

2021년 항공우주및소프트웨어공학 전공 종합설계 제안서

과제명: 전장 상황에서 정찰 임무 수행이 가능한 드론 개발
(Development of the UAV
for reconnaissance in battlefield situations)

팀명: 천리안

팀원: 전병륜(팀장), 김강산, 김송섭, 박보근, 전동환

작성일: 2021년 3월 16일

지도교수: 이선아 (서명)

목 차

1. 개요	1
1.1 과제의 개요	1
1.2 추진배경 및 필요성	1
2. 과제의 목표 및 내용	5
2.1 목표	5
2.2 연구/개발 내용	6
2.3 과제의 결과물	10
2.4 기대효과 및 활용방안	13
3. 주요 진척 상황 및 계획	3
3.1. 진척 상황 요약	3
3.2. 계획	4
4. 프로젝트 팀 구성 및 역할 분담	19
5. 과제 예산 및 내역	20
6. 과제 추진 일정	20

1. 개요

1.1. 과제의 개요

본 과제는 전장상황과 같은 사전 정보가 없는 제한된 환경에서 표적탐지와 지형탐색과 같은 정찰임무를 수행 가능한 자율비행 드론을 개발하는 것이다. Label 된 Dataset을 활용한 개, 군인, 표지판, 활자 등의 표적탐지 임무 수행, 카메라로 촬영한 이미지 데이터를 활용한 위치 및 자세 추정, LiDAR 센서를 이용하여 2차원 지도 생성, 탐색한 지형정보를 활용한 경로탐색 기술을 구현한다.

1.2. 추진배경 및 필요성

1.2.1. 드론봇 전투체계

육군은 2017년 ‘5대 게임 체인저(Game Changer)’라는 새로운 군사력 건설개념이 처음으로 공개했다. 5대 게임 체인저는 군 병력 감축과 복무기간 단축 등 안보환경 변화, 북한 핵·미사일 위협의 고도화, 전쟁 패러다임의 변화 등에 대응하기 위해 건설할 5대 핵심전력이다. 이 중 드론봇 전투단은 병력 감축 태풍을 맞은 육군이 가장 중점을 두고 있는 분야 중 하나이다. 중대부터 군단에 이르기까지 다양한 제대에서 다양한 목적을 가진 소형 무인기와 민간 상용 드론 등을 활용해 핵심 표적에 대한 정찰 능력과 타격수단을 연동하는 등 드론 전투단을 편성하겠다는 것이다.¹⁾



그림 1 드론봇 전투단(육군 제공)

드론봇 군사연구센터와 드론 교육센터를 설립하고 드론봇 전투단을 창설했지만, 정

1) 유용원, ‘육군의 5대 게임 체인저 전쟁 판도 바꾼다’, 주간조선, 2018.01.29

책과 비교하면 국내 무인 전투기술 수준은 다소 부족한 실정이다. 지상로봇과 항공로봇의 기술수준은 선진국에 대비 세계 7위 수준이고, 드론봇 업체도 대부분 영세하여 연구 인력이나 투자 예산도 부족하며, 주요 부품을 해외에서 수입하여 활용하고 있다²⁾.

1.2.2. 군용 드론 시장 현황 및 전망

군사용 무인기(UAV)의 저렴한 가격과 조종자의 안전확보 등 전장상황에서의 강점이 두드러지면서 군용 드론 시장 규모가 커지고 있다. 전 세계 군용 드론 시장은 2018년 121억 3,000만 달러에서 연평균 성장률 12.00%로 증가하여, 2025년에는 268억 2,000만 달러에 이를 것으로 전망된다(그림2 참조). 우리나라의 군용 드론 시장은 2018년 3억 2,910만 달러에서 연평균 성장률 11.72%로 증가하여, 2025년에는 7억 1,500만 달러에 이를 것으로 전망된다³⁾(그림3 참조).



그림 2 글로벌 군용 드론 시장 규모 및 전망(출처: “군용 드론 시장”, 연구개발특구진흥재단(2019))



그림 3 우리나라 군용 드론 시장 규모 및 전망(출처: “군용 드론 시장”, 연구개발특구진흥재단(2019))

특히 정보·감시·정찰·표적획득용 드론은 2018년 88억 7,120만 달러에서 연평균 성장률 11.78%로 증가하여 2025년에는 193억 4,540만 달러에 이를 것으로 추산된다⁴⁾(그림4 참조).

2) 류창수, 김명환, 정영진, “드론봇 전투부대 편성 및 운용개념에 관한 연구”, 국방과 기술, 480, 70, 2019

3) “군용 드론 시장”, 연구개발특구진흥재단(2019)

4) 3)과 동일

(단위: 백만 달러)

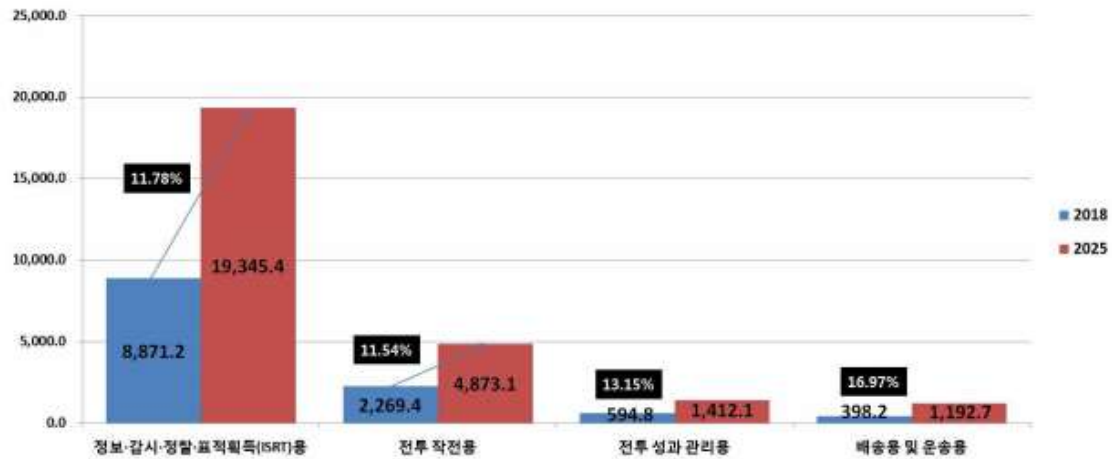


그림 4 글로벌 군용 드론 시장의 용도별 시장 규모 및 전망(출처: “군용 드론 시장”, 연구개발특구진흥재단(2019))

1.2.3. 임무수행 드론에 대한 관심

임무수행이 가능한 드론 경진대회가 매년 개최되고 있다. MathWorks에서 주최하는 ‘미니드론 자율비행 경진대회’, 육군본부에서 주최하는 ‘드론봇 챌린지 대회’, 방위사업청에서 주최하는 ‘전장상황에서의 자율비행 기술 경진대회’ 등이 있다. 드론봇 챌린지 대회의 경우 6개의 종목(원거리 정찰, 건물내부 정찰, 공격, 다수비행체 제어, 수송, 수풀지역 극복)에 대해 평가하며, 전장상황에서의 자율비행 기술 경진대회의 경우 자율비행과 표적탐지, SLAM이 실시간으로 가능한 드론을 요구한다.

1.2.4. 과제의 필요성

드론봇 전투단 편성을 위해선 드론봇에 대한 기술 수준이 뒷받침되어야 하지만, 국내 무인 전투기술은 다소 부족하다. 드론은 악천후시 운용, 적전자공격 회피/전자전 수행, 적R/D 피탐 및 회피, 침투하는 적 드론 탐지, 장기체공가능/이·착륙 공간 최소화, 저소음, 시설내부 및 정밀 감시표적 정보획득, 드론통합관제/조종, 실시간 수집영상 전송/분석체계 등 9가지 측면에서 제한사항이 있다⁵⁾.

9가지 제한사항 중 시장 규모가 가장 큰 정보·감시·정찰·표적획득과 관련이 있는 ‘시설내부 및 정밀 감시표적 정보획득’을 해결하기 위해서는 GPS 및 통신이 불가능한 전장상황에서 표적탐지 및 지형탐색이라는 정찰 임무가 수행 가능한 자율비행 드론개발이 필요하다.

5) 2)와 동일

구 분	현재 제한사항	미래 요구능력
① 약천후시 운용	• 강우 5mm/h, 풍속 10m/s 미만시 운용	• 약천후시 운용 가능(강우 : 10mm/h, 풍속 : 24m/s) * 프로펠러 없는 전천후 드론 개발중
② 적 전자공격 회피/전자전 수행	• 항재밍안테나 미장착, 상용주파수 사용 • 전자전 수행능력 제한	• 소형 항재밍안테나 장착, 드론 전용 주파수 확보 • GPS없이 영상 및 센서 기반으로 자율비행 • 전자전드론 운용
③ 적 R/D 피탐 및 회피	• 적지중심작전지역에서 적 R/D 탐지에 취약 • 개전초 적 방공체계 표적식별 제한	• (고정익)스텔스 기술 적용(형태, 재질, 도색) • 기만드론을 운용하여 적 방공체계 표적 식별
④ 침투하는 적 드론 탐지	• 야군 저고도탐지레이더로는 적 소형드론 탐지 제한	• 적 소형드론(0.6×0.6m) 탐지 가능 • 드론 통합관제체계와 연동하여 피·아 드론 탐지 및 식별 가능
⑤ 장기 체공 가능 /이·착륙 공간 최소화	• (회전익드론)평균 30~50분 비행 * 잦은 배터리 교체로 간단없는 표적 감시 제한 • (고정익드론)이·착륙 공간 부족	• 하이브리드엔진(12시간 이상), 수소연료전지(4시간 이상) 사용 • 수직 이·착륙형 드론 운용
⑥ 저소음	• 고도 100m 이격된 상태에서 전화벨 소리 크기로 인지(60db) * 적 인원에게 노출 가능	• 프로펠러 없는 저소음드론 운용 * 고도 100m 이격된 상태에서 소음청취 제한(40db 이하)
⑦ 시설내부 및 정밀 감시표적 정보 획득	• 시설내부에서 통신 및 GPS수신 제한 • 시설내 장애물 산재시 비행이 제한 • 정밀감시표적에 근접접근 및 은밀감시 제한	• 시설내부 자율비행 및 3차원 지리공간정보 활용 • 비행 중 필요시 지상 이동(차량, 궤도)하면서 운용 가능 • 조류·곤충 등을 모방한 생체모방형 초소형드론 운용
⑧ 드론통합관제/조종	• 드론 통합관제 제한 * 제대별 다양한 드론을 동시 운용시 충돌 가능성 내재 • 1:1 드론 조종	• 드론용 중앙방공통제소 구축 • 군집드론 알고리즘을 적용한 1다수 드론 조종 가능
⑨ 실시간 수집영상 전송/분석체계	• 장거리·대용량 영상정보 전송 제한 • 전 출처 영상 분석체계 부재	• 통신중계드론을 운용하여 실시간 대용량 영상정보 송·수신 • 전 출처 영상을 실시간 융합·분석·처리체계 구축

그림 5 제한 사항 및 미래 요구능력(발췌: 드론봇 전투부대 편성 및 운용개념에 관한 연구)

1.2.5. 과제의 창의성 혹은 실용성

본 과제의 창의성은 드론을 이용한 자율 임무 수행이다. 전 세계적으로 드론에 대한 관심은 증가하고 있지만, 드론을 이용한 자율 비행을 통한 임무 수행이라는 주제에 대한 접근 장벽은 높다. 시중에 판매되는 서적을 살펴보면 드론 제작과 관련된 서적은 다양하지만, 드론을 이용한 자율비행, SLAM, 사물인식 등을 동시에 다루고 있는 서적은 전무하다. 본 활동의 보고서는 서적을 대신하여 드론을 이용한 임무수행에 대한 기초지식을 제공할 수 있으며, 이러한 캡스톤 주제는 도전적이라고 할 수 있다.

본 과제의 실용성은 국방과학기술연구소에서 주최하는 “전장상황에서의 자율비행 기술경진대회”에서 같은 주제로 대회를 마련했다는 점에서 증명된다. 해당 대회에서 평가지표로 다루는 ‘자율비행 및 장애물 회피 실적 및 수행 경험’등은 국방과학기술연구소에서 이러한 기술을 필요로 한다는 점을 보여준다.

2. 과제의 목표 및 내용

2.1. 목표 (피드백 필요)

본 과제의 목표는 전장상황에서 정찰 임무 수행이 가능한 드론 개발이다. 본 과제의 전장 상황은 도심지 내의 전투로, GPS 및 통신이 불가하고 사전 정보가 없는 실내 환경이다. 이를 위해서 드론은 스스로 1)비행계획을 수립/비행하며 카메라를 통해 군인, 표지판, 활자와 같은 2)표적을 탐지하고, LiDAR 센서를 이용한 지형을 탐색한다. 표적과 지형 데이터를 이용해 3)표적의 정보가 나타난 2차원 지도를 생성한다.

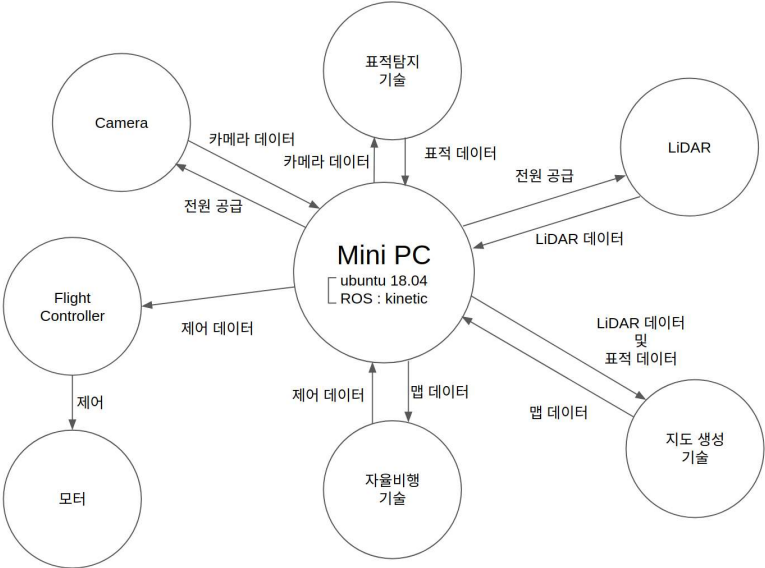


그림 6 Context Diagram

2.2. 연구/개발 내용(피드백 필요)

임무를 Usecase Diagram으로 나타내면 그림 8과 같다. Actor는 5개로 사용자, LiDAR, Camera, Motor와 Flight Controller로 나타낼 수 있다. LiDAR 센서는 지도생성기술, Camera 센서는 표적탐색기술 Usecase와 관계를 맺어, 외부로부터 수집한 데이터를 Mini PC로 전송한다. 자율비행기술 Usecase는 지도생성기술 Usecase로부터 데이터를 제공받아 비행경로를 계획하고, 계획한 데이터를 바탕으로 Flight Controller는 Motor를 제어하여 자율비행임무를 수행한다. 임무를 통해 완성된 data는 USB를 통해 옮겨지며, 운용자의 PC의 SW를 통해 data를 지도자료로 변환한다. 지도자료는 2차원 지형 및 인식한 표적의 개수와 위치정보를 담고 있다. 임무 수행 중 위급상황이 발생하는 경우 수동제어로 전환한다.

