제18회 임베디드SW경진대회 개발완료보고서

[자유공모]

□ 개발 요약

팀 명	플라이미투더문

Figure 1. 800급 메인 헥사콥터, 450급 테스트용 기체

Figure 2. 30m 상공에서 일정거리를 유지하며 사람 추적

작품명	돌리자베스
작품설명 (요약)	30m 상공에서 자율비행 드론이 탐색한 목표물을 일정거리 유지 추적, 목표물 촬영 영상 실시간 송신을 통한 지상 관제가 가능한 무인항공기 시스템
소스코드	자율비행 시스템: https://github.com/DWKE/FlyMeToTheHome.git 무인항공기 제어: https://github.com/DWKE/Firmware.git
시연동영상	https://www.youtube.com/watch?v=pcV_RXZqN2M&ab_channel=%EC%8B%A0%EC%88%98%EC%97%B0

□ 개발 개요

○ 개발 작품 개요

- 지상국과 통신하며 목표물 탐색, 추적 및 실시간 지상관제 영상전송이 가능한 무인항공기 시스템

○ 개발 목표

- 30m 상공에서 드론이 자율적으로 목표물(사람)을 인식한 후 해당 목표물과 일정거리를 유지함과 동시에 비행금지구역을 우회하며 추적한다. 동시에 관제를 위해 목표물 촬영 영상을 LTE 상용망을 통해 지상으로 실시간 전송한다.
  - 자율 비행: 드론이 초기 위치까지의 자동비행 이후 목표물을 자율적으로 설정하고 제약사항(비행금지구역)을 고려하며 자율 비행한다.
  - 목표물 인식 및 추적: 부착된 카메라로 사람을 인식하고, 물체 위치를 위도 경도 기반의 global 좌표로 변환한 후 추적한다.
  - 지상과의 실시간 통신: ROS 라이브러리와 LTE 상용망을 활용하여 실시간으로 지상에 영상과 비행데이터를 송신하고 지상에서 영상을 관제한다.

○ 개발 작품의 필요성

- **추적 비행드론의 활용**: 드론은 차량으로 꽉 막힌 도심 도로상황에서 하늘길을 이용하여 범죄차량 등의 특정 목표물을 제약없이 빠르게 추적이 가능하다. 이때, 지정된 법적규제구역인 비행금지구역을 회피할 수 있도록 한다. 지상국으로 현재 추적 대상 목표물 촬영 영상을 LTE 이동통신망을 통해 실시간 전송한다. 이는, 안심 귀가 동행 서비스에 활용되어 보호자가 추적 대상의 상황을 실시간으로 확인할 수 있도록 활용할 수 있다.
- **IoT 장비로써의 드론**: 드론이 상용망을 통해 비행경로상의 다양한 장비, 통신 허브와 연결되어 상호간의 정보를 교환하고 전달하는 하나의 IoT 기기로써 작동한다. 드론에서 수집중인 환경 데이터나 비행 데이터, 명령을 실시간으로 송수신함으로써 기존 자율시스템과 연계되어 많은 서비스를 제공할 수 있으며 안전한 운행이 가능하다.
- **오픈소스 기반의 자율비행 기술 발전 도모**: 기존 항공업계의 In-House 코드 중심의 폐쇄적이던 개발 생태계를 탈피하고자 하였다. 제어부터 미션수행까지 자율 시스템 환경을 구성하는데 있어 오픈소스를 최대한 활용함으로써 시스템이 무한확장 가능할 수 있도록 하였다. 이로써, 지상에서 자율주행차 등으로 구현되고 있는 다양한 서비스, 주행 기술을 빠른 속도로 하늘길에 적용할 수 있게 되는 등 무궁무진한 협력개발이 가능해진다.

□ 개발 환경 설명

○ Hardware 구성

메인기체로는 충분한 페이로드를 수용하며 긴 시간동안 비행이 가능하도록 800급 헥사콥터를 설계 및 제작하였다. 하지만 비행 여건 및 안전상의 문제로 간단한 자동비행 등의 알고리즘 테스트 시에는 450급 쿼드콥터를 서브기체로 사용하였다. 아래 <Figure 1>은 두 기체를 나타낸 사진이다.

Figure 1 (좌) 800급 메인기체/ (우) 450급 서브기체

자율 비행 경로 생성을 담당하는 싱글보드 컴퓨터를 중심으로 환경인지 센서를 부착하였고, 명령에 따라 기체를 제어하는 FCU를 중심으로 드론을 운항하게 하는 기자재를 연결하여 시스템 인터페이스를 구성하였다.

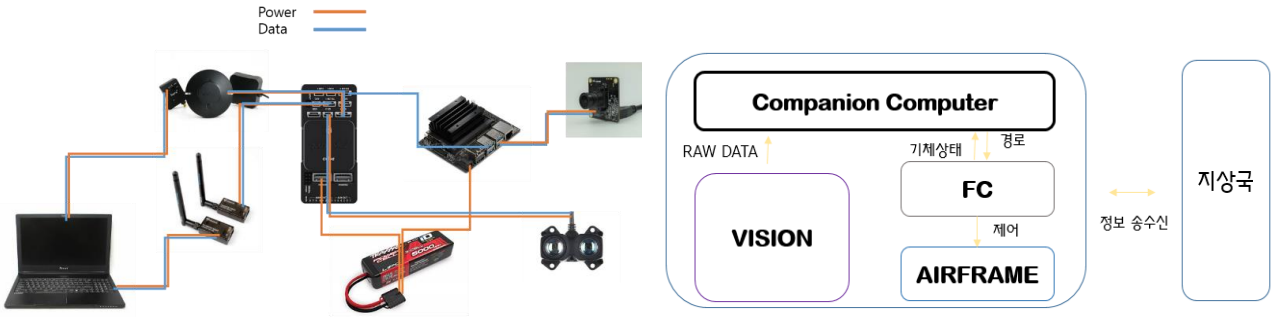

Figure 2 H/W Architecture Interface

○ Hardware 기능 (제어 방법 등 서술)

Companion Computer에 연결된 카메라를 통해 물체의 위치를 파악하여 생성된 이동 경로는 상용 FC(Flight Controller)인 Pixhawk2로 전송되어 최종적으로 드론의 모터를 제어한다. 해당 제어기는 오픈소스 펌웨어 PX4를 기반으로 최적화된 비행성능을 낼 수 있도록 알고리즘을 수정하였으며 Control 파트의 소프트웨어 구조는 다음 <Figurer3>과 같다. 각 노드는 uORB 메시징을 사용하여 publish, subscribe를 통해 비동기로 데이터를 전송한다.

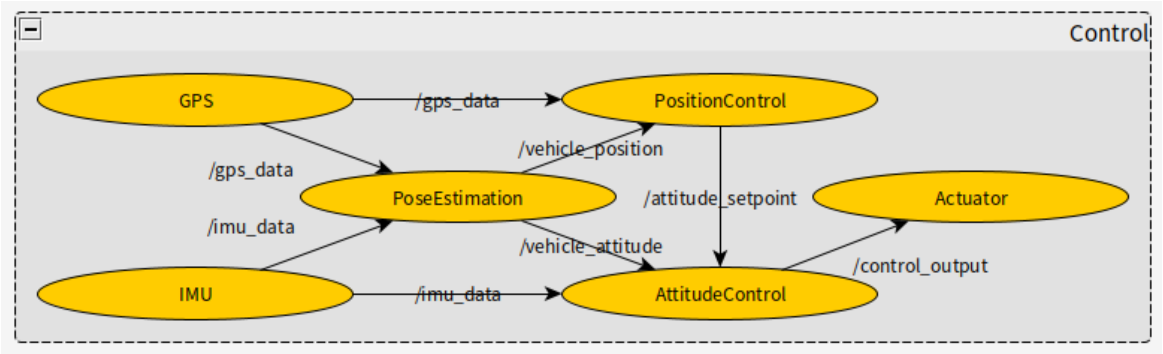

Figure 3 Control System Architecture

전체 제어프로세스는 다음과 같다.

