Main page

Welcome to the presentation of my thesis for my bachelor degree in software engineering.
----

introductions

The topic is autonomous drone flight, and to be more precise, the design and development of software applications that enables the drone to be controlled by software while being able to communicate with other robots in the network and be capable of indoor flight.

requirements

To achieve these requirements, this project is based on three main existing software components:

- ROS2, robot operating system, a middleware that creates virtual communication channels between all the components and allows exchange of information on runtime
- PX4 autopilot: flight software, installed in the flight controller, handles all the sensor data and is the interface from high level commands to low level hardware controls
- Optitrack, a motion capture system that is used to get the local position of the drone, this component was needed because GPS barometer and magnometer don’t work really well indoor.

1 - 1:40 MIN

System overview

There are 3 main hardware components

- Tracking with optitrack 8 cameras that provide precise 3d position and rotation with a server in the lab.
- Commander computer: ground computer, controls the drone, sends the actions to perform.
- Drone itself:

1) body
2) Flight controller: px4 software
3) Companion: single board computer mounted on the drone, receives commands and sends drone information through ROS messages

2 – 2:30 MIN

commander

I want to focus on the commander, so what runs on the ground computer

- Launch applicaion: main process, server like application

1) Connects to the drone, sends stay alive signals, receives status from drone
2) Interfaces with the other components with ROS messages
3) Handles commands. Instant (arm, set position) actions are sent to the drone, while long lasting action are handled with a thread

This is the core, then there are optional components

- GUI: quality of live component, can send commands through a graphical interface and provides the drone status on screen.

  You can easily send commands with the respective parameters to the main controller.
- Rviz, modified program to visualize paths, theoretical and real position, useful to analyze the history of the flight and validate the tracking.

This actually runs virtualized inside a docker container

3:30 – 3:40 MIN

Conenctions ?????

- Every system has it’s own message standard, many converters are in place to pass everything in ROS
- OPTITRACK - Opti → ROS geometry pose → coordinate conversion→ ROS (px4) pose
- FLIGHT CONTROLLER - Uorb → ROS (px4)
- GUI - Commander messages → ROS (px4) commands
- RVIZ - ROS (px4) pose → ROS navigation messages

Flight data

- 15 min autonomy
- Might have problems with very small movements (<5cm)
- Slow on purpose because we are in the lab

developement

- A lot of ideas were tested and scrapped

Improvement and conclusion

One of the most difficult part is to make all the components communicate with each other. I think I handled it well.