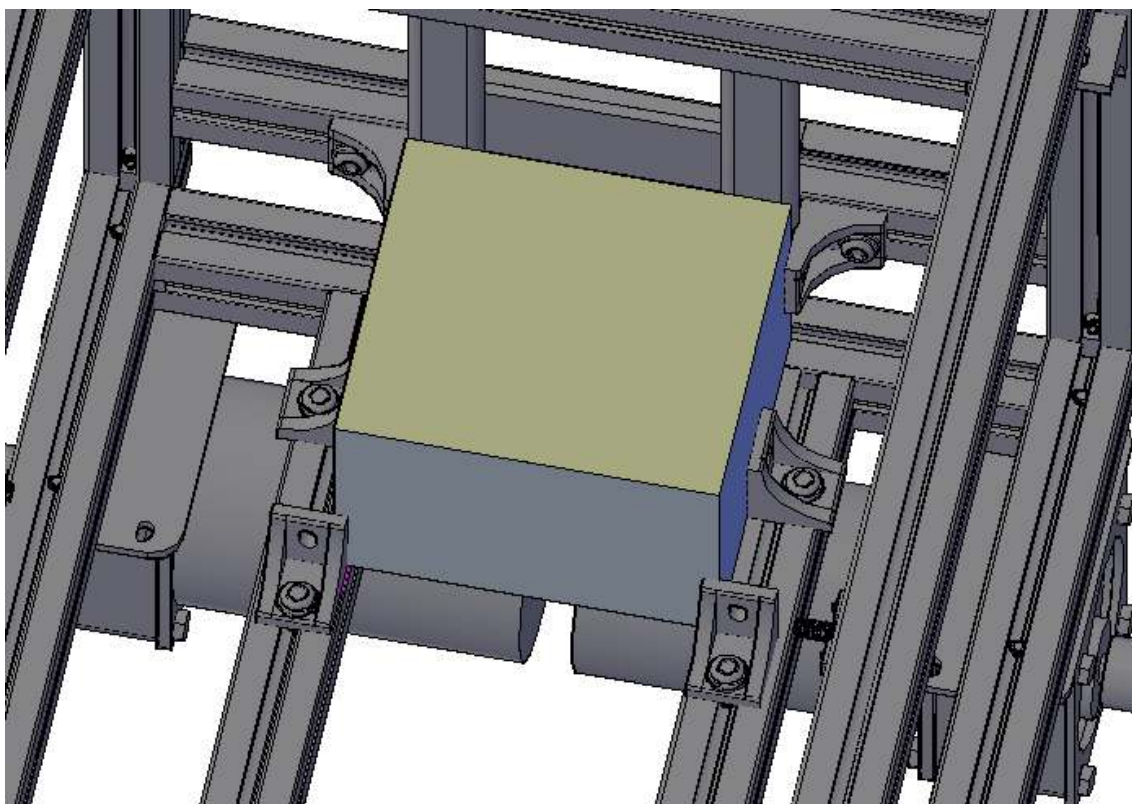


Fig. 97 배터리 체결방식 및 위치 2

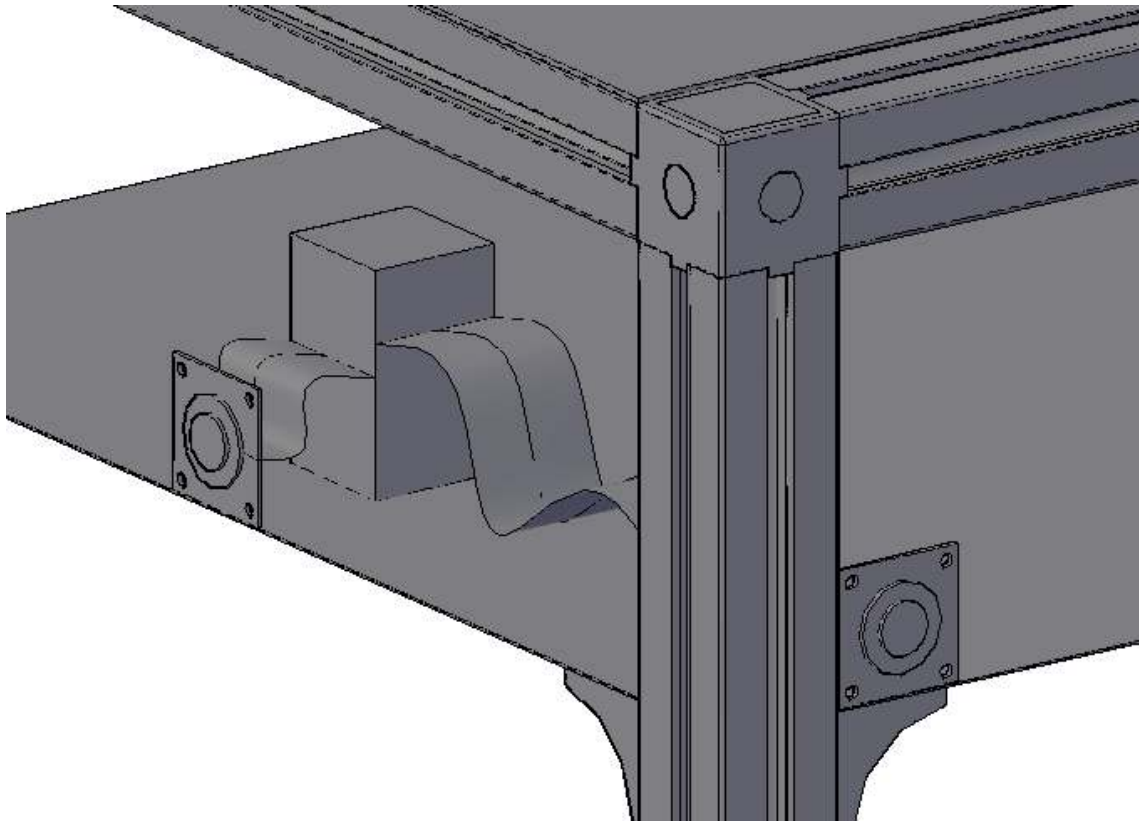


Fig. 98 카메라 모듈 체결 방식 및 위치