① Planning파트에서 자체 개발한 임무 운용프로세스를 거쳐 position setpoint(x, y, z, yaw)를 생성하고 MavLink 프로토콜(mavros)을 통해 이를 Position Controller에 전송한다.

② Pose Estimation은 수집되는 IMU, GPS 데이터를 사용하여 기체의 현재 위치 및 자세를 추정하고 roll, pitch, yaw(x, y, z) 각 축에 대한 Position 벡터, Velocity 벡터를 Position Controller에 전달한다. 같은 과정으로 Attitude 및 Rate 벡터를 Attitude Controller에 전달한다.

③ Position Controller는 목표 위치와 현재 위치를 바탕으로 attitude setpoint 벡터를 생성하여 Attitude Control에 전달한다.

④ Attitude Controller는 목표 자세와 현재 자세를 바탕으로 Torque벡터를 생성하고 이는 Mixer를 거쳐 모터 출력으로 변환된다.

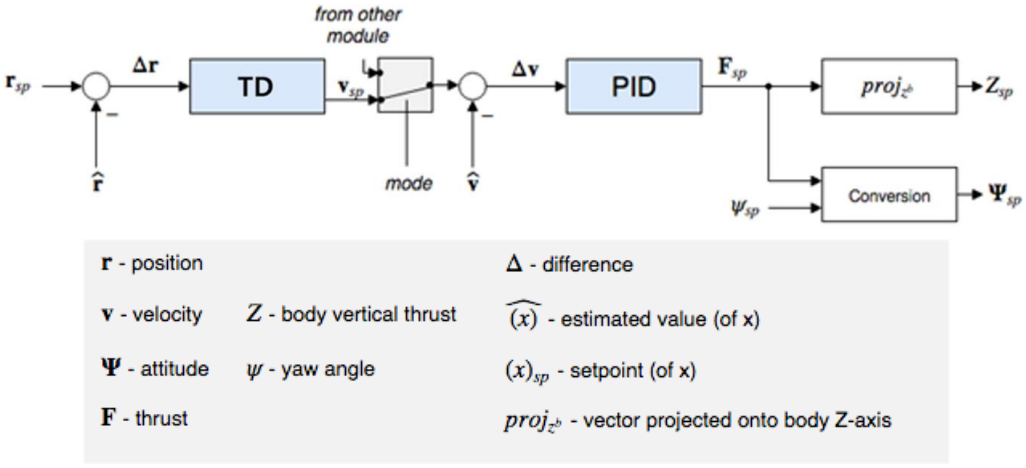

Figure 4 Position/Velocity Controller Diagram

<Figure 4>를 통해 나타난 Position Controller는 에러 미분 시 노이즈 증폭을 최소화하면서 빠르게 목표 상태에서 접근하도록 하는 TD(Tracking Differentiator) 제어를 기반으로 구성되었다. TD 제어를 통해 속도 벡터가 생성되면 현재 속도 벡터와의 에러를 사용해서 PID 제어를 통해 Thrust를 도출한다. 최종적으로 목표 vertical thrust와 attitude를 출력하여 자세제어기에 전달한다.

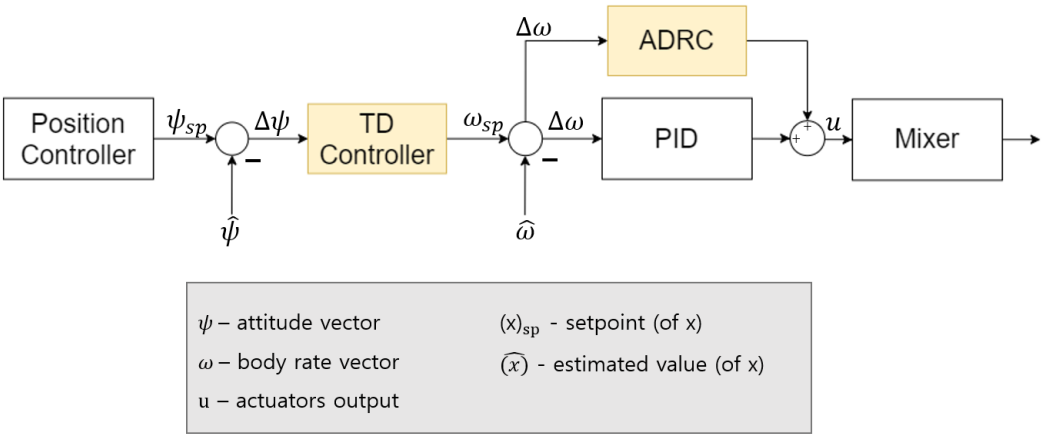

Figure 5 Attitude/Rate Controller Diagram

<Figure 5>는 Attitude/Rate Controller의 블록 다이어그램이다. Attitude Controller는 Position 과 마찬가지로 TD Controller를 사용하였다. 이를 통해 출력된 각 축에 대한 rate를 PID 제어한 값과 ADRC 제어한 값을 합하여 Torque를 계산한 후 Mixer에 전달하고, Mixer는 Actuator output을 도출해낸다. 최종적으로 도출된 모터 PWM 출력에 따라 드론이 제어된다.

○ Software 구성

전체 S/W 설계 및 각 모듈 간 통신 블록 다이어그램을 아래 <Figure 6>에 표현하였다. 자율비행 소프트웨어 개발을 위해 로봇용 미들웨어인 ROS(Robot Operating System)을 사용하여 노드 단위의 프로세스를 가능하게 하였고, 시뮬레이션 및 시각화 툴을 이용하여 편리한 개발환경을 구축하였다.

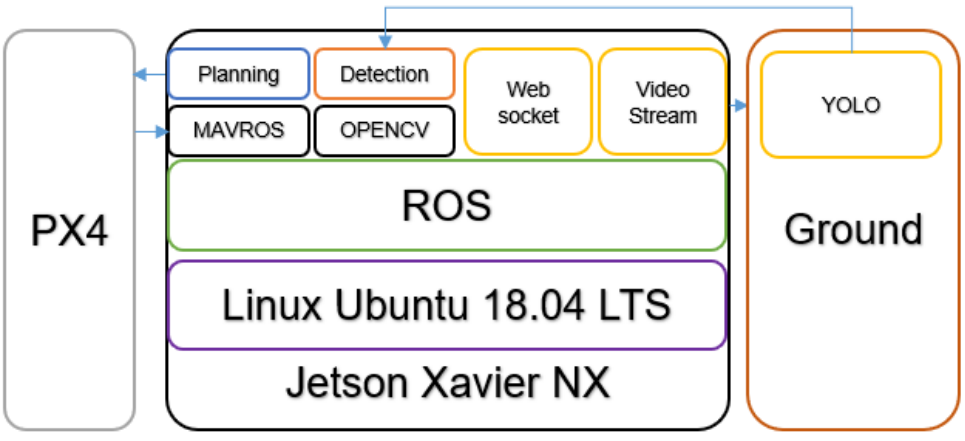

Figure 6 SW Architecture

① PX4: Flight Controller로 사용하는 Pixhawk2와 호환되는 오픈소스 비행제어 펌웨어

② ROS: Linux 기반 오픈소스 로봇 미들웨어, 다양한 임베디드 시스템 라이브러리를 지원

③ ROS package

✓ OpenCV 3.4.0: 실시간 컴퓨터 비전 프로세싱을 목적으로 하는 오픈소스 프로그래밍 라이브러리

- ✓ Detection: USB 카메라로부터 수집되는 image raw를 처리하여 사람을 탐지, 물체의 2D pixel 좌표를 3D Global 좌표로 변환하여 Planning에 전송
- ✓ Planning: 전달받은 Global 좌표로 드론이 비행금지구역을 우회하며 목표지점 좌표인 waypoint를 매순간 계획 생성 후 비행제어기 PX4로 전달
- ✓ MAVROS: 무인항공기 통신규약을 정의한 MavLink를 ROS에서 구현한 패키지로 Local/Global pose, vehicle, 기체 상태, 명령을 송수신하는데 사용