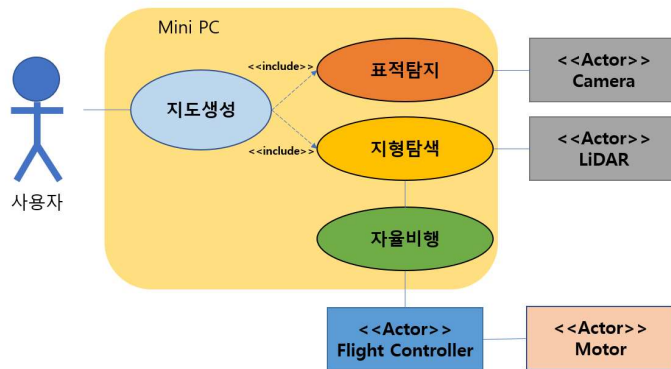


그림 7 usecase diagram

시스템	천리안 드론
<u>유스케이스명</u>	지도생성
<u>액터명</u>	사용자
<u>개요</u>	사용자는 표적탐지 Data, 지형탐색 Data를 통해 만들어진 지도를 확인한다.
<u>사전 조건</u>	Mini PC는 지형탐색 Data와 표적탐지 Data를 가지고 있어야한다.
<u>기본 흐름</u>	1. Mini PC는 지형탐색으로 생성된 Data와 표적탐지로 생성된 Data를 위치를 기준으로 합한다. 2. 사용자는 USB를 통해 통합된 지도를 얻는다.
<u>대체 흐름</u>	-
<u>예외 흐름</u>	-
<u>사후 조건</u>	사용자는 USB를 통해 생성된 지도를 확인한다.

시스템	천리안 드론
<u>유스케이스명</u>	표적탐지
<u>액터명</u>	Camera
<u>개요</u>	Camera는 관찰한 Data를 Mini PC로 보내서 표적을 탐지한다.
<u>사전 조건</u>	Mini PC에 학습된 모델이 있어야 한다.
<u>기본 흐름</u>	<ol style="list-style-type: none"> 1. Camera는 사진 Data를 수집한다. 2. Camera는 수집한 Data를 Mini PC로 보낸다. 3. 표적탐지모델은 수집된 Data를 통해 표적을 분류한다.
<u>대체 흐름</u>	기본흐름2에서 Precision이 설정해 놓은 임계 값 이하면 저장하지 않는다.
<u>예외 흐름</u>	-
<u>사후 조건</u>	-

시스템	천리안 드론
<u>유스케이스명</u>	지형탐색
<u>액터명</u>	LiDAR
<u>개요</u>	LiDAR는 센서를 통해 지형 Data를 수집하여 Mini PC로 보낸다.
<u>사전 조건</u>	-
<u>기본 흐름</u>	<ol style="list-style-type: none"> 1. LiDAR는 센서를 통해서 주변 지형의 Data를 수집한다. 2. LiDAR는 수집한 Data를 Mini PC로 보낸다. 3. Mini PC는 받은 지형 Data를 저장한다.
<u>대체 흐름</u>	-
<u>예외 흐름</u>	-
<u>사후 조건</u>	-

시스템	천리안 드론
<u>유스케이스명</u>	자율비행
<u>액터명</u>	Flight Controller, Motor
<u>개요</u>	Flight Controller는 Mini PC에서 정해진 경로에 맞게 Motor를 제어하여 자율비행을 한다.
<u>사전 조건</u>	Mini PC는 지형 Data를 가지고 있어야 한다.
<u>기본 흐름</u>	<ol style="list-style-type: none"> 1. Mini PC는 저장 돼 있는 지형 Data와 경로탐색을 통해 진행 경로를 결정한다. 2. Mini PC는 결정한 진행경로를 Flight Controller에 보낸다. 3. Flight Controller는 결정된 진행 경로에 맞게 Motor를 조절한다.
<u>대체 흐름</u>	-
<u>예외 흐름</u>	-
<u>사후 조건</u>	-













그림 11 usecase specification

● 자율비행

드론의 자율비행 레벨은 Level 0부터 Level 5까지 총 6단계로 구분할 수 있다. 본 과제에서 목표로 하는 자율비행레벨은 Level 2 단계로 비행 임무는 드론이 제어하되, 위급상황 발생 시 수동제어로 조종사가 제어한다.

지형탐색 결과를 바탕으로 비행경로를 계획한다. Flight Controller는 계획된 비행 경로를 바탕으로 모터를 제어하여 경로를 따라 주행한다. 주행의 종료는 비행성능에 기반을 둔 비행시간과 탐색가능성을 바탕으로 드론이 결정한다.

만약 응급상황 발생 시, 사전에 설정한 수동제어 모드를 통해 드론의 비행을 운용자가 제어할 수 있도록 한다.

DRONE INDUSTRY INSIGHTS						
THE 5 LEVELS OF DRONE AUTONOMY						
Autonomy Level	Level 0	Level 1	Level 2	Level 3	Level 4	Level 5
Human Involvement						
Machine Involvement						
Degree of Automation	No Automation	Low Automation	Partial Automation	Conditional Automation	High Automation	Full Automation
Description	Drone control is 100% manual.	Pilot remains in control. Drone has control of at least one vital function.	Pilot remains responsible for safe operation. Drone can take over heading, altitude under certain conditions.	Pilot acts as fall-back system. Drone can perform all functions 'given certain conditions'.	Pilot is out of the loop. Drone has backup systems so that if one fails, the platform will still be operational.	Drones will be able to use AI tools to plan their flights as autonomous learning systems.
Obstacle Avoidance	NONE	SENSE & ALERT		SENSE & AVOID	SENSE & NAVIGATE	

Source: DRONEII.COM

Date: March 12th 2019

DRONEII.COM
DRONE INDUSTRY INSIGHTS

© 2019 All rights reserved | DRONE INDUSTRY INSIGHTS | Hamburg, Germany | www.droneii.com

그림 12 드론 자율 비행 레벨(발췌: <https://dronelife.com/2019/03/11/droneii-tech-talk-unraveling-5-levels-of-drone-autonomy/>)

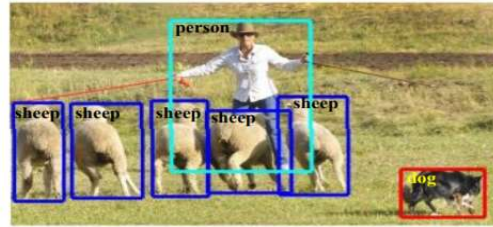
● 표적탐지

수집된 이미지 또는 제공된 이미지에서 사전에 지정된 클래스의 객체가 존재여부를 판단하는 기술로 Object Classification, Generic Object Detection, Segmantic Segmentation, Object Instance Segmentation등으로 분류된다. 우리는 Bounding Box를 이용하는 Generic Object Detection을 목표로 한다.

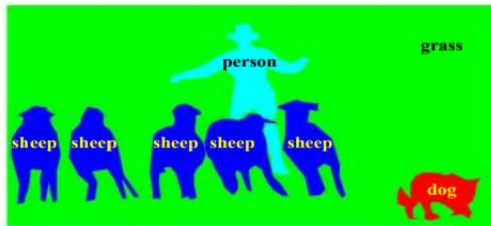
카메라 센서를 이용해 수집한 데이터에서 군인, 표지판, 활자 등을 인식하고 위치정보를 기록한다. 탐지한 표적의 클래스와 위치정보는 지도 생성에 사용된다.



(a) Object Classification



(b) Generic Object Detection (Bounding Box)



(c) Semantic Segmentation



(d) Object Instance Segmentation

그림 13 객체인식기술의 종류(발췌: Liu et al. Deep Learning for Generic Object Detection: A Survey, 2018)

● 지형탐색

지형탐색이란 SLAM을 바탕으로 지형 정보를 데이터화 하는 기술로 Odometry와 Mapping으로 구분 할 수 있다. Odometry란 자신의 상대속도를 바탕으로 드론의 궤적을 추정하는 기술이며, Mapping이란 지형정보와 추정궤적을 기반으로 공간에 대한 지도를 생성하는 기술이다. 본 과제에서는 LiDAR 센서를 이용하여 SLAM기술을 구현한다. 생성된 지형 정보는 자율 비행과 2차원 통합 지도생성에 사용된다.



그림 14 Cartographer's 2D SLAM(발췌: Wolfgang et al, "Real-Time Loop Closure in 2D LIDAR SLAM)", ICRA(2016), 1271)

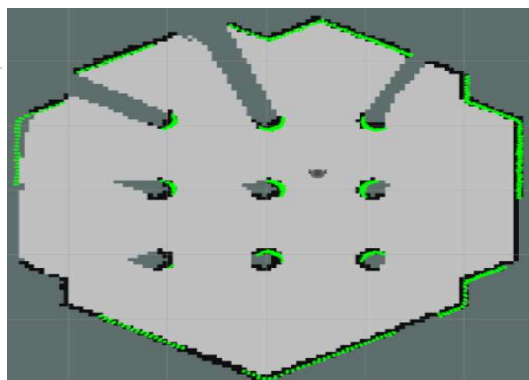


그림 15 SLAM in ROS (발췌: <https://emannual.robotis.com/docs/en/platform/turtlebot3/slam/>)

2.3. 과제의 결과물

항목	예상 결과물	결과물에 대한 설명 요약
설계	드론 HW 설계	임무 형상 및 설계 요구도
		장비 규격 및 구성
		CATIA 3D 모델링 및 도면
	SW 아키텍처 설계	Context diagram
		Usecase diagram
		class diagram
		sequence diagram
		아키텍처 설계
수학적 분석	드론 specification	로터 블레이드 공력 성능 계산
		드론 공력성능 계산
		동역학 모델링
		제어기 분석
시뮬레이션	구조해석	CATIA를 이용한 드론의 무게 중심 등을 분석
	표적탐지기술 검증	표적탐지율 및 실시간성을 바탕으로 평가
	지형탐색기술 검증	지도생성 정확도 비교
	자율비행기술 검증	경로 탐색/비행 기술 구현
		지형탐색기술과 통합
		Coverage 평가
	가상환경 시험	가상환경 구축 및 비행시스템 시험
실험 및 제작	프로토타입 제작 및 시험	설계한 시스템들이 모두 구현된 드론 제작 및 시험
	시험 평가	시험결과 평가

2.3.1. 설계

● 드론 HW 설계

- 1) 시나리오 분석을 바탕으로 임무수행에 필요한 specification을 바탕으로 센서를 파악한다.

구분	최소사양	구분	최소사양
중량	3kg 이하	변속기	ESC(4EA): $\geq 20A$ OPTO
비행 시간	10분 이상	Mini PC	i7 NUC 미니PC, CPU 3.5GHz 이상 RAM 16GM이상 및 SSD 250GM 이상
크기	쿼드 로터 프레임 대각선 휠베이스 =650mm	3D 카메라	글로벌 셔터 Depth Range: 0.13~3m Depth of FOV: 85*85
프로 펠러	12*4.5인치(4ea)	IMU	3축 자이로센서 3축 가속도계 3축 자기계 압력센서
모터	Recommended Load $\geq 600g/axis$ Max. Load $\geq 1,600g/axis$	2D LiDAR	측정거리: 30m 이상 Max. Range: 60m Rotational Frequency: $\geq 5Hz$ Sampling Frequency: 4K~10K
배터리	리튬 폴리머 1EA($\geq 6000mAh/22.2V$)	1D LiDAR	측정거리 최소 0.1m, 최대 30m

- 2) Payload 계산을 통한 드론 부품을 결정한다.
- 3) CATIA를 이용한 3D 모델링 및 도면을 작성한다.

● SW 아키텍처 설계

- (1) 시나리오 분석을 통해 프로젝트 요소들 간의 입출력을 표현하는 Context Diagram을 작성한다.
- (2) HW 설계에서 작성한 임무 요구도를 바탕으로 Usecase Diagram을 작성한다
- (3) 실시간 동작들을 확인하기 위한 Sequence Diagram을 작성한다.
- (4) 다양한 모델요소의 구성 및 배열을 표현하기 위해 클래스 및 Package의 입출력의 상관관계를 나타내는 class diagram을 작성한다.
- (5) 작성한 diagram을 바탕으로 아키텍처 구조를 완성한다?

2.3.2. 수학적 분석

● 드론 Specification

- (1) 날개 요소 운동량 이론을 바탕으로 프로펠러의 공력성능을 계산한다.
- (2) 드론의 공력성능과 장비를 고려하여 비행성능을 계산한다.
- (3) 드론의 동역학식을 계산하고 동역학적 특성을 파악한다.
- (4) 드론의 동역학 특성을 바탕으로 제어기를 선택/분석한다.

2.3.3. 시뮬레이션

● 구조해석

CATIA를 이용한 3D모델링 파일을 이용해 무게중심 및 관성모멘트를 파악한다.

● 표적탐지기술 검증

- (1) IoU(Intersection over Union)를 기준으로 표적탐지 알고리즘을 조사한다.
- (2) 국방과학기술연구소(ADD)에서 제공한 Dataset을 이용하여 모델을 학습시킨다.
- (3) ADD에서 제공한 Validation Dataset을 이용해 모델의 표적탐지율을 mAP와 Macro F1 Score, 실시간성을 $\frac{\text{평가 데이터셋의 총 이미지 수}}{\text{소요된 시간}}$ 으로 평가한다.
- (4) 실제 환경에서 표적탐지율과 실시간성을 평가한다.