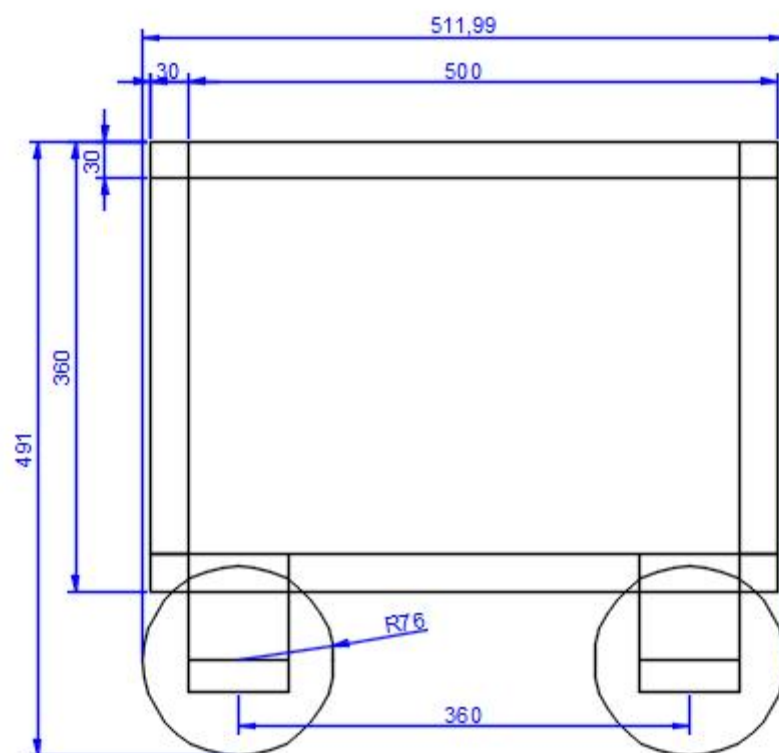


Fig. 99 2D 우측면도

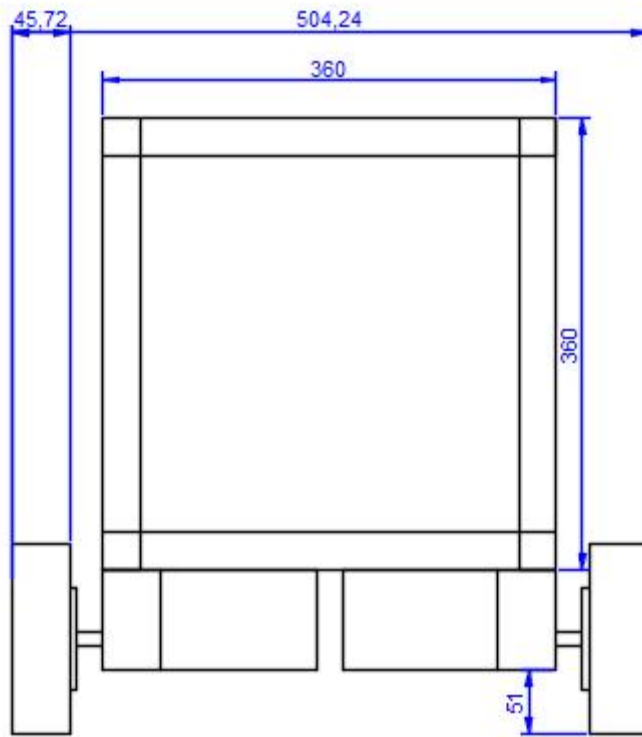


Fig. 100 2D 정면도

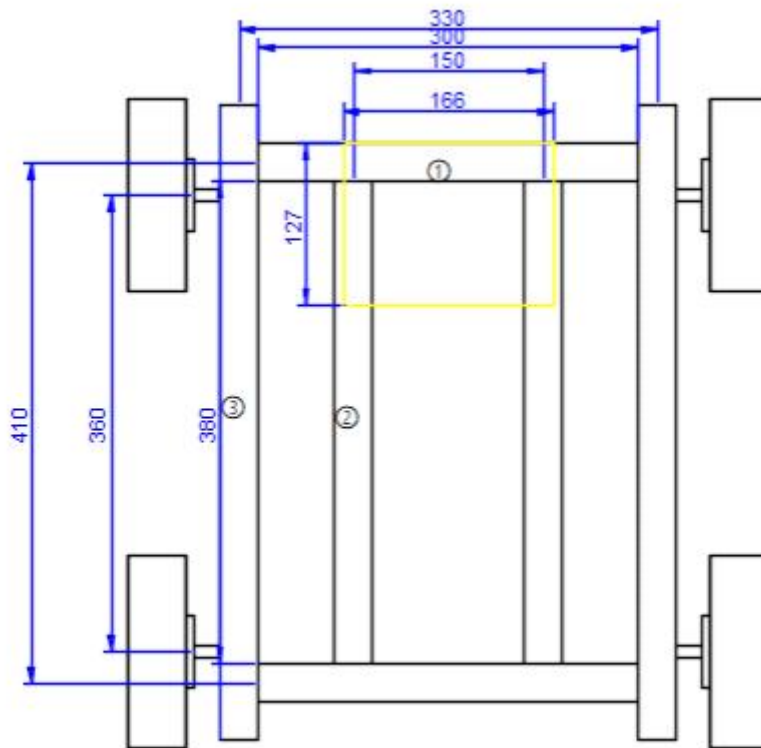


Fig. 101 2D 평면도

13. 회의록

기계공학 종합 설계1 5분반 1조					
회의일자	2024.03.04	장소	보통카페	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병춘, 장한솔				
회의주제	기계공학 종합 설계				