④ Networking

- ✓ Websocket Server: 상공에서 고정IP LTE 단말기를 활용하여 기체의 위치, 탐지 상황 등을 Web을 매개로 실시간 송수신, Websocket package를 이용하여 간단한 JSP파일로 Web 브라우저에서 동작하도록 함
- ✓ Video Stream: 기체의 실시간 수집 영상을 Web_video_server를 이용하여 실시간 전송

⑤ 지상시스템

- ✓ YOLO: 임베디드 시스템 프로세스의 성능한계로 사람 탐색에 실패했을 경우, 지상에서 비디오를 수집하여 YOLO v5를 통해 정확하게 처리한 사람위치를 Xavier NX로 실시간 재전송하여 사용한다.
- ✓ Q Ground Control: 오픈소스 무인항공기 지상관제시스템

⑥ Simulation

- ✓ PX4, Gazebo, QGC: MAVROS에서 지원하는 UDP proxy를 이용하여 시뮬레이션 환경을 구축하여 알고리즘을 검증. MAVROS의 명령을 받은 PX4 시스템의 작동을 Gazebo를 통해 시뮬레이션, QGroundControl을 통해 기체 상태를 모니터링
- ✓ Validation 패키지: 추출한 비행데이터를 기반으로 사람과의 거리, 고도 등을 목표치와 비교하여 정확성 확인 가능

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

자율비행 소프트웨어와 비행제어 펌웨어를 통합한 전체 ROS 노드 설계도는 다음 <Figure 7>과 같다. 노란 원은 독립된 프로세스인 노드이고, 화살표는 각 노드의 통신방향과 함께 publish 또는 subscribe되는 토픽이름을 /object_detected 등과 같이 나타내었다. 해당 설계도를 바탕으로 개발하여 실제 모든 노드와 프로그램을 구동한 전체 그래프는 ROS의 GUI 툴인 RQT를 이용하여 <Figure 8>에 나타내었다.