● 지형탐색기술검증

- (1) 2D LiDAR 센서를 이용한 SLAM 기술을 조사한다.
- (2) 선정된 센서와 호환될 수 있도록 기술을 최적화한다.
- (3) 지형탐색기술 검증을 위한 환경을 설계 한다.
- (4) 시뮬레이션을 진행하여 생성된 지도와 설계한 지도의 정확성을 비교한다.

● 자율비행기술검증

- (1) 경로계획기술 및 경로비행기술을 조사한다.
- (2) 지형탐색기술을 기반으로 경로계획기술을 구현한다.
- (3) 경로계획기술의 시뮬레이션을 가상환경에서 진행한다.
- (4) 시뮬레이션 결과를 바탕으로 Coverage를 평가한다.
- (5) 경로비행기술을 구현하여 가상환경에서 시뮬레이션을 진행한다.

- **가상환경 시험**

- (1) Gazebo/Airsim과 같은 가상환경을 이용해 비행시험을 위한 환경을 구축한다.
- (2) 구축한 환경에서 시험비행을 진행하여 전반적인 시스템의 완성도를 점검한다.

2.3.4. 실험 및 제작

- **프로토타입 제작 및 시험**

- (1) 작성한 도면을 바탕으로 센서를 조합하여 드론을 제작한다.
- (2) 임무수행을 위한 프로그램을 통합하여 Mini PC에 적용한다.
- (3) 전장환경과 유사한 환경을 설계/구축한다.
- (4) 구축한 환경에서 비행시험을 진행한다.

- **시험결과 평가**

비행시험을 통해 수집한 데이터를 시뮬레이션에서 검증한 결과와 비교한다.

2.4. 기대효과 및 활용방안

2.4.1. 기대효과

- **기술적 측면**

본 개발 과제에서 표적탐지 기술을 통해 전장상황에서 위험요소인 군인(군인 특공, 군인파병, 여군, 여군특공 등), 표지(시작, 출구, 끝), 마크(방사능 생화학)등을 식별한다. 지형탐색기술은 사전정보가 없는 위험지역에 대하여 2차원 지도를 작성한다. 자율비행기술은 지형탐색기술을 통해 획득한 지도를 이용하여 시야가 없고, 통신이 불가능한 환경에서도 정찰이 가능하다.

- **사회적 측면**

기존의 드론은 운용을 위해 전문 인력이 요구되므로 부대 여건에 따라 사용이 제한된다. 본 드론은 운용을 위한 교육을 간소화하므로 운용 편의성을 살려 전투체계에 다양성을 추가할 수 있다. 또한 경진대회 및 학술회에 출전하여 기술 소개 및 교류를 통한 드론 기술 분야의 발전을 촉진한다.

2.4.2 활용방안

- SLAM, 표적탐지, 자율비행 각각의 기술들이 다양한 기술들과 융합하기에 용이하기 때문에 전체적인 프로젝트를 코드와 명세를 Github에 올려 형상관리 및 배포를 함으로써 향후 후배들의 캡스톤, 교 내외 대회 및 KCC 학술대회 등 다양한 프로젝트에 활용되는 것을 기대한다.
- 전장 상황 이외에도 건물 혹은 터널의 붕괴, 테러상황과 복잡하고 위험한 선체 내부, 발전소, 동굴 탐사 등의 상황에서 인명 피해에 대한 부담 없이 정보 수집을 할 수 있다.

3. 주요 진척 상황 및 계획

3.1. 진척 상황 요약

항목		진행내용	진행률(%)
설계	드론HW 설계	임무 형상 및 설계 요구도	100
		장비 규격 및 구성	80
		CATIA 3D 모델링 및 도면	80
	SW 아키텍처 설계	Context Diagram	100
		Usecase Diagram	100
		Class Diagram	20
		Sequence Diagram	0
수학적 분석	드론 Specification	드론 비행 성능 계산	100
		동역학 모델링	20
		제어기 분석	0
컴퓨터 해석	구조해석	CATIA를 이용한 드론의 무게 분석	50
	표적탐지기술 검증	표적탐지 기술 구현	90
		표적탐지 기술 평가	30
	지형탐색 기술 검증	지형탐색 기술 구현	80
		지형탐색 기술 평가	5
		경로탐색/비행 기술 구현	33
		지형탐색기술과 통합	0
		가상환경 구축	30
제작 및 실험	3D 프린팅	추력시험을 위한 구조물 제작	0
		센서 설치를 위한 구조물 제작	0
	추력시험	추력시험 장치 제작 및 추력 성능 측정	0
	프로토타입 검증	프로토 타입 제작	0
		프로토 타입 평가	0

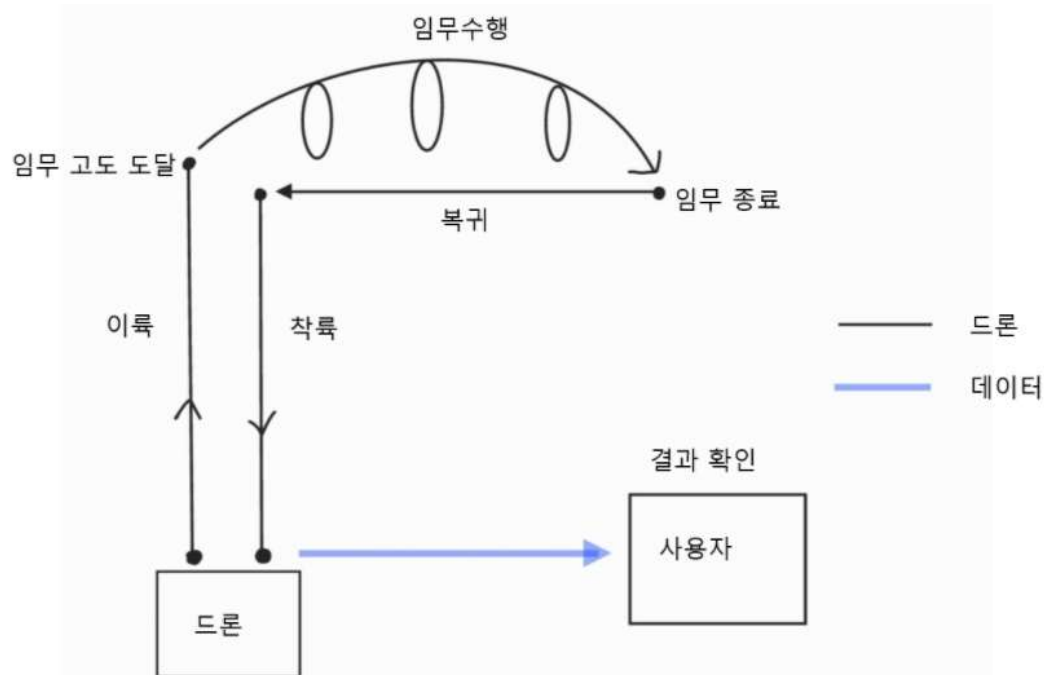
3.2. 설계

제안하는 시스템에 대한 창의형 또는 창의 융합형인 경우 CATIA 도면을 포함하고, 시스템을 구성하는 각각의 요소별 세부적인 설계 내용 등을 기술한다. 가능하다면 (각 구성요소별) 두 가지 이상의 개념(설계원리 등)들에 대한 장단점 비교 분석 포함. 그림과 함께 설명.

3.2.1. Requirement

본 과제는 전장상황에서 정찰임무 수행이 가능한 드론을 개발하는 것이다.

임무요구도(운용 개념도)

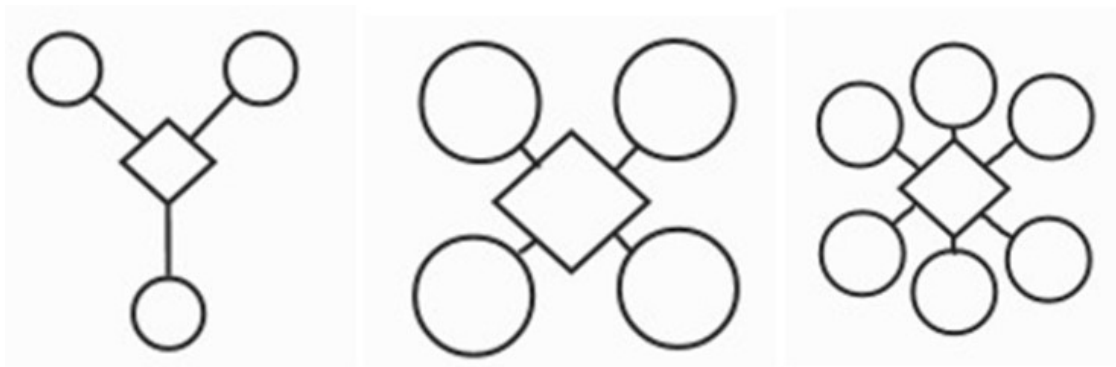


성능 요구조건

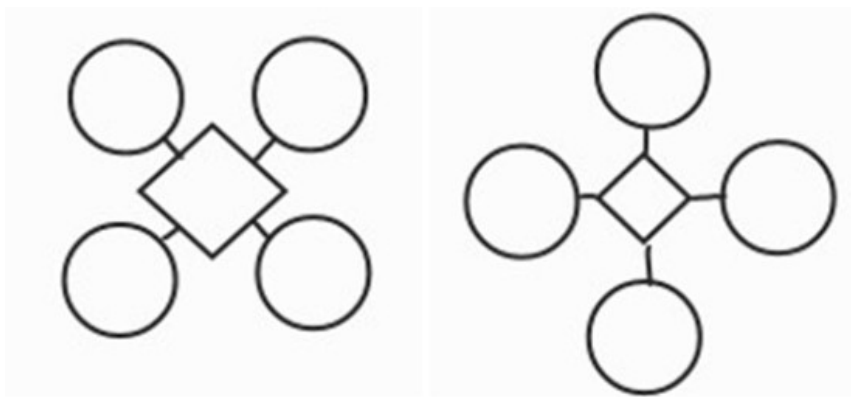
HW		SW	
총 중량	3000g	운영환경	Ubuntu 18.04
형상	쿼드콥터, X자형	Memory	4 GB 이상
크기	600 급	언어	Python, C++
운영 고도	지면 기준 170cm	용량	SSD: 32GB, SD: 64GB
운영 시간	10 min		
탐색 거리	20 m		

3.2.2. 형상 설계

드론은 멀티콥터의 한 종류로 모터의 수에 따라 트라이콥터, 쿼드콥터, 헥사콥터 등으로 나뉜다. 멀티콥터 중 가장 흔한 형상은 쿼드콥터이다. 모터수가 홀수 인 경우, 작용반작용으로 발생하는 기체 회전을 고려한 설계가 필요하다. 모터가 짝수인 경우, 작용반작용 문제가 해결된다. 모터가 2개인 경우는 작용반작용을 해결하지만 자세제어 문제가 남게된다. 쿼드콥터부터는 자세제어문제 또한 해결된다. 헥사 등 모터 수가 많은 경우, 안정적인 비행이 가능하지만 모터 수에 따른 비용증가와 소모전류 증가 등의 문제가 발생한다. 따라서 설계 구조가 가장 간단하고, 부수적인 문제가 가장 적은 쿼드콥터 형상으로 설계했다.

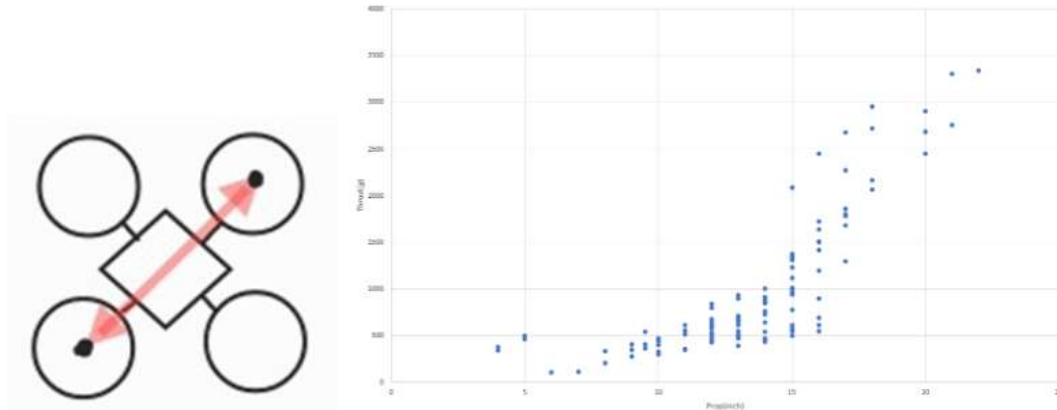


드론은 정면 위치에 따라 X 형과 + 형으로 구분할 수 있다. X 형 드론은 피치/롤 조종 중 하나만 조종하여도 모터 전체의 출력이 변화하여 빠른 반응속도와 안정적인 비행이 가능하다. 반면에 +드론의 경우 피치/롤 중 하나만 조종하는 경우, 한쌍의 모터 출력만 변화하여 반응이 X 형보다 느리다. 비행제어의 안정성을 확보하기 위해 X 형으로 설계했다.



Wheel Base Length(WBL)은 대각선상에 위치한 모터의 축간 거리로 크기를 가늠하는 척도이다. 드론의 중량이 클수록 긴 프로펠러를 사용해야하고, 긴 프로펠러는 WBL의 길이를 증가시킨다. 본 드론은 총 중량이 3000g으로 출력이 50~70% 상태

에서 모터 한 개당 750g의 추력을 담당해야한다. 아래는 모터 T-Motor사 드론용 모터의 추력 vs 프로펠러 길이 그래프이다. 본 자료를 바탕으로 750g에 필요한 프로펠러는 12-15 inch임을 확인했다. 프로펠러와 드론 부품 및 payload의 충돌을 방지하고, payload의 배치 공간을 확보하기 위해 프레임은 600급 이상으로 설계했다.



쿼드콥터 형상, X형, 600급 이상 이라는 조건을 만족하는 드론 프레임으로 Tarot IRON-MAN Sport 650 선정하였다.