[회의내용 요약]

1. 종합 설계 2 진행 계획 러프 스케치
3월 ~ 4월 초 안으로 하드웨어 설계 끝내고 소프트웨어 관련은 추후 고민
S/W 관련 공부는 하드웨어 준비 전부터 미리미리 해 둘 것
2. 설계 필요 요소 리마인딩



향후계획 / 다음 목표

정했던 큰 주제들을 세분화하여 구체적인 설계 목적 후보군 추리기

기계공학 종합 설계1 5분반 1조					
회의일자	2024.03.11	장소	보통카페	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병준, 장한솔				
회의주제	기계공학 종합 설계				

[회의내용 요약]

1. 기계 공학 종합 설계 1 보고서에서 미진했던 점 있는지 확인 후 대안 생각 해 보기
2. 충북대 자율 주행 캠프 자료 공유 및 토대로 자율 주행 기초 공부
3. 노트북에 ROS 설치 후 가벼운 파이썬 코딩 (Hello World, Turtle Bot 등) 연습
4. GitHub에서 우리가 쓸 수 있을 만한 코드 찾아 보기



향후계획 / 다음 목표

무작정 GitHub를 뒤지지 말고, 구체적으로 필요한 알고리즘 분류 후 나눠서 찾아보기

기계공학 종합 설계1 5분반 1조					
회의일자	2024.03.18	장소	보통카페	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병준, 장한솔				
회의주제	기계공학 종합 설계				

[회의내용 요약]

1. 자율 주행에 필요한 알고리즘 분류

- 구동 : SLAM , Navigation, Object Detecting, Avoid
- 통신 : 휴대폰과 아두이노, 젯슨나노의 통신 / 각 센서의 입력을 받아오는거
- 제어 : 메카닉 휠 제어 (방향전환 메커니즘, PID 제어 등)
- 인식(?) : 장애물 인식, 물체(찾는 물건) 분류, RFID 정보 받아오기

2. 필요한 알고리즘에 대해 세부 분류



향후계획 / 다음 목표

세부 분류한 정보 토대로 공부하기

기계공학 종합 설계1 5분반 1조					
회의일자	2024.03.25	장소	ICT관	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병준, 장한솔				
회의주제	기계공학 종합 설계				

[회의내용 요약]

1. 종합 설계 구매 물품 수령 후 프레임 작업
2. 가공 해야 할 부품 (아크릴판 크기에 맞게 가공) 들 가공 도면 CAD 제작
3. 브라켓 프로파일에 들어갈 정도로 가공하기
4. 가공 의뢰 맡기기
5. 앱 인벤터로 핸드폰에서 아두이노에 신호 보내는 어플 만들기



향후계획 / 다음 목표
 가공 의뢰 보내기
 젯슨 나노 연결해보기

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.04.01	장소	보통카페	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병춘, 장한솔				
회의주제	종합 설계 설계 목적 구체화하기				

[회의내용 요약]

1. 프레임 완성 및, 센서 등 구성요소 위치 맞춰보기
2. 추가로 필요한 체결 물품 등 확인
3. 바퀴 연결 후 모터 드라이버 작동 수동으로 확인
4. 각 바퀴 엔코더 연결 후 엔코더 값 제대로 받아 오는지 확인 엔코더 값 토대로 Odometry 작성 혹은 PID 제어 가능할지 토의



향후계획 / 다음 목표
LiDAR 사용 위주로 계획 진행하기

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.04.08	장소	보통카페	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병춘, 장한솔				
회의주제	종합 설계 설계 목적 구체화하기				

[회의내용 요약]

1. LiDAR 센서값 받아오는 과정 코딩 후 Rviz통해 확인하기
2. 빅스비로 어플 실행하고 음성 입력해서 아두이노 통해서 출력 시켜보기



향후계획 / 다음 목표

LiDAR로 SLAM 맵핑 마무리하고 PID제어 최대한 진행하기

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.04.15	장소	보통카페	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병춘, 장한솔				
회의주제	종합 설계 계획 및 지팡이 알고리즘 확립				