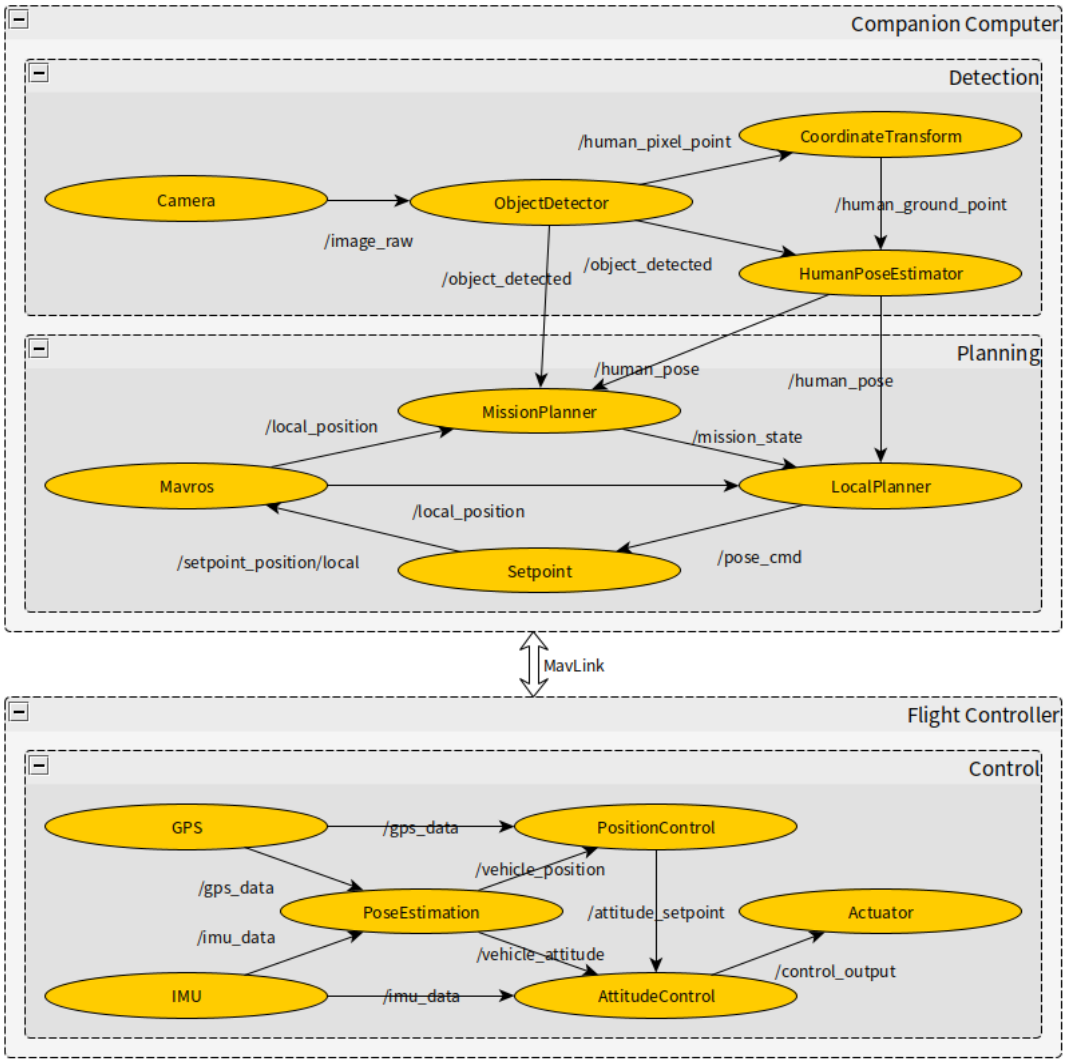

Figure 7 통합 S/W 구조

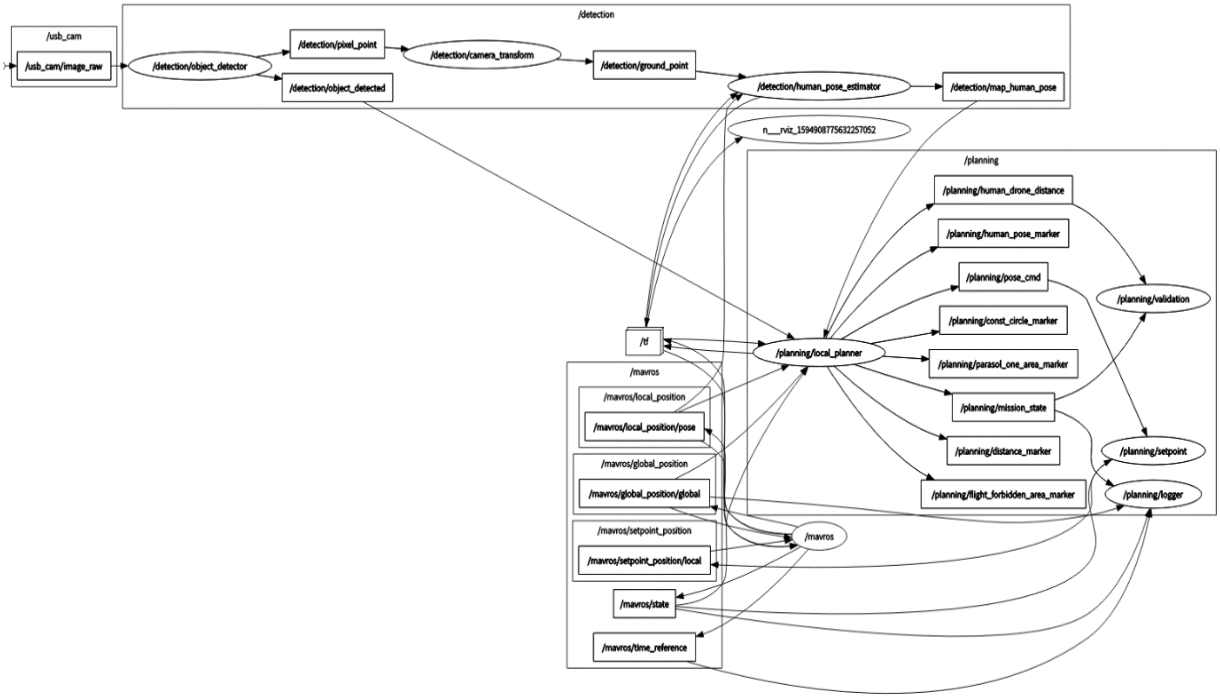

Figure 8 전체 S/W 구동 결과 그래프

○ Software 기능 (필요 시 알고리즘 설명 포함)

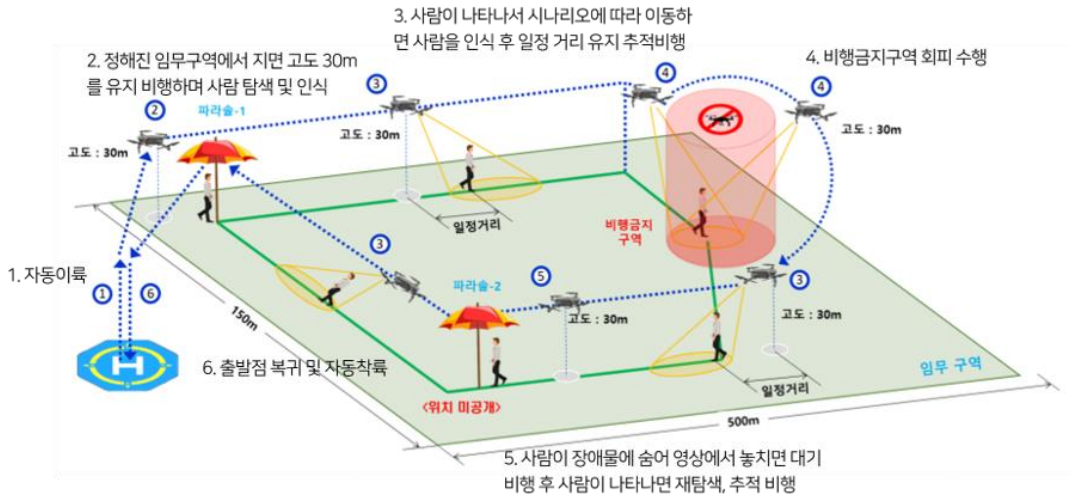

Figure 9 물체 추적 및 영상관제 시나리오

전체 임무수행 시나리오는 다음과 같다. 무인항공기는 주어진 이륙위치에서 Off-Board mode 로 자동 이륙 후 일정고도(30m)에 도달하면 첫번째 파라솔을 바라볼 수 있는 임무구역까지 자동비행을 진행한다. 첫번째 파라솔과 일정 거리 유지 대기비행 중 Vision 센서를 통해 사람이 탐색되면 해당 이미지 좌표를 기준좌표에 대한 3D 지면좌표로 변환한다. 이 후 Planning 알고리즘에 따라 인식된 사람 좌표와 일정거리를 유지하며 비행을 할 수 있는 waypoint 가 생성되고, 기체 자세제어를 통해 사람을 추적한다.

추적 비행 중 비행금지 구역에 다다르면 해당 구역을 회피하면서도 사람과의 일정거리를 유지하는 추적비행을 진행한다. 또한, 대상이 장애물에 가려 인식되지 않을 경우, 해당 위치에서 대기비행을 진행하며 재탐색을 한다. 대상이 목적지에 복귀하면 무인항공기는 이륙지점으로 돌아와 자동착륙을 진행하며 임무를 완수한다.

해당 임무과정에서 생성되는 모든 비행 및 영상데이터를 지상국에서 실시간으로 모니터링, 저장하며 정확한 임무수행을 실시간으로 확인한다.

① Planning Algorithm

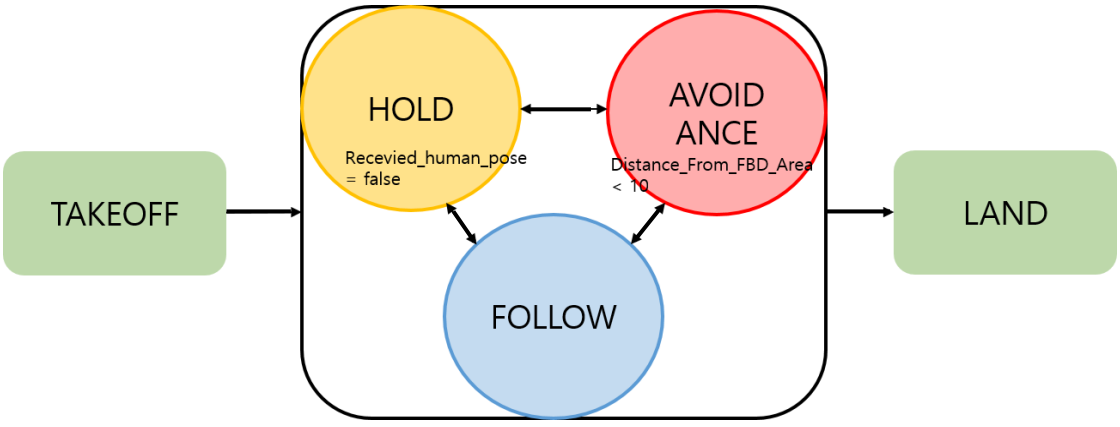

Figure 10 Planning Finite State Machine

Finite State Machine은 자율 운행체의 Mission 설정에 활용되는 Behavior Planning 기법으로 모드와 모드변환조건의 집합으로 구성된다. 본 팀은 이륙과 착륙을 제외하고, Following, Hold, Avoidance의 3가지 모드로 상태머신을 구성하였다. Following mode는 Detection 파트에서 사람을 탐지하고 좌표를 전달해주었을 때의 모드로, 드론이 이동할 다음 Waypoint를 탐색한다. 사람과 일정 거리만큼 떨어져 있는 점들을 생성하여 그 중 현재 드론 위치와 가장 가까운 점을 선택하여 다음 Waypoint로 결정해 FC에 넘겨준다. Following mode중 비행금지구역과 일정거리 이상 가까워지면 avoidance mode로 전환하게 되는데, avoidance 모드로 전환 시 비행금지구역 내부에서 생성된 Waypoint를 제외하고 다음 Waypoint를 선택하게 된다. Detection part에서 사람 탐색에 실패하게 되면 Hold mode로 넘어가 사람 좌표를 수신할 때까지 대기하게 된다.