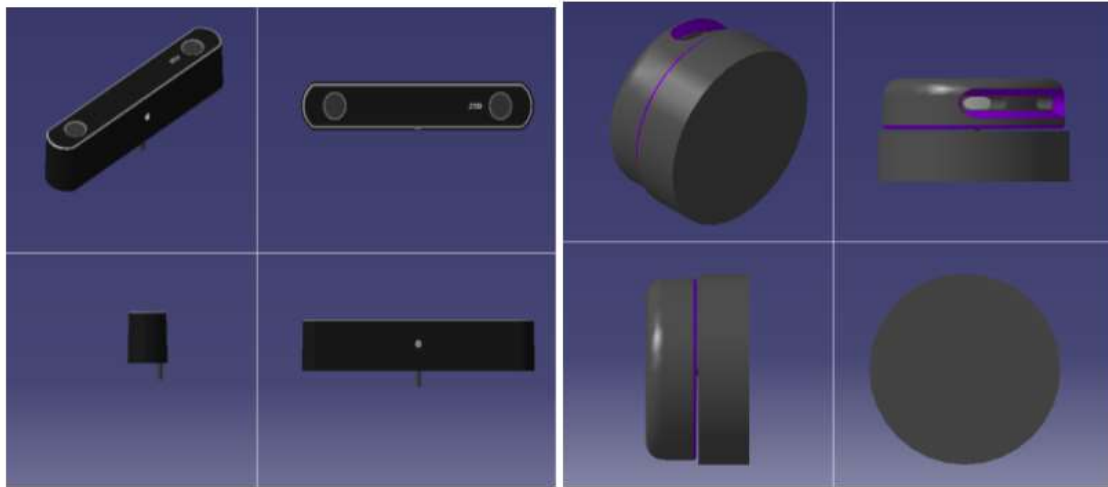
3.2.3. Payload 선정

드론에서 payload란 기체에 탑재가능한 센서 혹은 화물 등을 의미한다. 본 기체에 임무수행을 위해 탑재하는 payload는 카메라센서, 2D LiDAR 센서, 1D LiDAR 센서, Flight Controller, Mini PC 등이 있다.

표적탐지 임무 수행을 위해 사용하는 카메라 센서로 ZED2 Stereo Camera를 선정하였다. 카메라는 드론의 정면에 배치되어 비행동안 촬영을 진행한다. 중량은 Data Sheet 기준 166g으로 무게 중심에 위치하지 않기 때문에 모멘트를 유발한다. 이는 배터리 위치 조정을 통해 보정할 계획이다. 표적탐지를 위한 YOLO V5 모델을 이용해 촬영 영상 속 표적을 탐지하여 표적정보와 위치정보를 추출하며, 추출된 정보는 통합지도 작성 시 사용된다.

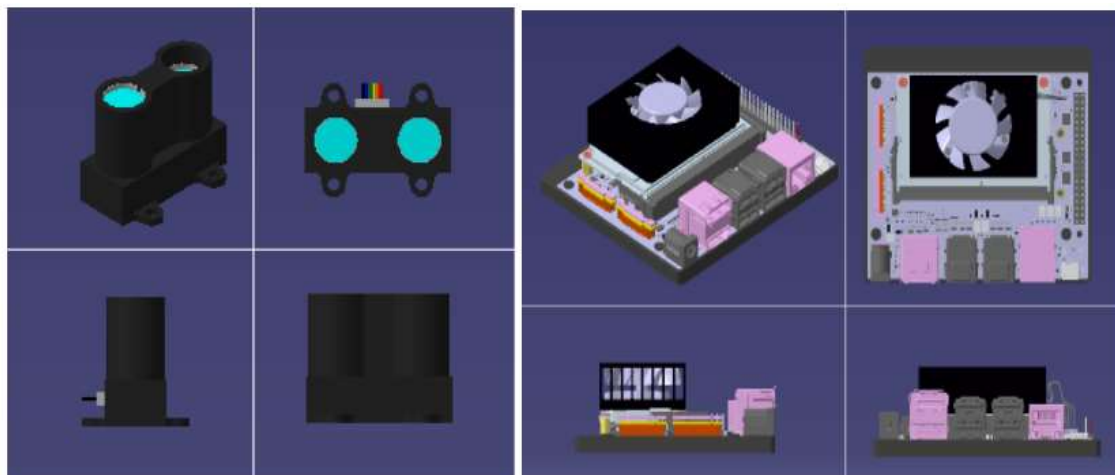
지형탐색 임무 수행을 위해 사용되는 2D LiDAR 센서로 RPLIDAR A3를 선정하였다. 2D LiDAR 센서는 드론의 중앙, 정상에 위치하여 내부 구조물 및 프로펠러로 인한 시야가 가려짐을 방지한다. 이 같은 위치에 배치하기 위해, 3D 프린팅 구조물 설계를 통한 마운트 설계가 필요하다. 2D LiDAR 센서로 수집한 정보를 바탕으로

Cartographer라는 2차원 SLAM 모델을 이용하여 지도를 생성한다. 생성한 지도는 드론의 비행경로 계획과 통합지도 작성에 사용된다.



1D LiDAR 센서는 드론의 고도측정에 사용하기 위해 드론의 하단에 배치된다. 드론의 하단 중심부에는 배터리가 위치하므로, 드론의 중심부가 아닌 배터리 Panel과 Chassis 사이, 배터리 옆에 위치한다. 1D LiDAR는 Pixhawk와 호환되는 센서로, 수집한 데이터는 Pixhawk로 전달되어 PID 제어에 필요한 Z축 데이터로 사용된다.

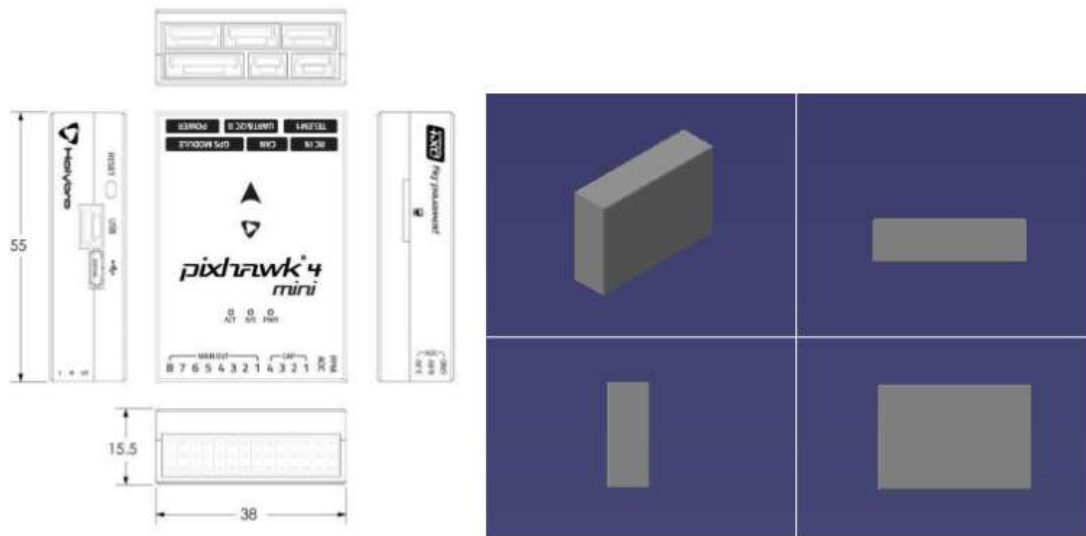
Mini PC는 NVIDIA Xavier NX를 사용한다. 형상은 103mm x 90.5mm x 31mm로 모든 장비 중 가장 많은 면적을 차지한다. 고려할 형상으로 포트와 방열팬이 있다. 카메라센서와 LiDAR센서의 케이블이 가로막히지 않도록 포트 앞을 다른 장비로 가로막아서는 안된다. 방열팬이 상단을 향해 위치해 있어, 제 기능을 발휘하기 위해 2D LiDAR 센서는 방열팬과 거리를 유지해야한다.



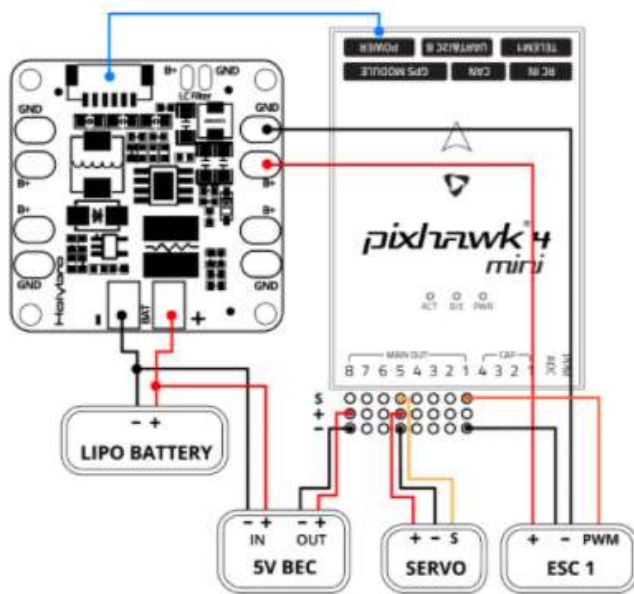
3.2.4. Empty Weight

드론에서 Empty Weight이란 Payload를 제외한 무게로 Flight Controller, 수신기, 모터, ESC, 배터리, Frame 등이 있다.

Flight Controller은 Pixhawk 4 Mini 를 사용한다. FC는 Mini PC에서 얻은 비행 경로를 바탕으로 드론의 자세 및 동작을 제어한다. 사용하는 Pixhawk 4 Mini는 기본형인 Pixhawk 4에서 불필요한 포트들을 제거한 장치로, 더 가볍고 크기가 작다. FC는 PID제어에 이용될 데이터들이 수집되는 IMU센서가 내장되어 있으며, 드론의 앞뒤를 구분하는 장치이다. 때문에 드론의 중앙에 배치되는 것이 중요하지만, Hub Plate의 경우 Mini PC가 그 자리를 차지하고 있다. Chassis와 Hub Plate 사이의 경우, 카본 파이프를 고정하는 Arm Copter Attachment로 인해 Pixhawk 배치를 위한 충분한 공간이 확보되지 않는다. 두 조건들을 회피하기 위해 Chassis와 배터리 Panel 사이에 FC를 배치한다. Pixhawk 4 Mini는 SD카드 돌출부를 제외한다면 직육면체 모양이므로, 직육면체 형상으로 설계하였다.

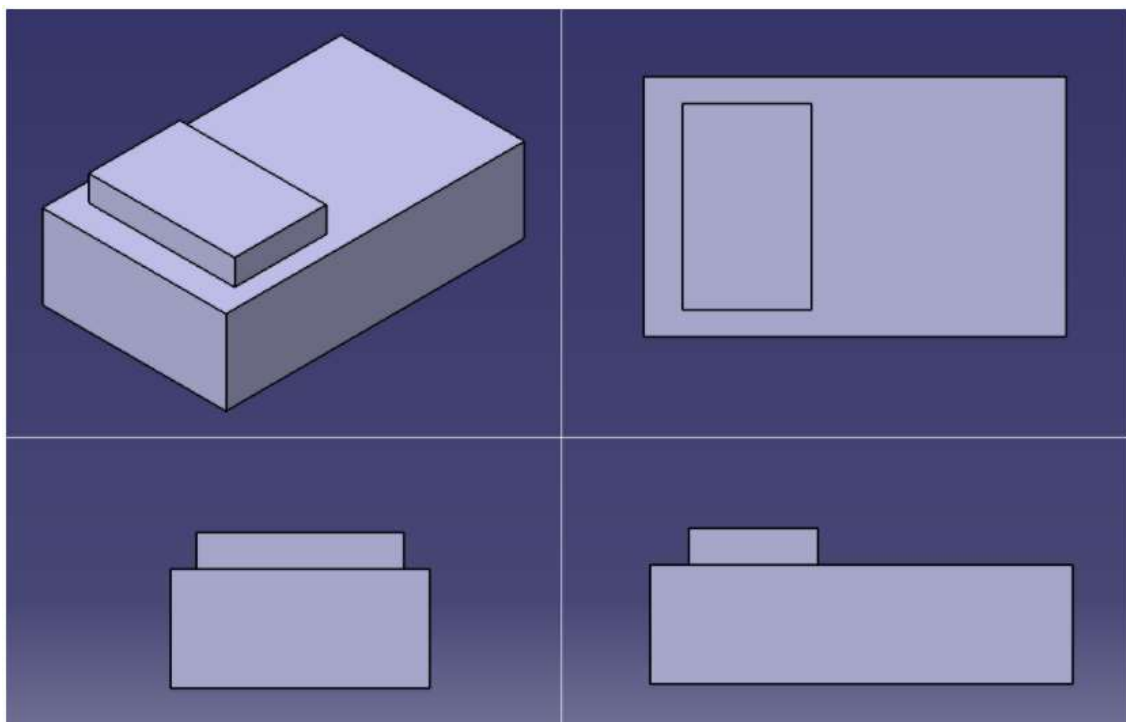


Pixhawk와 함께 고려하는 장비로 Pixhawk Power Module이 있다. Power Module은 이후 언급할 ESC의 BEC(Battery Elimination Circuit) 기능과 관련이 있다. ESC는 OPTO Type과 BEC Type으로 구분할 수 있다. BEC는 모터의 전압이 수신기가 사용할 수 있도록 낮춰주는 기능이며, OPTO는 BEC기능을 지원하지 않는 변속기이다. 드론의 경우 BEC 타입의 변속기를 사용하는 경우, 변속기가 4개가 부착되기 때문에 고장을 방지하기 위해 3개의 변속기는 BEC를 제거해주거나 OPTO 타입의 변속기와 함께 외장 BEC를 따로 장착한다. 파워모듈은 언급한 외장 BEC 기능을 수행하므로 수신기를 위한 별도의 BEC가 필요하지 않다.