[회의내용 요약]

1. 모터에 내장된 엔코더값 받아와서 PID제어로 제어 최적화하기
2. 엔코더값과 바퀴 물성 이용해서 오도메트리 추정
3. LiDAR이용해서 SLAM Mapping 하기



향후계획 / 다음 목표

SLAM 진짜 끝내고 PID제어 선정값 토대로 바퀴 제어 해보기

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.04.22	장소	제1공학관 524호	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병준, 장한솔				
회의주제	지팡이 알고리즘 확립 및 종합 설계 계획 재정립				

[회의내용 요약]

1. 저번주에 못했던 SLAM 구현
2. 바퀴 PID제어한거로 구동 로직 짜보기 (메카닉 휠 이용한 회전 등)



향후계획 / 다음 목표
SLAM 맵핑 학교에서 실제로 해보기
RFID 사용하기

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.04.29	장소	보통 카페	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병춘, 장한솔				
회의주제	기계적 설계 및 제품 모델링, 시장 조사와 특허 조사				

[회의내용 요약]

1. SLAM 실제 영상 찍을 곳에서 Mapping 잘 되는지 확인
2. PID제어 토대로 메카닉 휠 구동하는 코드 확인
3. RFID 센서 입력값 받아오는 법 코딩



향후계획 / 다음 목표

RFID로 데이터 읽고, 사용자에게까지 음성으로 전달하기
카메라 물건인식 최대한 해보기

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.05.06	장소	도서관 회의실	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병준, 장한솔				
회의주제	종합 설계 보고서 작성 및 그를 위한 근거 확립				

[회의내용 요약]

1. Yolo v3와 카메라 센서값 이용해서 물건 인식 하기
2. 카메라 센서 값 받아오기
3. RFID 값 받아온거 사용자(어플)에 정보 보내기
4. 우노에서 받아온 물건 정보 음성으로 사용자에게 전달하기



향후계획 / 다음 목표

카메라 사용해서 물건인식

지금까지 해왔던거 물건인식 제외하고 합쳐서 사용해보기

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.05.13	장소	보통 카페	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병춘, 장한솔				
회의주제	보고서 작성과 그에 필요한 근거 확립				

[회의내용 요약]

1. Yolo 이용해서 물건인식
2. 앱 시작부터 주행시작, RFID 인식까지 통합하기
3. 통합된 S/W 토대로 실제 주행해보기



향후계획 / 다음 목표

각 CPU간 통신 꼬이는 이유 확인하고 고치기

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.05.20	장소	도서관 회의실	작성자	홍기원
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병춘, 장한솔				
회의주제	보고서 작성과 그에 필요한 근거 확립				

[회의내용 요약]

1. 아두이노 - 젯슨나노 통신간에 노드 순서 꼬이는 이유 확인해보기
2. 보고서 쓸 내용 확인 해 보고 자료 미리미리 정리 해 두기
3. 오도메트리 추정이 가끔 불안정한데 이유 알아보고, SLAM 네비게이션까지 발전 시도



향후계획 / 다음 목표
구동 영상 촬영
SLAM 네비게이션 시도해보기
보고서 작성

기계공학 종합 설계1 5분반 EMOM조					
회의일자	2024.05.27	장소	도서관 회의실	작성자	윤상
프로젝트명	기계공학 종합 설계				
참석자	홍기원, 한준모, 윤상, 박병춘, 장한솔				
회의주제	보고서 작성과 그에 필요한 근거 확립				

[회의내용 요약]

1. 실제 구동영상 촬영
2. 보고서 작성
3. SLAM 네비게이션 최대한 해보기



향후계획 / 다음 목표

종합설계 발표 준비 및 남은 기간 추가/보완 사항 생각해보기