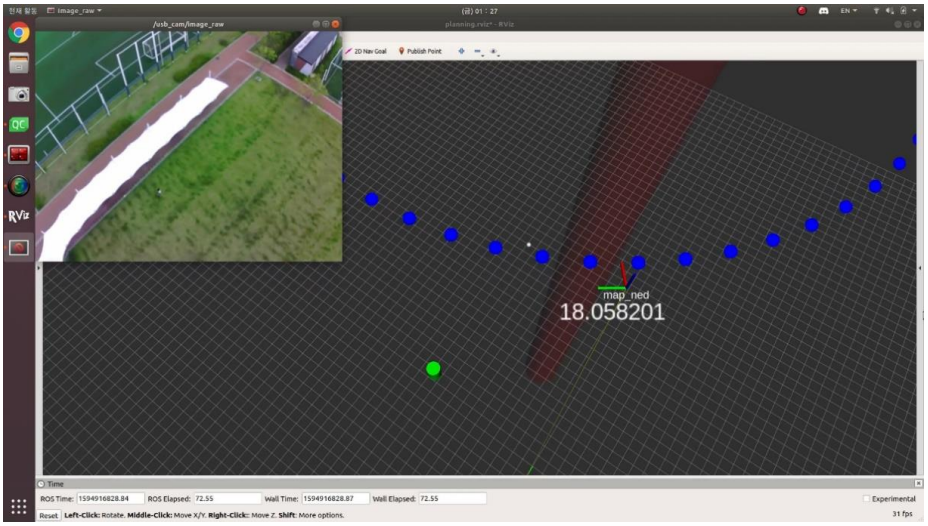

Figure 11 회피 및 추적 비행 Planning 시뮬레이션 화면

② Detection Algorithm

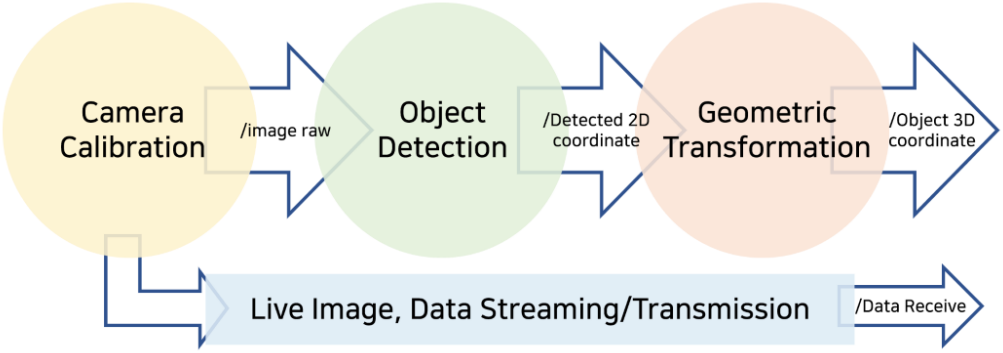

Figure 12 Detection Process

USB 카메라로부터 실시간 수집되는 이미지는 오픈소스 영상처리 라이브러리 OpenCV 함수를 이용하여 데이터 전처리, 필터를 거쳐 대부분의 배경 등의 노이즈를 제거한다. 이후, OpenCV 와 YOLOv5 등의 Object Detection 알고리즘을 적용시켜 실제 2D 이미지에서의 대상물체 pixel point 를 검출한다. 아래 <Figure 13>을 통해 물체 인식 및 좌표검출 알고리즘을 순서도로 나타내었다.

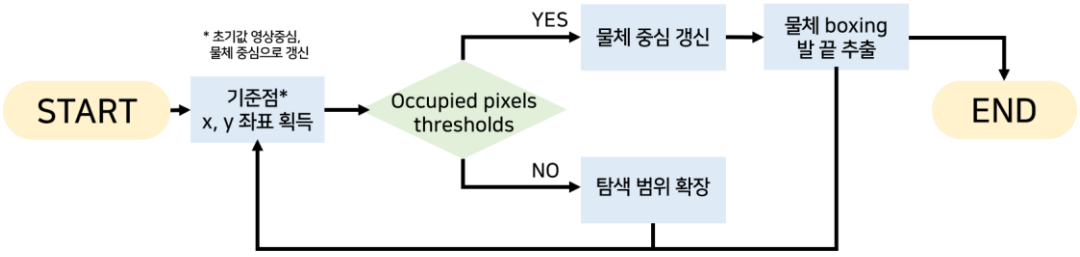

Figure 13 물체인식 알고리즘 Flow Chart

Detection 이 완료된 후 인식한 물체의 2D pixel 좌표를 camera calibration 데이터 및 현재 드론의 자세정보를 이용하여 파라미터를 도출한 후 3D 지면 좌표로 변환한다. 3D Geometry Transform 과정을 통해 획득한 기준좌표(map)에 대한 물체의 실제 지면좌표를 최종적으로 Planner 에 전송한다.

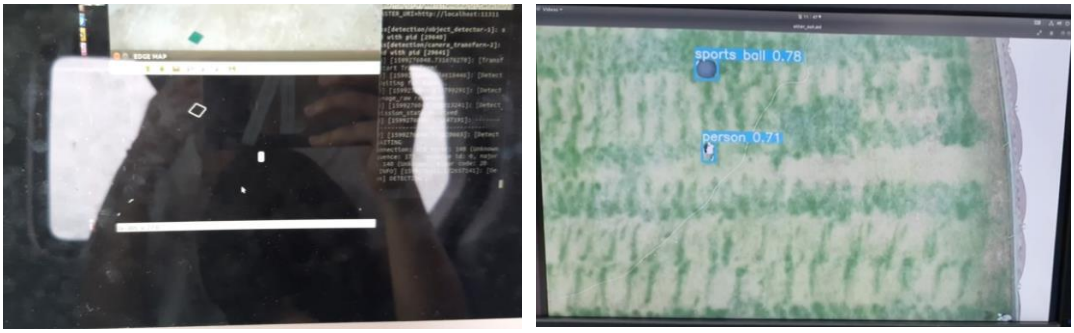

Figure 14 OpenCV, YOLOv5를 이용한 Object Detection

마지막으로 전체 비행과정에서 물체 인식 상황 및 실시간 수집 영상, 비행 데이터를 관제 센터에 전송하기 위해 영상 스트리밍 서비스를 구현하였다. 외부 인터넷 망에 연결된 컴퓨터로부터 전송되는 rostopic 들을 인터넷이 연결된 임의의 컴퓨터에서 수신 가능하게 함으로써 지상국의 웹 브라우저를 통해 실시간 데이터 수집 및 모니터링한다. 아래는 실시간으로 지상국에서 수집한 영상데이터 및 개발 프로세서 화면이다.