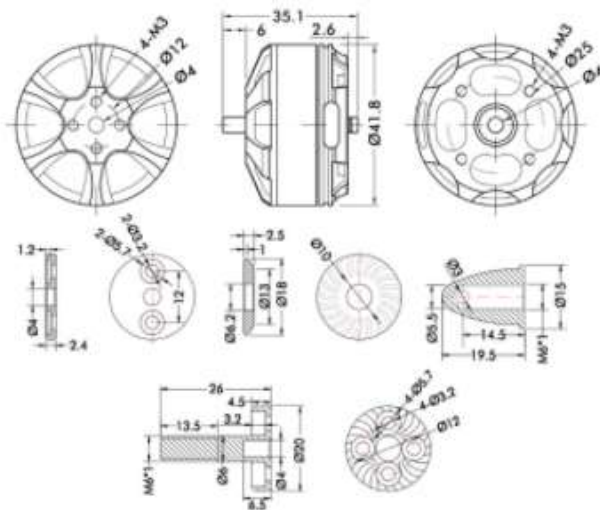


출처: https://docs.px4.io/v1.9.0/en/assembly/quick_start_pixhawk4_mini.html

수신기는 RX601을 선정하였다. 쿼드콥터에 대해 Pixhawk와 수신기가 연결되기 위해서는 5ch이상 지원이 되어야 한다. 본 수신기는 6ch을 지원하며, 보유하고 있는 조종기 devo 7과 호환이 되는 수신기이다. 수신기는 드론의 Hub Plate와 Chassis 사이에 배치된다.

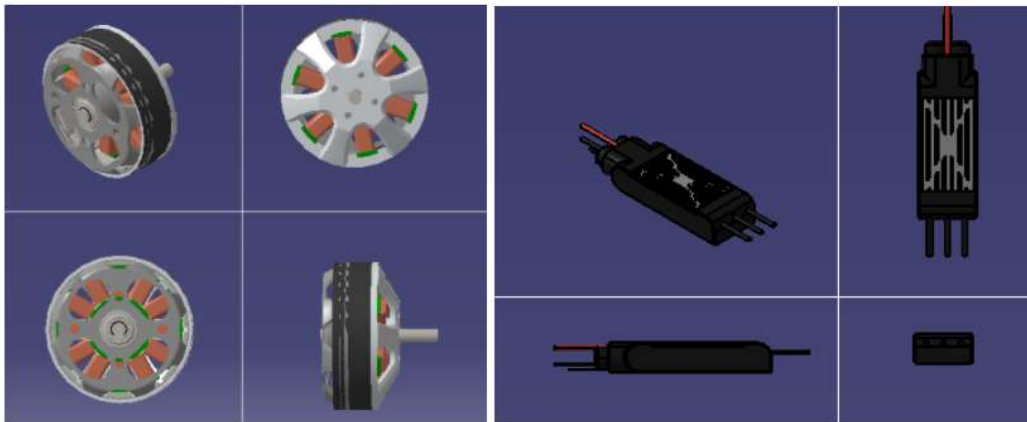


모터는 Requirements를 만족시키는 T-motor 사의 MN3508 700kV로 선정하였다. 프로펠러를 샤프트에 고정하는 방식은 많지만 본 모터는 어댑터 방식과 스크류 고정 방식을 지원한다. 어댑터 방식이란, 샤프트에 어댑터와 프로펠러를 고정한 뒤, 어댑터를 조여 고정하는 방식이며 프로펠러의 중심부를 두껍게 설계하는 플라스틱 프로펠러에 많이 쓰인다. 스크류 고정 방식은 모터에 나사를 이용해 고정하는 방식으로 카본 재질의 얇은 프로펠러에 많이 쓰인다. 플라스틱 프로펠러는 비행 중 파손이 잦아 여유분이 필요하므로, 카본 프로펠러를 선정하였다.

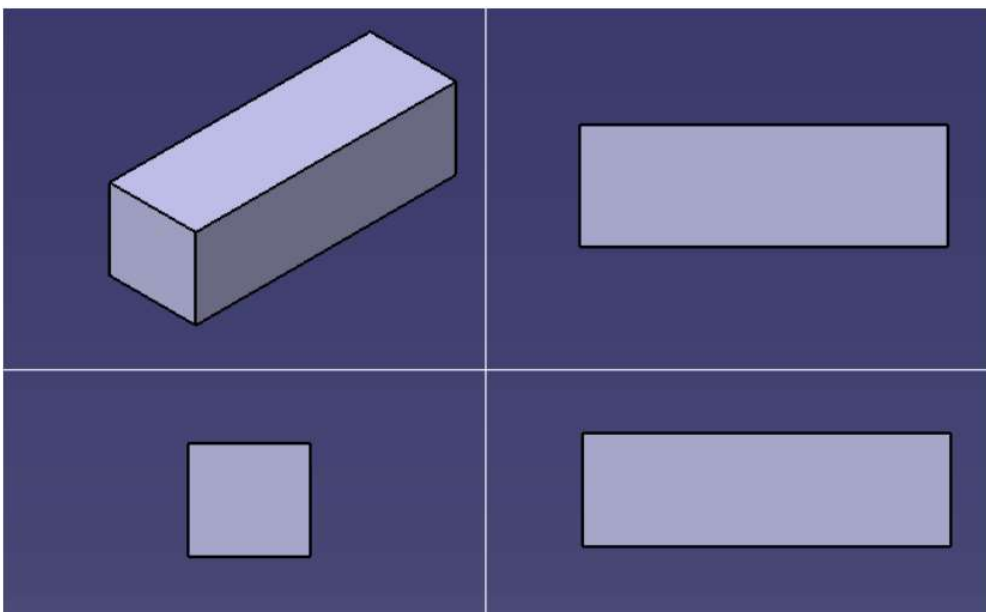


추력시험은 Load Cell을 이용하여 시험을 진행한다. Load Cell에 모터를 장착하기 위한 마운트를 3D모델링 할 예정이다.

ESC(변속기)는 XRotor Pro 40A를 선정하였다. OPTO 타입의 변속기로 BEC기능을 지원하지 않는다. OPTO 변속기의 경우 BEC기능을 하는 선이 접지선 역할을 하게 되는데, 이는 신호의 잡음을 제거하는 역할을 수행한다. 본 변속기는 Version A와 Version B가 존재한다. Version A는 소켓이 연장된 전선에 달린 형태이고, Version B는 소켓이 변속기에 직접 붙어있는 형태이며, 본 장치는 Version A이다. ESC는 드론 Arm Pipe의 지름보다 크기 때문에 Pipe 내부에 배치하는 대신 Pipe 하단에 배치하고 케이블타이를 이용해 고정한다.



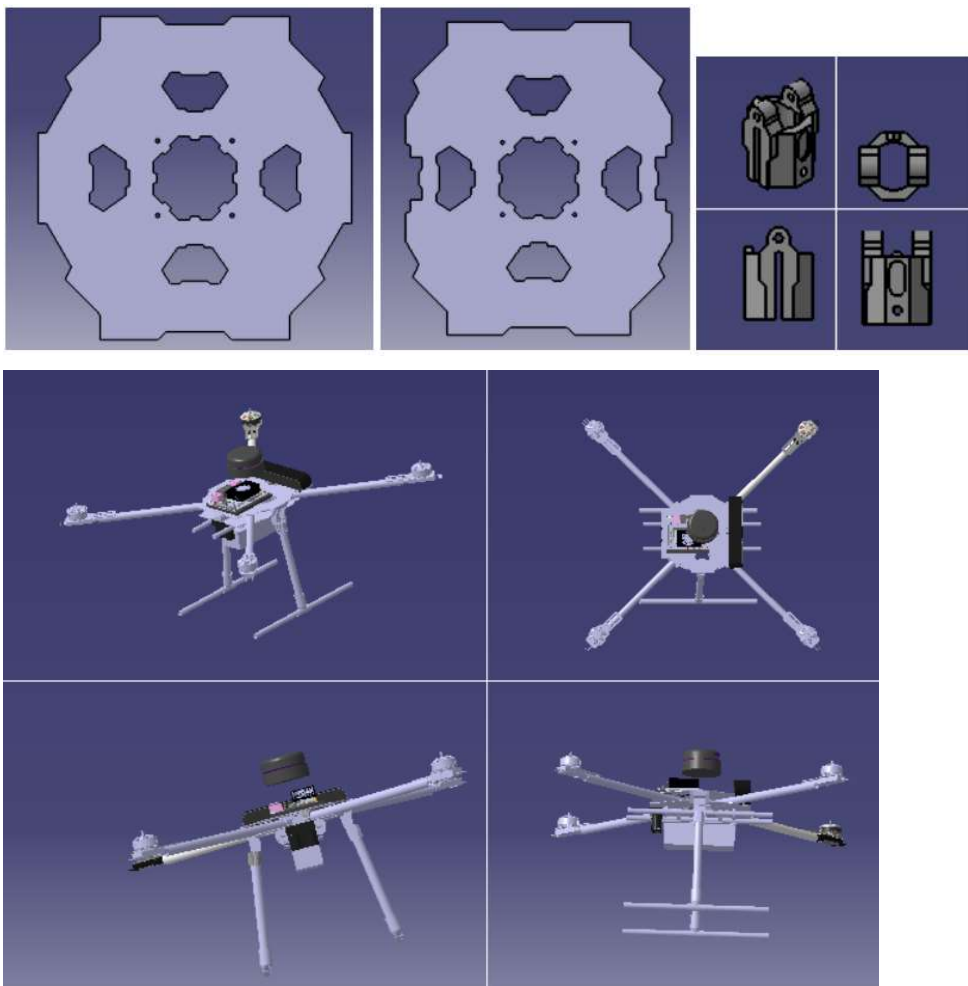
배터리는 Dinogy Graphene 14.8V 7200mAh 4s 80c로 선정하였다. 배터리는 드론의 무게중심을 선정하는데 가장 중요한 장비이다. 위치 조절이 가능한 장비 중 가장 무겁고, 가장 아래에 배치되므로 드론의 무게중심 및 안정성을 조절을 함께 담당한다. 배터리는 Specification에 따르면 136mm x 45mm x 42mm, 528g의 직육면체로 3D모델링 할 수 있다.



선정한 Frame은 Hub Plate, Chassis, Arm, Panel, 착륙장치로 나뉜다. Hub Plate는 Payload와 같은 장비들이 배치되는 공간으로 2D LiDAR 센서, 카메라 센서, Mini PC가 부착된다. 특히 2D LiDAR 센서를 Mini PC 위에 배치하기 위해 3D 프린팅을 이용한 마운트 제작이 필요하다. 제조사에서 제공하는 도면을 바탕으로 공차를 고려하여 나사구멍에 유의하여 제작해야한다.

Chassis는 Hub Plate 아래 층으로, Hub Plate와 함께 Arm을 고정한다. Chassis 와 Hub Plate 사이에는 Pixhawk Power Module 및 RX601이 배치되고, Chassis 밑에는 FC가 배치된다. Arm은 Arm Copter Attachment와 Arm Pipe, 모터마운트로 구성된다. ESC가 모터마운트 아래에 배치된다.

Panel은 배터리가 배치되는 Panel과 Panel과 Chassis를 연결해주는 Pendant 및 Pipe, Mount로 구성된다. Mount는 Pipe를 Chassis 밑에 고정시키고, Pipe는 Pendant를, Pendant에 Panel이 고정된다. 언급한 장치 중 착륙장치와 모터마운트의 3D 모델링이 계획되어 있다.



3.2.5 S/W 설계

0. 사전조건 : 사람과의 충돌이나 카메라로 정보 수집이 부적절한 곳은 차단하고 시작한다.(화장실가는길 막기, 계단 방화벽으로 막기)

1. 드론 출발지에서 주행 시작 : 드론에 자율비행 시작 신호를 보내고 1d 센서를 통해 높이 조절을 하며 호버링한다. 처음 위치에서 호버링을 한 후 2d LiDAR 센서를 통해 주변 데이터를 수집하며 SLAM 기술을 통해 실시간으로 지도를 작성한다. 실시간으로 작성된 지도를 통해서 깊이 우선 탐색법(DFS)를 적용하여 주행한다.

2. 영상정보를 통한 탐지 : 주행을 하는 도중 미리 학습된 객체를 발견하면 ZED카메라와 YOLOv5 기술을 이용하여 객체를 탐지하고 그 객체에 대한 정보를 저장한다.

3. 드론 출발지점으로 복귀 : 잔여 배터리가 절반이하로 남거나 DFS에 근거한 모든 경로가 다 탐색이 되었다고 판단하면 A* 알고리즘이나 왔던 경로를 그대로 되돌아가는 방법으로 출발지점으로 복귀한다.

4. 주행하는 동안 작성한 지도와 객체의 정보를 확인 : 출발지점으로 복귀를 성공한 드론은 Cartographer로 작성된 지도와 탐지한 객체의 위치정보를 하나로 통합한다. 천리안 팀원은 드론의 보드에서 원하는 데이터를 얻는다.

3.3. 수학적 분석

3.3.1. 센서 및 재료 분석

1) 모터

Voltage (V)	Prop	Throttle	Current (A)	Power (W)	Thrust (G)	RPM	Efficiency (G/W)	Operating Temperature (°C)
14.8	T-MOTOR 11*3.7CF	50%	3.1	45.88	380	5300	8.28	41
		65%	5.6	82.88	630	6500	7.60	
		75%	7.9	116.92	780	7200	6.67	
		85%	10.5	155.40	960	8000	6.18	
		100%	12.7	187.96	1110	8500	5.91	
	T-MOTOR 12*4CF	50%	3.8	56.24	460	4700	8.18	48
		65%	7.4	109.52	800	6300	7.30	
		75%	10	148.00	1000	6900	6.76	
		85%	13.5	199.80	1200	7500	6.01	
		100%	16.1	238.28	1360	8100	5.71	

MN3508의 Testing Data는 위와 같다.

14.8V의 배터리를 사용하는 경우 12인치의 프로펠러를 사용한다면 출력의 50~65% 사이에서 호버링 조건(1개당 추력 = 750g)을 만족함을 알 수 있다. 모터의 추력은 Testing Data와 오차가 크다. 외부 환경의 영향을 받으며, 실험에 사용된 모터가 본 과제에서 사용할 모터와 수치적으로 다르기 때문에 추력시험이 필요하다.

2) 변속기

변속기의 전류는 모터 최대전류의 1.2배 이상으로 선택한다. 모터의 최대 전류는 실험값 기준 16.1A로 최대전류는 19.32A이다. 변속기의 전류는 20A이상을 요구한다.