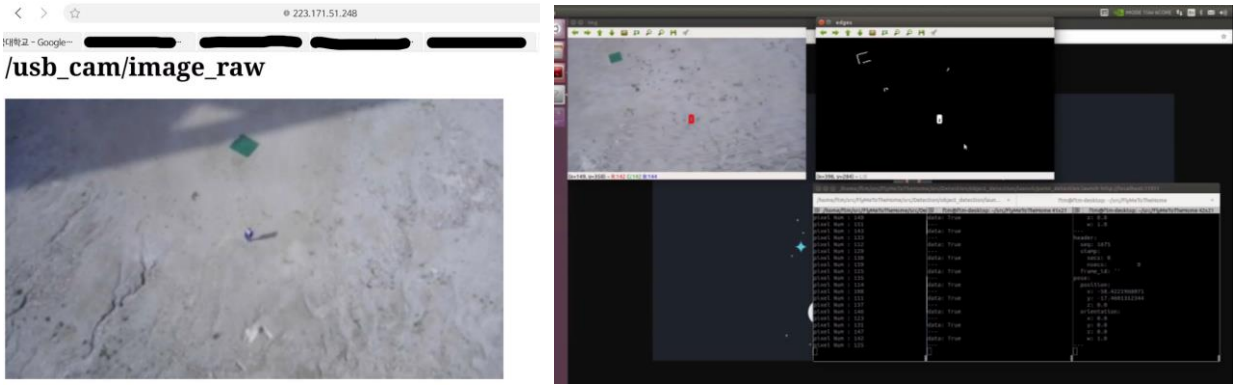

Figure 15 실시간 수집 영상데이터

③ Simulation

<Figure16>은 알고리즘 테스트 시 기체의 움직임을 시뮬레이션 하는 Gazebo 화면과 기체의 자세 및 waypoint 등을 시각화하는 Rviz 이다. 이를 통해 미션 수행 코드 개발 과정에서 매번 실제 비행테스트를 진행하지 않고도 알고리즘 수정 및 피드백, 디버깅 작업을 진행할 수 있었다. 또한 실제 비행 시에도 작성된 코드에서 각각의 비행상태를 실시간으로 출력하여 5 가지의 비행 수행모드를 지속적으로 확인할 수 있었다. 또한, 모든 비행 데이터는 rosbag 이라는 로깅(logging)시스템을 이용해 저장하여 다시 컴퓨터에서 실제 비행 데이터를 토대로 시뮬레이션을 진행할 수 있도록 하였다. 따라서 더 정밀한 피드백을 얻어 알고리즘 검증 및 디버깅 수정이 가능하였다.

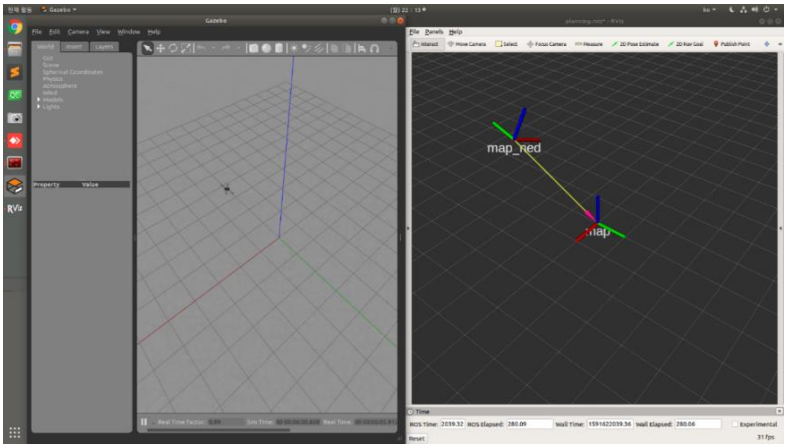

Figure 16 Gazebo Simulation & Rviz Visualization 화면

④ Control

자체 임무운용 ROS 소프트웨어(Detection, Planning)으로부터 도출된 목표점을 향해 비행하는 위치/속도, 자세를 계산함에 있어 자체 유도제어기를 개발하고 이를 오픈소스 PX4 에 적용하고자 했다. 이에 따라 PX4 내의 핵심 유도 제어기인 Position, Attitude Controller 모듈을 분석한 후 수정 및 보강하여 사용했다. 아래 <Figure17>을 통해 제어기 수정보강 전/후 제어기 성능을 비교한 그래프를 첨부하였다.

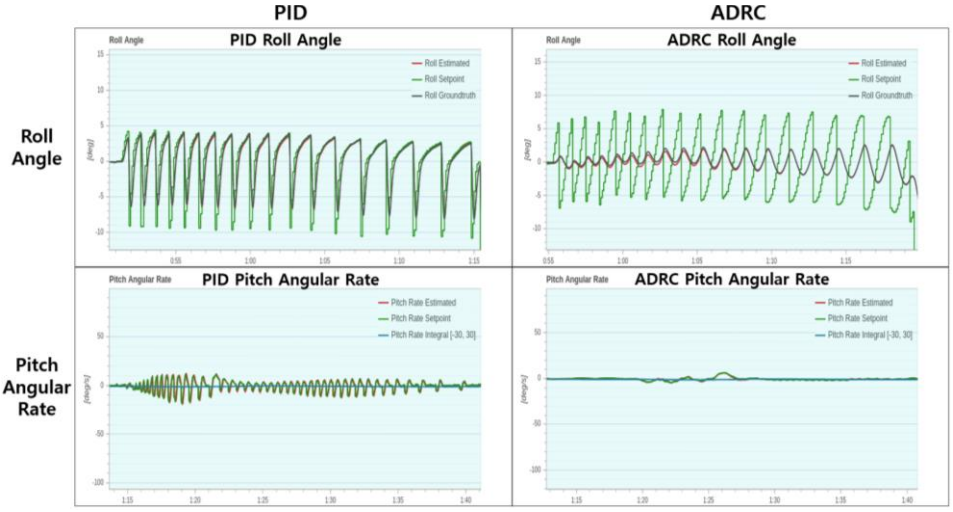

Figure 17 PX4 수정보강 전(PID)/후(ADRC) 제어기 성능 비교

○ 프로그램 사용법 (Interface)

- 비행 전: Google earth에서 비행금지구역, 미션 시작지점, 착륙지점을 kml 파일로 추출하여 위치 입력
- Launch: Object_Detection,planning_real,usb-cam-ftm-view 3개의 launch 파일 실행
- 드론: 조종기-수신기 Binding, Safety Switch를 누른 후 이륙지점에서 대기
- 지상: 지상에서 자율비행 시작 명령전달, 드론에 할당된 고정 IP의 8080포트로 접속하여 실시간 영상 및 비행데이터 수집

○ 개발환경 (언어, Tool, 사용시스템 등)

Table 1 개발환경

분류		내용
SW	OS	Linux Ubuntu 18.04 LTS, PX4
	Firmware	PX4 1.10.1 기반
	IDE	Qt Creator 4.9.2 for ROS, Visual Studio Code
	개발언어	C++(ROS,PX4), Python(ROS), JSP(Websocket)
	Simulator	Gazebo, Rviz
HW	Computer	Nvidia Jetson Xavier NX
	FC	Pixhawk 2.1
	CAM	ELP-USBFHD08S-LC1100
	Frame	800mm HexaCopter
	LTE Module	CNR-500W

개발 기술의 기반 시스템은 다음과 같이 구성되어 있다.

① ROS(Robot Operating System): 전 세계적으로 활발히 사용되고 있는 Linux 기반 로봇 미들웨어이다. 노드 단위의 분산객체 시스템으로써 각 센서 및 알고리즘 별 세부 미션의 개발과 디버깅이 편리하여 시나리오 기반의 로봇 작동에 특화되어 있다. ROS는 오픈소스 영상처리 라이브러리인 OpenCV 등을 기본 지원하여 Vision, LiDAR 센서 등을 사용하기 편리하다. Simulation, Visualization tool을 제공하여 효율적인 개발이 가능하다. 라즈베리파이, 아두이노와의 호환성뿐만 아니라 다양한 라이브러리, 로봇 모델 및 시뮬레이션 툴을 제공하기 때문에 개발된 SW의 추후 확장성을 생각해볼 수 있다. 본 프로젝트에서는 임베디드 보드에 Linux Ubuntu 18.04 OS를 설치하고 ROS melodic 버전을 설치해서 사용했다. ROS는 C++, Python 언어를 지원하며 본 팀에서는 대부분 C++로 노드를 작성하고 Python도 일부 사용했다.