3) 배터리

모터의 전류 소모는 호버링 상태에 대한 전류소모를 바탕으로 선형보간법을 이용하여 예측했다.

$$(800g-460g)/(7.4A-3.8A)*(소모전류-3.8A)+460g=750g$$

$$\text{소모전류} = 6.87A$$

모터 4개의 소모전류는 27.48A이며, payload 및 FC의 전류 소모의 합은 4.525A이므로, 드론의 전류 소모를 32A로 예측했다.

$$\text{비행시간} = \text{배터리 용량} \times \text{배터리 방전} / \text{소모 전류} \times 60\text{min} / 1\text{h}$$

$$\text{배터리 방전} = 0.8(20\% \text{ 이하로 사용하는 경우, 배터리 고장 위험이 있음})$$

$$\text{비행시간} = 10.8\text{min}$$

예측한 비행시간은 10.8분으로 Requirements에서 요구하는 비행시간 10분을 만족한다.

3.3.2. 비행 동역학 및 제어

(1) 항공기의 운동특성을 해석하기 위해 동역학을 바탕으로 드론의 비선형 운동방정식을 유도한다.

(2) 비선형 운동방정식을 이용하여 선형 운동방정식을 유도하고, 선형 운동방정식을

이용하여 제어기 설계 및 해석을 위한 전달함수를 구한다.

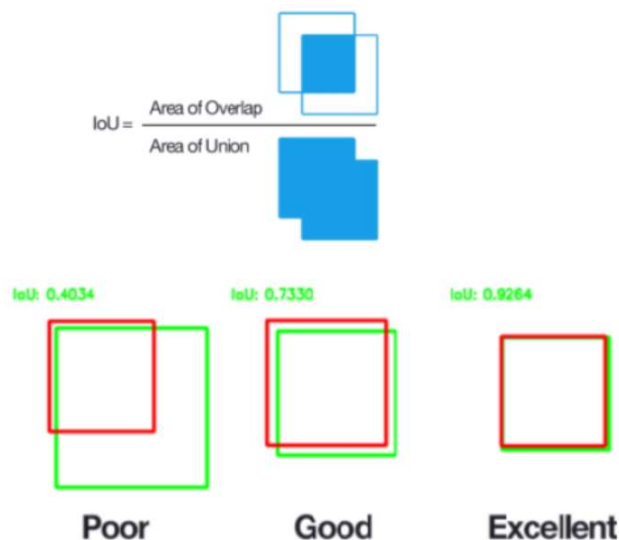
(3) 선형 운동방정식을 상태방정식 형태로 나타내고, 운동 해석 및 상태방정식을 이용한 전달함수를 구한다.

(4) 선형화된 드론 모델의 자동조정치와 유도 알고리즘을 설계한다.

3.3.3. 표적탐지 모델의 성능 평가

평가 지표를 확인 할 수 있는 툴인 Tensorboard를 사용하기 이전에 Tensorboard를 통해 얻을 수 있는 평가 지표들을 수식적인 접근으로 이해하며 분석하고 특히 precision, recall 지표를 동시에 반영하는 mAP, F1 Score를 평가한다.

1) IoU: 물체 탐지에서 Bounding Box를 얼마나 잘 예측했는지를 측정하는 수치이며 교집합을 합집합으로 나눈 값으로 정의 된다



2) confidence: 기준 신뢰도로서 경계박스 후보를 배경인지 물체인지를 판단하는 임계값으로 사용되며 임계값 이하의 경계박스후보는 배경으로 간주해서 제거 함으로 recall과는 반비례 관계이고 precision과는 비례관계이다.

3) precision (정밀도): 탐지된 객체의 정확도, 0.0 ~ 1.0 사이의 값을 가지며 높을수록 좋다.

$$Precision = \frac{TP}{TP + FP} = \frac{TP}{all\ detections}$$

4) recall (재현율): 전체 객체중에서 탐지한 비율, 0.0 ~ 1.0사이의 값을 가지며 높을수록 좋다.

$$Recall = \frac{TP}{TP + FN} = \frac{TP}{all\ ground\ truths}$$

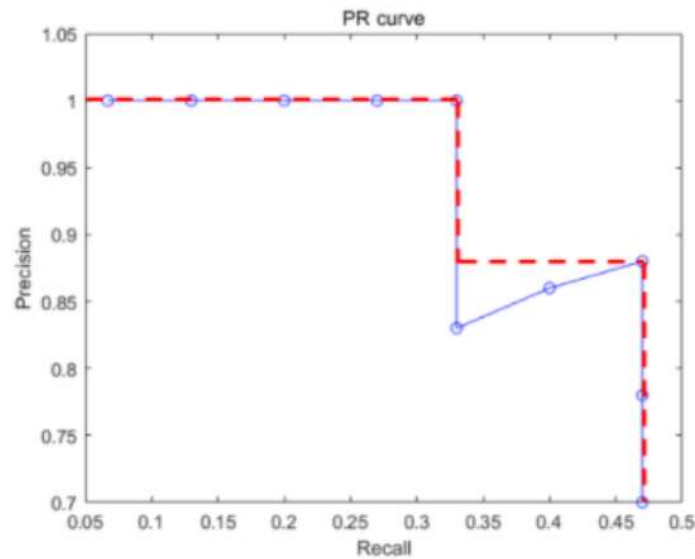
실제 상황 (ground truth)	예측 결과 (predict result)	
	Positive	Negative
Positive	TP(true positive) 옳은 검출	FN(false negative) 검출되어야 할 것이 검출되지 않았음
Negative	FP(false positive) 틀린 검출	TN(true negative) 검출되지 말아야 할 것이 검출되지 않았음

(TP, TN, FN, FP에 대한 정의)

5) F1Score: precision과 recall의 조화평균이며 0.0 ~ 1.0 사이의 값을 가지며 높을수록 좋으며 정밀도와 재현율이 둘다 높으면 F1Score의 값이 높기 때문에 하나의 값으로 2가지 정보를 유추할 수 있다.

$$F1Score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$$

6) AP (Average Precision): 변화하는 recall에 따라 대응되는 precision값을 그래프로 나타낸 RP그래프를 적분하여 얻은 값



7) mAP (mean Average Precision): 위의 AP를 클래스마다 평균한 값

8) FPS: frame / second

3.3.4. 자율비행 기술

본 과제에서 자율비행 기술은 z 축 이동을 무시한 2D 자율비행을 목표로 한다. 이는 로봇의 자율주행과 같은 구조로 생각할 수 있다. 대표적인 자율주행 문제로 미로찾기 알고리즘이 있다. 해당 조는 미로찾기 알고리즘 중 대표적인 우선법과 깊이 및 너비 우선 탐색 알고리즘을 공부했다. 우선법은 가장 간단하게 구현할 수 있는 알고리즘이지만, 미로의 형태와 시작 위치에 영향을 받아 미로에서 탈출하지 못할 위험이 존재하므로 적절하지 않다. 이런 우선법의 단점을 보완하기 위해 고안된 알고리즘이 너비 우선 탐색과 깊이 우선 탐색이다.

너비 우선 탐색과 깊이 우선 탐색은 그래프 및 트리에서 원하는 값을 찾기 위해 사용하는 알고리즘이지만 이미 탐색한 곳을 기억하고 탐색하는 특성이 있어 미로찾기 문제에서도 자주 사용되는 알고리즘이다. 너비 우선 탐색은 루트 주변을 먼저 탐색하고 멀리 떨어져 있는 지역을 탐색하는 알고리즘이고, 깊이 우선 탐색은 가장 깊은 지경까지 탐색한 후 가장 마지막 분기점에서 옆으로 이동하는 방식으로 탐색을 진행하는 알고리즘이다. 너비 우선 탐색은 루트 주변부를 먼저 탐색하는 특성 때문에 방과 방 사이를 이동해야 하는 현실의 주행에서는 비효율적인 이동 동선을 가질 수 있다. 이런 특성은 제한된 배터리를 이용해 효율적으로 탐색을 진행해야 하는 본 과제의 목표에 적합하지 않다. 그러므로 해당 조는 자율비행 알고리즘으로 깊이 우선 탐색을 채택하였다.

현재 자율주행 기술의 구현은 간단히 구현한 의사코드를 바탕으로 간단한 미로찾기 예시를 만들어 구현에 성공하였다.

```
while(still_in_maze):
```

```
    stack.push(nearby_places)           # 스택에 이동해야 할 곳을 저장한다.
```

```
    goto(stack.pop())                  # 다음 이동해야할 곳으로 이동한다.
```

*실제 코드 부록 참조

<깊이 우선 탐색의 의사코드>

ground_truth	map	odometry
[[1 1 1 1 1 1 1 1 1 1]]	[[1 1 1 1 1 1 1 1 1 1]]	[[0 0 0 0 0 0 0 0 0 0]]
[0 0 0 0 1 1 1 0 0 1]	[0 0 0 0 1 1 1 0 0 1]	[1 1 1 1 0 0 0 1 1 0]
[1 0 1 0 0 0 1 0 1 1]	[1 0 1 0 0 0 1 0 1 1]	[0 0 0 1 1 1 0 1 0 0]
[1 0 1 0 1 0 0 0 1 1]	[1 1 1 0 1 0 0 0 1 1]	[0 0 0 0 0 1 1 1 0 0]
[1 0 0 0 0 0 1 1 1 1]	[1 1 1 1 1 0 1 1 1 1]	[0 0 0 0 0 1 0 0 0 0]
[1 0 1 0 1 0 1 1 1 1]	[1 1 1 1 1 0 1 1 1 1]	[0 0 0 0 0 1 0 0 0 0]
[1 0 1 1 1 0 0 1 0 1]	[1 1 1 1 1 0 0 1 1 1]	[0 0 0 0 0 1 1 0 0 0]
[1 0 0 0 1 1 0 1 0 1]	[1 1 1 1 1 0 1 0 1 1]	[0 0 0 0 0 0 1 1 0 0]
[1 1 1 0 1 1 0 0 0 0]	[1 1 1 1 1 1 0 0 0 0]	[0 0 0 0 0 0 1 1 1 1]
[1 1 1 1 1 1 1 1 1 1]	[1 1 1 1 1 1 1 1 1 1]	[0 0 0 0 0 0 0 0 0 0]

<그림 1. 왼쪽부터 실제 미로, 제작된 지도, 이동경로>

그림 1의 배열은 각각 실제 미로, 깊이 우선 탐색을 통해 제작된 지도, 깊이 우선 탐색의 이동 경로를 의미한다. 실제 미로의 좌측 상단을 입구로 하여 우측 하단의 출구로 잘 탈출한 모습이다. 하지만 본 과제의 드론은 실내에서 건물의 구석구석 이동하며 건물 전체의 지도를 그리는 것이 목표이므로 입구와 출구를 벽으로 막아 진행해 보았다.

ground_truth	map	odometry
[[1 1 1 1 1 1 1 1 1 1]]	[[1 1 1 1 1 1 1 1 1 1]]	[[0 0 0 0 0 0 0 0 0 0]]
[1 0 0 0 1 1 1 0 0 1]	[1 0 0 0 1 1 1 0 0 1]	[0 1 1 1 0 0 0 1 1 0]
[1 0 1 0 0 0 1 0 1 1]	[1 0 1 0 0 0 1 0 1 1]	[0 1 0 1 1 1 0 1 0 0]
[1 0 1 0 1 0 0 0 1 1]	[1 0 1 0 1 0 0 0 1 1]	[0 1 0 1 0 1 1 1 0 0]
[1 0 0 0 0 0 1 1 1 1]	[1 0 0 0 0 0 1 1 1 1]	[0 1 1 1 1 1 0 0 0 0]
[1 0 1 0 1 0 1 1 1 1]	[1 0 1 0 1 0 1 1 1 1]	[0 1 0 1 0 1 0 0 0 0]
[1 0 1 1 1 0 0 1 0 1]	[1 0 1 1 1 0 0 1 0 1]	[0 1 0 0 0 1 1 0 1 0]
[1 0 0 0 1 1 0 1 0 1]	[1 0 0 0 1 1 0 1 0 1]	[0 1 1 1 0 0 1 0 1 0]
[1 1 1 0 1 1 0 0 0 1]	[1 1 1 0 1 1 0 0 0 1]	[0 0 0 1 0 0 1 1 1 0]
[1 1 1 1 1 1 1 1 1 1]	[1 1 1 1 1 1 1 1 1 1]	[0 0 0 0 0 0 0 0 0 0]

<그림 2. 왼쪽부터 실제 건물, 제작된 지도, 이동경로>

그림 2의 배열은 각각 실제 건물의 구조, 깊이 우선 탐색을 통해 제작된 지도, 깊이 우선 탐색의 이동 경로를 의미한다. 이동 가능한 모든 이동 경로를 탐색하여 실제 건물과 같은 지도를 작성했음을 알 수 있다. 남은 과제는 크게 두가지로 음수 좌표의 표현과 실제 드론에의 적용을 꼽을 수 있다.

먼저, 실제 드론의 시작 위치를 원점 (0, 0)이라 가정하면 탐색의 시작 위치에 따라 음수의 방향으로 이동해야하는 가능성을 배제할 수 없다. 하지만 배열에서 음수 좌

표는 표시할 수 없다. 이를 가장 간단히 해결할 방법은 원점의 위치를 표기할 배열을 이용해 해결할 수 있을것으로 생각된다.

3.3.5. 지형탐색 기술

SLAM이란 Simultaneous Localization and Mapping의 약자로 동시적 위치 추정 및 지도 작성이라는 뜻이다. 즉, 로봇이 미지의 환경을 탐색하면서 로봇에 장착된 센서만으로 로봇 스스로 자신의 위치를 추정하는 것과 동시에 미지 환경의 지도를 작성하는 것을 의미한다.