② PX4: 무인항공기 분야에서 널리 사용되며 거대한 개발 생태계를 가진 오픈 소스 Autopilot Firmware로, 검증된 유도제어항법 시스템과 안전한 Failsafe 기능을 제공한다. 멀티콥터의 위치 및 자세 제어에 대한 전문적인 지식이 없는 사용자도 목표점 명령을 통해 자율비행 드론 제어를 가능하게 한다. PX4의 각 기능을 담당하는 구성요소들은

모듈화 되어있어 사용자가 쉽게 수정 및 재사용이 가능하다.

③ MAVROS: 무인항공기에서 표준으로 사용되는 통신 프로토콜인 MAVLink(Micro Aerial Vehicle Link)를 ROS에서 사용하기 위한 오픈소스 라이브러리이다. 임베디드 컴퓨터(이하 Off-Board Control Unit)에서 구동되는 ROS 프로그램은 목표점을 설정하는 /setpoint, 자세를 제어하는 /control 등의 다양한 topic을 publish할 수 있다. PX4에서는 /global_position, /imu 등의 다양한 데이터를 publish하므로 Off-Board control unit에서는 이를 subscribe하여 비행상태와 센서 데이터를 실시간으로 받아와 알고리즘 계산에 활용한다.

해당 오픈소스들을 활용하여 자율비행 시스템을 다음 <Figure18>과 같이 구성하였다. 자체 개발한 ROS 자율비행 소프트웨어는 추적 미션을 수행하기 위해 도출된 비행 목표점(position setpoint)를 PX4에 전송한다. PX4에서는 목표점을 받아 기체의 위치 및 자세를 제어하고 GPS 센서 데이터를 ROS에 전송한다. 두 모듈은 MAVLink 프로토콜을 사용하여 통신한다.

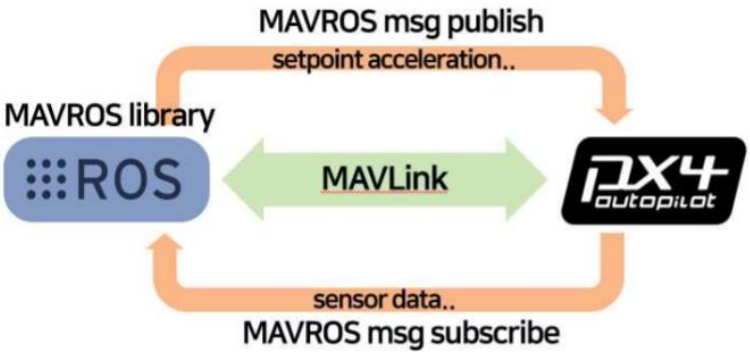

Figure 18 자율비행 전체 시스템 구조도

□ 개발 프로그램 설명

○ 파일 구성

① Detection

✓ CV_pointer: OpenCV 기반 Object Detection을 수행하여 사람의 2D 픽셀좌표 출력

✓ Point_transform: 출력된 2D 픽셀좌표를 3D global 좌표로 변환 및 Planner로 전송

② Planning

✓ Global_points_loader & ENU_conversion: 비행의 이륙위치, 비행금지구역, 사람의 출발점을 저장한 kml파일을 불러와 local planer에 전달

✓ Local_planner: Planning 부분의 주요 기능을 맡고 있는 노드로 이륙과 착륙, Mission 중 비행 모드 설정을 담당

✓ Setpoint: local planner에서 출력한 waypoint를 받아 드론의 FC로 전달

✓ Gazebo_simulation: 구현한 알고리즘 디버깅용 시뮬레이션 구동 노드

✓ Logger: 비행 중 데이터(GPS data, 비행 모드, 시간 등) 기록

③ Control

✓ 위 하드웨어 제어 프로세스 참고

④ Launch

✓ Planning_real/ Point_detection/ Usb_cam.launch

○ 함수별 기능

Table 2 Package 별 함수기능

Cv_Pointer	Callback 함수	실시간 usb image 수신, mission state 수신
	Detection	OpenCV 내장함수를 이용한 전처리, 물체 인식, 좌표 추출
Point_transform	Callback 함수	물체 pixel point, mission state, 맵/드론 변환관계 수신
	getTFValue	좌표계 변환을 통한 변환관계 데이터 수신 및 저장
	Quaternion2RotMat	변환관계 Quaternion을 Rotation, Translation 변수로의 변환
	getTransformed	물체의 3D 지면좌표 추출
	LowPassFilter	안전한 운행을 위한 노이즈 데이터 보정

Localplanner::	생성자	Subscriber, publisher, service Client초기화. TAKEOFF 미션 전달
	Wait	현재 위치, 미션이 지정될 때까지 정지
	Callback 함수	연결상태, mission point, 드론 방향, 사람 위치, detection 여부, local pose를 subscribe
	Takeoff	안정적인 이륙을 위해 5단계의 이륙절차 수행.
	Following	사람 위치를 기반으로 드론이 이동할 다음 Waypoint 탐색. 사람과 일정 거리만큼 떨어져 있는 점들을 생성하여 그 중 드론과 가장 가까운 점을 골라 이동거리 최소화
	Hold	물체가 탐지되지 않았을 경우 대기
	avoidance	비행금지구역이 일정 거리 내로 들어왔을 때 수행. 생성한 점 중 비행금지구역 내에 있는 점을 제외하고 다음 Waypoint 선택
	land	사고 방지를 위해 3단계로 나누어 착륙절차 수행
Setpoint::	생성자	Subscriber, publisher, service 초기화. 드론 시동 요청
	Precheck	드론 FCU와의 연결 확인
	Callback함수	현재 상태, 위치를 Subscribe
	pubInitialPose	초기위치, 현재위치 publish
	preSetting	드론 비행 모드, 암 상태 확인
	Setpoint	Local planner에서 생성한 local position 전달
global_points_loader	importKmlFile	미리 지정한 좌표 kml파일을 업로드
enu_conversion::	setOrigin	ENU Conversion의 기준점 설정
	EnuConversion	입력된 LLH좌표를 기준점에 대한 local좌표로 변환하여 localplanner에 전달
Simulation::	생성자	Subscriber, publisher, service Client초기화, 사람 속도와 대기시간 지정
	Wait	Simulation 시작점을 받을 때까지 대기
	Callback함수	Mission,가상의 미션 시작점과 비행금지구역위치 Subscribe
	calcUnitVec	두 좌표간 Vector 계산
	Move	가상의 사람좌표 publish
	markPoint	Rviz에 사람, 드론, Local Planner의 Waypoint 후보를 보여줌
Validation::	생성자	Subscriber, publisher, service Client초기화
	Precheck	미션이 시작되고 거리정보를 받을 때까지 대기
	Callback함수	미션정보, 거리정보, 사람정보 subscribe
	varianceCal	사람을 추적하는 동안 사람과 드론까지의 거리의 평균과 표준편차 취합
Logger::	생성자	Subscriber, publisher, service Client초기화
	Callback	시간, global pose, mission state, 비행모드 subscribe

○ 기술적 차별성

① 모듈화를 통한 프로세스 간 종속성 약화

✓본 시스템은 크게 Detection, Planning, Control, Communication의 4가지 모듈로 구성

✓각 모듈을 독립적으로 구현하여 종속성을 낮추어 Input/Output 프로토콜만 일치하면 다른 미션을 수행하는 자율비행 드론의 개발에 쉽게 활용될 수 있다.