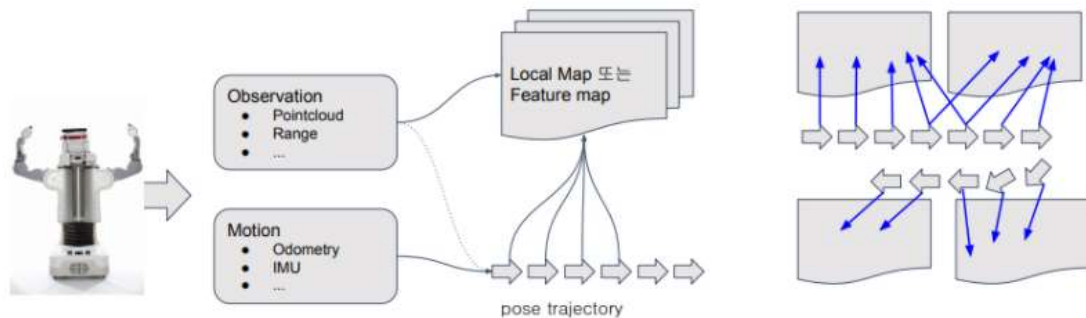
SLAM은 1990년 Extended kalman filter의 사용으로 처음 제안이 되었다. 그 이후 실내 실외 환경에 맞춰서 GPS, IMU, V2X 같은 센서들과 딥러닝을 이용하여 더욱 정확한 SLAM을 정밀한 위치 추정 및 HD맵을 만들 수가 있었고 초반에 filter-based SLAM에서 2010년 이후 Graph-based SLAM가 다수를 차지하였다.

지도 작성에 사용되는 센서로는 거리 센서가 많이 사용되는데 이는 초음파센서, 광선 감지기(LiDAR), 전파탐지기(radar), 레이저 레인지 파인더(LRF, Laser Range Finder), 적외선 스캐너 등이 많이 사용된다. 거리 센서 이외에도 카메라를 이용하기도 하는데 스테레오 카메라를 이용한 거리 측정으로 거리 센서 처럼 이용되거나, 일반 카메라를 이용한 Visual SLAM도 있다. 그리고 환경에 마커를 붙여서 인식하는 방식도 제안하고 있다. 예를들어, 천장에 마커를 장착하여 카메라로 이를 마커를 구별하는 등의 마커 기반의 방법이 있다.

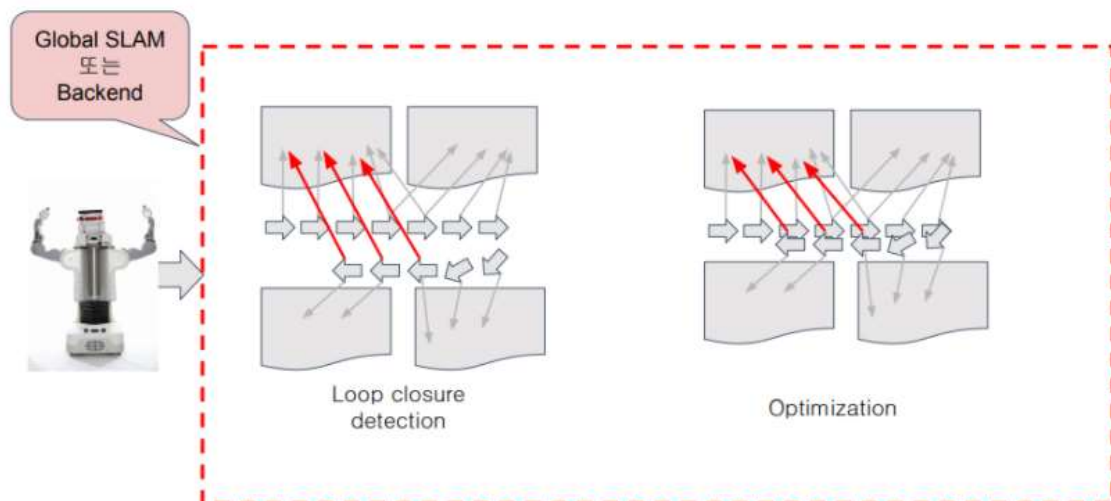
그중에서 Visual SLAM 방식은 보통 IMU와 쌍을 이루는 카메라를 사용하여 탐색 경로를 매핑하는 것이다. IMU도 사용되는 경우, 이를 Visual-Inertial Odometry 또는 VIO라고 한다. Odometry는 모션 센서 데이터를 사용하여 시간에 따른 로봇의 위치 변화를 추정하는 것을 말한다. 일반적으로 Visual SLAM에서 설정점 (알고리즘에 의해 결정된 관심 지점)은 연속적인 카메라 프레임을 통해 추적되어 특징점 삼각 측량이라고하는 3D 위치를 삼각 측량한다. 이 정보는 3D 맵을 생성하고 로봇의 위치를 식별하기 위해 다시 전달된다.

LiDAR 기반 SLAM 방식은 IMU와 쌍을 이루는 레이저 센서를 사용하여 Visual SLAM과 유사하게 방을 매핑하지만 한 차원에서 더 높은 정확도를 제공한다. LiDAR는 여러 송수신기로 물체를 비추어 물체 (예 : 벽 또는 의자 다리)까지의 거리를 측정하는 것이다. 각 송수신기는 빠르게 펄스 광을 방출하고 반사 된 펄스를 측정하여 위치와 거리를 결정한다

LiDAR SLAM의 한 종류인 Cartographer는 구글에서 개발하여 오픈소스로 공개한 Graph-Based SLAM이다. 로봇을 통해서 얻는 데이터는 Observation과 Motion으로 두가지 타입의 데이터로 나뉘는데. 이렇게 얻어진 두가지 타입의 데이터는 다시 pose trajectory와 Local Map으로 변환이 된다.

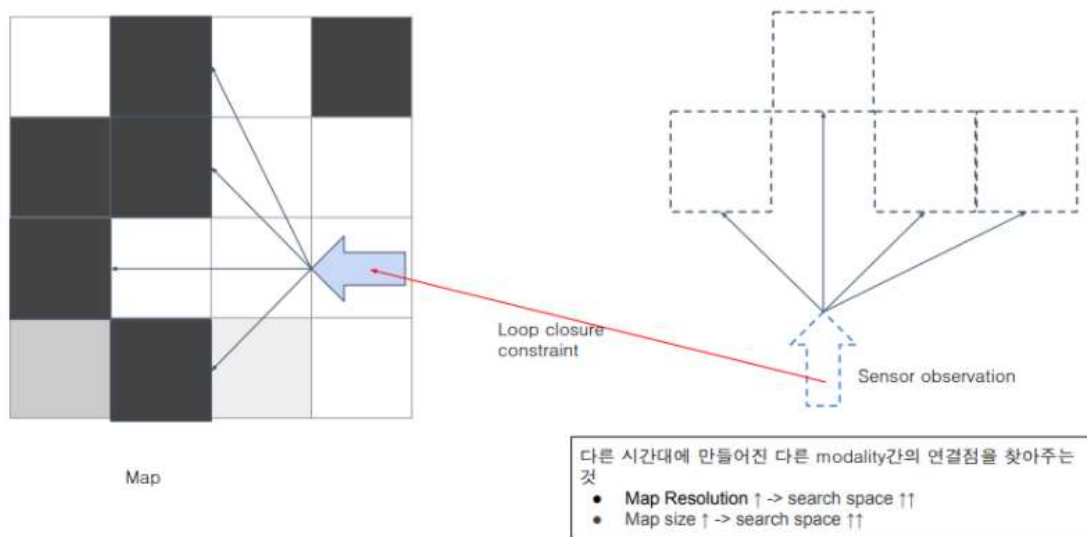


Pose trajectory는 로봇이 현재 움직여온 정보에 대한 데이터를 말하며, Local Map은 Observation 데이터와 Pose trajectory를 합쳐서 만든 데이터로 움직이면서 만들어진 센서 값들과 관찰한 데이터를 모두 모아놓은 것이다.

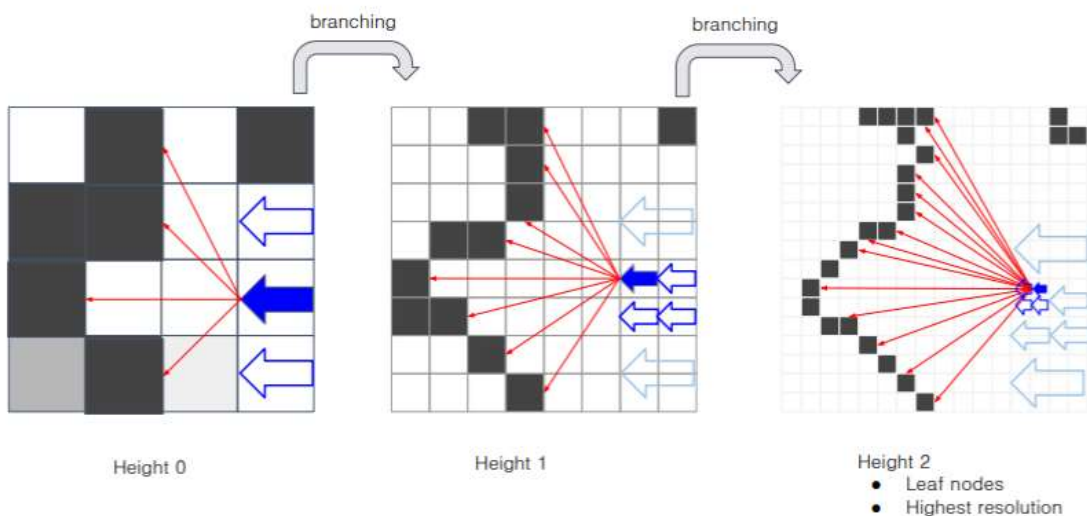


위에서 만들어진 정리가 안된 "지도 조각"들을 완성된 지도로 만들어 주기 위해서는 Loop closure detection 과 Optimization(최적화) 과정을 거쳐야 한다.

루프 폐쇄 검출(loop closure detection)이란 로봇의 이동 궤적상에서 현재의 위치가 이전에 방문했던 위치인지를 판단하는 것으로 검출된 결과를 환경 맵 최적화 단계에서 제약조건으로 활용하도록 하는 것이다.



이러한 과정에서 이전에 방문했던 위치와 일치하는 곳을 찾으면 Branch-and-Bound 기법을 통해 지도를 더욱 상세하게 만든다.



Branch-and-Bound 기법은 처음에는 Low Resolution Level에서 검색을 하고, 일치하면 Level을 높여가며 더욱 상세하게 맞춰가는 작업을 한다는 개념이다. Cartographer에서는 DFS branch and bound(BBS) 기법을 이용해서 작업 시간을 더욱 빠르게 만든다.

Algorithm 3 DFS branch and bound scan matcher for (BBS)

$best_score \leftarrow score_threshold$

Compute and memorize a score for each element in \mathcal{C}_0 .

Initialize a stack \mathcal{C} with \mathcal{C}_0 sorted by score, the maximum score at the top.

while \mathcal{C} is not empty **do**

 Pop c from the stack \mathcal{C} .

if $score(c) > best_score$ **then**

if c is a leaf node **then**

$match \leftarrow \xi_c$

$best_score \leftarrow score(c)$

else

 Branch: Split c into nodes \mathcal{C}_c .

 Compute and memorize a score for each element in \mathcal{C}_c .

 Push \mathcal{C}_c onto the stack \mathcal{C} , sorted by score, the maximum score last.

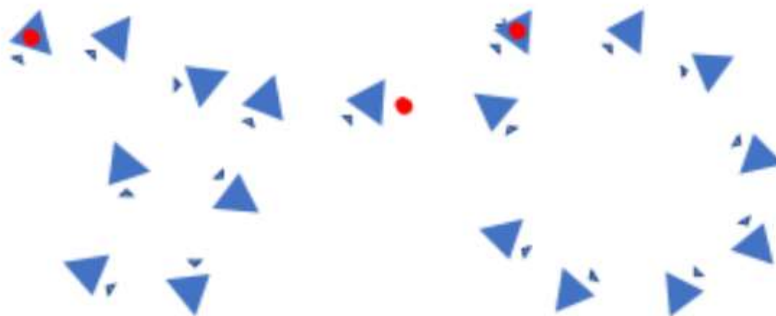
end if

end if

end while

return $best_score$ and $match$ when set.

만약 이러한 Loop closure detection이 없다면 누적된 궤적 오차 때문에 예측된 로봇의 궤적이 심하게 뒤틀리는 것을 확인 할 수 있다. - 그림 (a),(b) -



(a) 루프 폐쇄 검출을 사용하지 않은 경우

(b) 루프 폐쇄 검출을 사용한 경우

아래의 도표에서는 뮌헨의 도이치 박물관에서 실험을 한 데이터이다.

- 약 30분, 2.2km의 trajectory 분량의 데이터
- 6분 이내의 최적화된 맵 생성 가능
- 실시간보다 약 5.3배 빠르다고 주장
- Intel Xeon E5-1650 (3.2GHz, 2.2G mem)

TABLE II QUANTITATIVE COMPARISON OF ERROR WITH [21]			TABLE III QUANTITATIVE COMPARISON OF ERROR WITH [9]		
	Cartographer	GM		Cartographer	Graph FLIRT
Aces			Intel		
Absolute translational	0.0375 ± 0.0426	0.044 ± 0.044	Absolute translational	0.0229 ± 0.0239	0.02 ± 0.02
Squared translational	0.0032 ± 0.0285	0.004 ± 0.009	Absolute rotational	0.453 ± 1.335	0.3 ± 0.3
Absolute rotational	0.373 ± 0.469	0.4 ± 0.4	Freiburg bldg 79		
Squared rotational	0.359 ± 3.696	0.3 ± 0.8	Absolute translational	0.0452 ± 0.0354	0.06 ± 0.09
Intel			Absolute rotational	0.538 ± 0.718	0.8 ± 1.1
Absolute translational	0.0229 ± 0.0239	0.031 ± 0.026	Freiburg hospital (local)		
Squared translational	0.0011 ± 0.0040	0.002 ± 0.004	Absolute translational	0.1078 ± 0.1943	0.18 ± 0.27
Absolute rotational	0.453 ± 1.335	1.3 ± 4.7	Absolute rotational	0.747 ± 2.047	0.9 ± 2.0
Squared rotational	1.986 ± 23.988	24.0 ± 166.1	Freiburg hospital (global)		
MIT Killian Court			Absolute translational	5.2242 ± 6.6230	8.3 ± 8.6
Absolute translational	0.0395 ± 0.0488	0.050 ± 0.056	Absolute rotational	3.341 ± 4.797	5.0 ± 5.3
Squared translational	0.0039 ± 0.0144	0.006 ± 0.029	TABLE IV LOOP CLOSURE PRECISION		
Absolute rotational	0.352 ± 0.353	0.5 ± 0.5	Test case	No. of constraints	Precision
Squared rotational	0.248 ± 0.610	0.9 ± 0.9	Aces	971	98.1 %
MIT CSAIL			Intel	5786	97.2 %
Absolute translational	0.0319 ± 0.0363	0.004 ± 0.009	MIT Killian Court	916	93.4 %
Squared translational	0.0023 ± 0.0099	0.0001 ± 0.0005	MIT CSAIL	1857	94.1 %
Absolute rotational	0.369 ± 0.365	0.05 ± 0.08	Freiburg bldg 79	412	99.8 %
Squared rotational	0.270 ± 0.637	0.01 ± 0.04	Freiburg hospital	554	77.3 %
Freiburg bldg 79			TABLE V PERFORMANCE		
Absolute translational	0.0452 ± 0.0354	0.056 ± 0.042	Test case	Data duration (s)	Wall clock (s)
Squared translational	0.0033 ± 0.0055	0.005 ± 0.011	Aces	1366	41
Absolute rotational	0.538 ± 0.718	0.6 ± 0.6	Intel	2691	179
Squared rotational	0.804 ± 3.627	0.7 ± 1.7	MIT Killian Court	7678	190
Freiburg hospital (local)			MIT CSAIL	424	35
Absolute translational	0.1078 ± 0.1943	0.143 ± 0.180	Freiburg bldg 79	1061	62
Squared translational	0.0494 ± 0.2831	0.053 ± 0.272	Freiburg hospital	4820	10
Absolute rotational	0.747 ± 2.047	0.9 ± 2.2			
Squared rotational	4.745 ± 40.081	5.5 ± 46.2			
Freiburg hospital (global)					
Absolute translational	5.2242 ± 6.6230	11.6 ± 11.9			
Squared translational	71.0288 ± 267.7715	276.1 ± 516.5			
Absolute rotational	3.341 ± 4.797	6.3 ± 6.2			
Squared rotational	34.107 ± 127.227	77.2 ± 154.8			

3.4. 컴퓨터 해석

3.4.1. 표적탐지 기술

표적탐지 모델을 학습시키고 이를 평가하는 방법은 Precision, Recall, F1 Score,

AP, mAP등 다양한 수치들을 비교하는 것이다. 이를 편하게 해주는 도구중 하나인 TensorBoard는 실시간으로 학습 과정을 그래프로 확인 할 수 있고 기존에 학습했던 모델과의 동시 비교 분석을 시각화 해주기 때문에 이를 사용하여 표적탐지 모델의 성능을 평가 하거나 재학습 시킬 계획이다.

3.4.2 가상환경 구축(Gazebo)

본 과제에서는 수많은 센서들을 결합하여 드론을 설계 하였고 level 2 ~ 3 수준의 자율비행을 목표로 하고 있기 때문에 가상환경을 통한 시뮬레이션은 장비 손실과 안전상의 문제를 해결하기 위해 필수적이다. 본 과제에서는 드론제어 안정성 평가, 자율주행 알고리즘 평가를 수행해야 한다. 처음에는 Gazebo 와 QGC (Ground Control System)를 이용하여 순수 소프트웨어로만 구성된 SITL을 수행한다. 이후에 픽스호크와 결합하여 FC 하드웨어를 사용한 HITL을 수행하여 최종적으로 실제 비행을 거쳐서 비행의 안전성 및 자율 비행 알고리즘에 대한 평가를 할 계획이다.

3.5. 제작 및 실험평가 중간 결과

3.5.1. 3D 프린팅

추력시험을 위한 구조물 및 2D LiDAR 마운트를 모델링 파일을 바탕으로 3D 프린팅을 이용해 출력한다. 교내 공장동에서 3D 프린팅을 진행하며, 프린팅에 필요한 필라멘트의 경우 출력 이후 공장동 요구사항에 맞추어 구매/전달한다. 필요에 따라 레이저 커팅으로 대체하여 진행한다.

3.5.2. 추력시험

추력시험 장치를 제작하여 모터의 추력시험을 진행한다. 스톱스에 따른 모터의 추력 및 소모 전류를 측정하여 기존에 계산한 수식과 비교한다.

3.5.3. 프로토타입 제작 계획

- 1) TX2 보드에 ubuntu 18.04, ros melodic, QGC(GCS) 및 구현된 기술에 필요한 환경을 구성한다.
- 2) 현재 구현된 SLAM(cartographer)기술, 표적탐지 기술(yolo v5) 등을 TX2 보드에 탑재한다.
- 3) 자율 비행 알고리즘을 SITL, HITL 검증 기법으로 검증하고 QGC 미션파일 자동화 업로드 기법을 조사하여 적용한다.

4) 시스템 통합

- 자율비행 시스템에서는 LiDAR 센서에서 반환되는 데이터를 사용해 미션 파일을 생성한다. 미션파일 자동화 업로드 기법을 통해 GCS에 미션파일을 업로드하고 픽스 호크에게 명령을 내려 드론을 비행시킨다.
- 표적탐지 시스템에서는 ZED 카메라 센서에서 반환되는 데이터를 가지고 표적의 클래스, 깊이, timestamp 데이터를 저장한다.
- SLAM 시스템에서는 LiDAR 센서에서 반환되는 데이터를 사용하여 2D지도 데이터를 저장한다.

최종적으로 ros의 takeoff 신호를 받아 각 시스템이 실행되도록 자동화 시스템을 구현한다.

3.5.4. 프로토타입 데모시나리오 및 평가 계획

- 데모시나리오
 - 사전 조건: 화장실, 계단, 유리창 등을 막아 시나리오에 방해요소 없앤다.
 - 1) 501호 뒷 문앞에서 출발한다.
 - 2) 드론은 head 방향으로 움직이기 시작함과 동시에 지도작성 및 영상정보를 얻는다.
 - 3) 드론은 깊이 우선 탐색법에 의해 자율주행을 시작한다.
 - 4) 드론은 506호에서 장애물에 대한 영상정보를 얻는다.
 - 5) 모든 탐색을 마친후 시작점으로 돌아오며 영상정보에서 탐지한 클래스 정보를 작성한 지도위에 출력한다.
 - 6) 팀원은 출발지점으로 돌아온 드론의 보드에서 완성된 지도를 얻는다.
- output: 통합지도(표적에 대한 클래스정보가 포함된 2D지도)

4. 프로젝트 팀 구성 및 역할 분담

팀원	역할
전병륜(팀장)	드론 HW 설계, 로터 블레이드 및 드론 공력성능 계산
김강산	표적탐지기술 구현 및 가상환경 구축
김송섭	지형탐색 기술 구현 및 통합
박보근	드론 동역학 모델링 및 제어기 모델링
전동환	드론 길찾기 및 자율비행 기술 구현

5. 과제 예산 및 내역

- 예산: 1,050,000W
- 예산 내역

항목	비용(원)
기체프레임	150,000
Flight Controller	500,000
Propeller	100,000
3D 프린팅용 필라멘트	100,000
회의비	100,000
기타 소모품	100,000
총합	1,050,000

※예산 편성은 항목에 따라 1.5배를 한 것으로, 여유롭게 책정함

6. 과제 추진 일정

항목	3월				4월				5월				6월
주	1	2	3	4	1	2	3	4	1	2	3	4	1
설계													
	-드론 HW 설계 -SW 아키텍처 설계												
수학적 분석													
	-드론 specification 작성 -기자재 주문												
시뮬레 이션													
	-구조해석 -표적탐지기술 구현 -지도생성기술 구현 -자율비행기술 구현 -가상환경 구축 및 시뮬레이션												
실험 및 검 증													
	-프로토타입 제작 및 시험 -시험결과 평가												
보고서 작성 및 발 표													
	-영상자료 촬영 및 중간발표 -보고서 발표												

7. 참고 문헌

연번	종류	제목	저자	발행년도	발행자
1	블로그	물체 검출 알고리즘 성능 평가방법 AP(Average Precision)의 이해 (https://bskyvision.com/465)	-	2019	-
2	논문	Object Detection in 20 Years: A Survey	Zhengxia Zou et al.	2020	IEEE
3	웹사이트	자율주행네트워크(https://www.ciena.kr/insight/what-is/What-Is-an-Autonomous-Net-work_ko_KR.html)	-	-	-
4	뉴스기사	자율주행차와 자율주행의 차이(https://www.motorgraph.com/news/articleView.html?idxno=24498)	박홍준	2019	Motor Graph
5	블로그	자율주행 6단계 (https://styleandeasy.com/2019/03/14/tech-talk-untangling-the-5-levels-of-drone-autonomy/)	-	2019	-
6	블로그	미로 알고리즘 우선법 : (https://gusdnd852.tistory.com/7)	고현웅	2020	-
7	블로그	DFS, BFS 의 개념과 구현 (https://jeinalog.tistory.com/18)	정민정	2019	-
8	블로그	깊이 우선 탐색과 너비 우선 탐색(https://devuna.tistory.com/32)	-	2020	-
9	블로그	깊이 우선 탐색 (https://starkyng.tistory.com/entry/DFSDepth-First-Search-%E4%B9%A4EC%ED%84%EC%8A%B0%E%84%A0ED%83%80EC%83%89)	-	2019	-
10	블로그	길찾기 bfs 알고리즘 (https://jackpot53.tistory.com/123)	-	2019	-
11	논문	Loop Closure Detection Using Variational Autoencoder in Simultaneous Localization and Mapping (https://www.koreascience.or.kr/article/CFKO201736257096995.page)	신동원, 호요성	2017	koreaS cience
12	논문	Real-Time Loop Closure in 2D LIDAR SLAM (https://static.googleusercontent.com/media/research.google.com/ko//pubs/archive/45466.pdf)	Wolfgang Hess, et al	2016	IEEE
13	오픈소스	Cartographer 오픈소스 (https://github.com/cartographer-project/cartographer_ros/)	Michael Grupp	2017	Github

14	웹사이트	Running Cartographer ROS on a demo bag (https://google-cartographer-ros.readthedocs.io/en/latest/demos.html)	-	2021	Google
15	논문	A Survey of Simultaneous Localization and Mapping with an Envision in 6G Wireless Networks	Baichuan Huang et al.	2020	arXiv

8. 부록

부록 A. 소스코드

A.1. Stack.py (스택 기능 구현)

```
class Stack:
    def __init__(self):
        self.top=[]
    def __len__(self):
        return len(self.top)
    def push(self,item):
        self.top.append(item)
    def pop(self):
        if not self.isEmpty():
            return self.top.pop(-1)
        else:
            print("Stack underflow")
            exit()
    def clear(self):
        self.top=[]
    def isEmpty(self):
        return len(self.top) == 0
```

A.2. Maze.py (미로 생성 및 벽 정보 전달)

```
import numpy as np
class Maze:
    def __init__(self):
        self.maze = []
        self.startX = 0
        self.startY = 0
```

```

        self.setDefault()
# 기본지도
def setDefault(self):
    self.maze = [[1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
                  [1, 0, 0, 0, 1, 1, 1, 0, 0, 1],
                  [1, 0, 1, 0, 0, 0, 1, 0, 1, 1],
                  [1, 0, 1, 0, 1, 0, 0, 0, 1, 1],
                  [1, 0, 0, 0, 0, 0, 1, 1, 1, 1],
                  [1, 0, 1, 0, 1, 0, 1, 1, 1, 1],
                  [1, 0, 1, 1, 1, 0, 0, 1, 0, 1],
                  [1, 0, 0, 0, 1, 1, 0, 1, 0, 1],
                  [1, 1, 1, 0, 1, 1, 0, 0, 0, 1],
                  [1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]
    self.maze = np.array(self.maze, dtype = np.int8)
    self.startX = 1
    self.startY = 1
def setPosition(self):
    return self.startX, self.startY
# SLAM; 주변 지역 벽을 탐지해 반환
# {좌표 : 벽}
def getWall(self, x, y):
    dic = {}
    for i in (-1, 1):
        if(0 <= x+i and x+i < self.maze.shape[0]):
            dic[ (x+i, y) ] = self.maze[x+i, y]
    for j in (-1,1):
        if(0 <= y+j and y+j < self.maze.shape[1]):
            dic[ (x, y+j) ] = self.maze[x, y+j]
    return dic

```

A.3. DFS.py (깊이 우선 탐색을 이용한 지도 작성)

```

import numpy as np
from Stack import Stack
from Maze import Maze
# 초기화
stack = Stack()
maze = Maze() # 실제 지도
map_ = np.ones((2, 2), dtype=np.int8) # 드론이 그리는 지도
odom = np.zeros((2, 2), dtype=np.int8) # 드론의 이동 경로
dic = {}
x, y = maze.setPosition() # 초기 위치
odom[x, y] = 1
map_[x, y] = 0

```

```

while True:
    dic = maze.getWall(x,y) # SLAM; 주변의 벽을 탐지(4방향)
    for i in dic.keys():
        # 현재 가지고 있는 지도밖의 벽 혹은 길이 나왔을 때
        # 지도의 크기를 키운다.
        if(map_.shape[0] <= i[0]):
            map_2 = np.ones((i[0] + 1, map_.shape[1]), dtype=np.int8)
            odom2 = np.zeros((i[0] + 1, map_.shape[1]), dtype=np.int8)
            map_2[:map_.shape[0], :map_.shape[1]] = map_
            odom2[:map_.shape[0], :map_.shape[1]] = odom
            map_ = map_2
            odom = odom2
        if(map_.shape[1] <= i[1]):
            map_2 = np.ones((map_.shape[0], i[1] + 1), dtype=np.int8)
            odom2 = np.zeros((map_.shape[0], i[1] + 1), dtype=np.int8)
            map_2[:map_.shape[0], :map_.shape[1]] = map_
            odom2[:map_.shape[0], :map_.shape[1]] = odom
            map_ = map_2
            odom = odom2
        # SLAM에서 얻은 데이터에서 길을 지도에 표시 후 스택에 저장
        if dic[i] == 0 and odom[i] != 1:
            map_[i] = dic[i]
            stack.push(i)
        # 갈 수 있는 모든 길을 이동했을 때 알고리즘 종료
        if stack.isEmpty():
            break
    x, y = stack.pop()
    odom[x, y] = 1 # x,y로 이동한 것 기록
    # 미로에서 탈출 했을 때 알고리즘 종료
    if y == maze.maze.shape[1] - 1:
        break
# 결과 출력
print("odometry\n", odom)
print("map\n", map_)
print("position : (" + str(x) + ', ' + str(y) + ')')
print('ground_truth\n', maze.maze)
print('sucess')

```