② Simulation 활용

✓서울 내 비행금지구역 등의 규제로 인한 비행테스트 어려움

✓각 모듈을 점검할 수 있는 Simulation Tool 활용(Gazebo, PX4 Firmware 가상 구동 등)

✓현실과 유사한 환경에서 알고리즘 수정 가능

✓Rosbag 이라는 logging 시스템을 적극 활용하여 이전 비행테스트 로그를 기반으로 In-door에서 알고리즘 성능향상 환경 구성

③ 물체 좌표 변환을 통한 기준 좌표계 통일

✓Detection에서 단순히 형태를 인식하는 것에서 벗어나, 물체의 위치를 위도 경도 기반의 global pose로 변환하여 지상의 한 기준 좌표계를 기준으로 모든 시스템을 표현 가능하게 함.

✓이는 단순히 사람 한 명의 위치를 지도에 띄우는 수준에서 벗어나 고가의 LiDAR 대신 저렴한 카메라 만으로도 이동하는 물체의 위치를 표현하는 실시간 mapping을 가능하게 하여 자율주행 및 비행 기술의 발전을 도모할 수 있다.

④ Finite State Machine을 통한 Behavior planning

✓Detection과 Planning 파트는 미션 중 맞닥뜨릴 수 있는 위험 및 예외 상황에 대비하기 위하여 비행상황을 설정하고 매순간 공유한다.

⑤ 여러 마스터 간 실시간 Topic 전달 시스템 구축

✓기존 하나의 ROS Master에 여러 컴퓨터가 연결되어 토픽을 주고받던 시스템에서 벗어나 Websocket Server를 통하여 다른 외부 환경에서도 메시지를 주고받을 수 있도록 함

✓이를 통해, 단순 웹 프로그램으로 간단히 드론과 같은 ROS 기반 로봇을 통제가능하고 각각의 독립적인 기기가 IoT 장비처럼 서로를 정보를 주고받음에 의의가 있음.

□ 개발 중 발생한 장애요인과 해결방안

○ Detection 알고리즘 및 프로세서 한계

- 아주 높은 상공(30m)에서 임베디드 보드를 이용하여 사람을 인식함에 있어 알고리즘 및 구동 프로세서 한계 발생. NVIDIA Xavier에서 직접 YOLO를 구동하여 사람을 탐색하려 했으나 성능 한계로 인하여 구동 중 다른 프로세스에 영향을 주어 정상적인 알고리즘 구동에 무리가 있었음
- LTE 상용망을 통한 비디오 실시간 스트리밍을 통해 지상의 Workstation에서 Yolo를 구동하여 OpenCV 알고리즘으로의 탐색 실패 시 FAILSAFE 역할을 할 수 있도록 함.

○ 테스트 기체 및 시뮬레이션 활용

- 800급의 본 기체를 바로 투입하여 Planning 알고리즘과 Tracking & Following 테스트를 진행하기에는 위험부담이 컸고, 코로나 19로 인한 교내 대운동장, 광나루 비행장 등의 폐쇄로 테스트 장소 확보 어려움 발생
- 간단한 알고리즘은 450급의 테스트용 쿼드콥터를 제작하여 검증
- logging 시스템을 활용하여 모든 비행테스트에서 출력되는 모든 로그 데이터를 저장해둔 후 In-door 환경에서 다시 play하며 비행 디버깅 및 알고리즘 검증 및 수정
- 작은 기체 및 시뮬레이션 활용을 통해 간소화된 비행 준비로 개발시간 절약 및 성공적인 개발 및 테스트 완료

○ 드론 법적 규제 및 비행한계

- 약 10kg의 드론을 30m 라는 높은 높이에서 테스트 비행을 하고 알고리즘을 수정하기 위해 장소물색, 알고리즘 수정 등 많은 시간 및 비용 발생
- 이전과 같이 시뮬레이션을 활용한 실내 디버깅 및 알고리즘 개발 환경 구성
- 조종자 팀원의 초경량비행장치 무인멀티콥터 국가 자격증 취득을 통한 안전한 테스트비행 진행

□ 개발결과물의 차별성

○ 비용 대비 성능

- 물체와의 거리를 탐지하기위해 수백만원대가 넘어가는 열화상카메라 또는 감지거리가 짧은 Stereo camera를 활용하지만, 단 monocular 카메라와 프로세서 하나만으로 물체와의 거리를 3차원으로 계산하는 모든 과정을 알고리즘만으로 처리

- 낮은 화질에서도 물체를 추적할 수 있는 알고리즘을 구현하여 센서 한계 및 비용에 비해 더 높은 고도 등의 극한 상황에서 작동할 수 있는 경제적이고 간단한 하드웨어 기반의 자율비행 시스템 개발

○ 탐지 및 추적 성능

- 기존 상용화되고있는 추적 드론은 저고도 환경에서 임무 수행
- Detection 개발 도중 10m, 20m 등 비교적 낮은 거리에서의 훈련 데이터는 많지만 그보다 더 높은 거리 환경에서의 테스트 데이터는 찾기 어려움
- 단순 2D 이미지 기반의 추적이 아닌 3D Geometry 변환을 통해 화면에서 물체를 잃거나, 비행상태에 문제가 생겨도 물체의 이전 global 위치 데이터를 이용하여 다시 비행상태 복구 및 정상적인 추적이 가능
- IoT 장비로 활용됨으로써 자율주행 시스템 등의 기존 IoT 장비와의 데이터 동시 수집 및 병합, 공유가 가능

○ 오픈소스 기반 자율비행 플랫폼 구축

- 최근 정부에서 '한국형 도심항공교통(K-UAM) 로드맵'을 통해 하늘길 출퇴근을 가능케할 수 있는 이동수단(모빌리티)의 2025년 현실화, 30년 상용화 목표를 발표
- 기존 항공업계에서는 기체를 안정적으로 띄우고 운행하는 항법, 유도제어 기술이 핵심 연구대상이었음
- 기술을 선도하기 위해서는 비교적 빠르게 진행되고 있는 자동차 등의 기존 자율시스템 분야와의 협업이 필요
- 제어, 미션파트를 모듈로 분리함으로써 자율시스템과 항법 제어파트 간의 이식성, 확장성이 뛰어나 빠른 성능 향상 및 알고리즘 적용에 최적화된 자율비행 플랫폼으로 활용될 수 있음.

□ 개발 일정

No	내용	2020年															
		6月				7月				8月				9月			
1	미션 설정																
2	기체 설계 및 제작																
3	SW 플랫폼 구축																
4	Planner 설계																
5	경로 알고리즘 개발																
6	Simulation & Debug																
7	제어 알고리즘 개발																
8	'인식' 기초 스터디																
9	카메라 선정																
10	'인식' 시스템 구축																
11	탐지 알고리즘 작성																
12	물체 추적 개발																
13	3D 좌표변환																
14	실 비행테스트																

□ 팀 업무 분장

No	구분	성명	참여인원의 업무 분장
1	팀장	신수연	[Detection 팀장] : Detection 전체 시스템 설계 및 알고리즘 개발 총괄
2	팀원	이기은	[Planning 팀장] : Planning 전체 시스템 설계 및 알고리즘 개발 총괄 : 항공기 제어 알고리즘 개발
3	팀원	홍동원	[SW Platform 팀장] : SW Platform 환경 구성, 자율비행 예비 조종자 : Ground Control System 개발
4	팀원	김정석	[Communication 팀장] : 드론과 지상간 네트워크 연결 및 비디오, Websocket 연결 관리 : Topic 송수신 시스템 구축
5	팀원	전진표	[HW Platform 팀장] : 기체 HW 설계 및 제작, 수동비행 조종자
6	팀원	김주엽	[Detection 팀원] : 실시간 수집 이미지에서 사람 인식 알고리즘 작성 : 2D 좌표 수집 물체 지면 좌표로의 변환
7	팀원	윤병욱	[Planning 팀원] : 사람 거리 유지 알고리즘 개발 참여 : Yolo 훈련 데이터 수집 및 훈련 가중치 생성