Contents

I. Introduction

This document is going to describe the non-linear control system which can be used for drone controller. Thanks to laboratory LCIS projects, some non-linear controllers for the Crazyflies were introduced.

In this project, we will focus to control drone and then integrate with AI for our business. Follow this document, we will present the simple guide to start with Crazyflie.

II. Crazyflies

Firstly, if you have no idea about Crazyflies, please take a look to this [link](https://www.bitcraze.io/documentation/hardware/crazyflie_2_1/crazyflie_2_1-datasheet.pdf) which describes detail about the lastest version of this drone.

1. Drone model

We will introduce a dynamic model of the drone which describes the relation between the position of the quadcopter in the inertial reference frame (IF) and the roll-pitch-yaw (symbolized respectively φ, θ, ψ) as shown in the figures below:

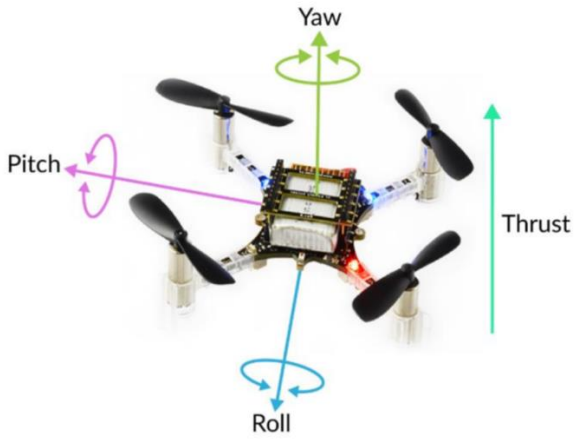


Figure 1: Roll-pitch-yaw and thrust

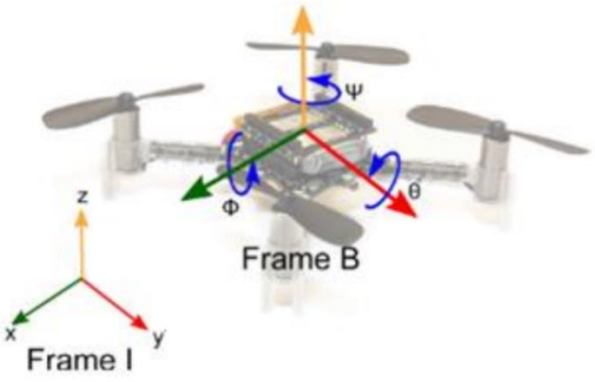


Figure 2: Body reference frame (BF) and inertial reference frame (IF)

Assuming that the quadcopter is a rigid body, we have its Newton-Euler equations:

$$m.\ddot{\xi} = m.\vec{g} + \vec{T}.{}^{I}_{B}R \quad (1)$$

Where:

$\xi = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$: the position of the quadcopter in the IF.

$\vec{T} = \begin{bmatrix} 0 \\ 0 \\ T \end{bmatrix}$: the thrust (in the BF).

$\vec{g} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}$: the gravitational acceleration

${}^{I}_{B}R = \begin{bmatrix} c\theta c\psi & s\phi s\theta c\psi - c\phi s\psi & c\phi s\theta c\psi + s\phi s\psi \\ c\theta s\psi & s\phi s\theta s\psi + c\phi s\psi & c\phi s\theta s\psi - s\phi c\psi \\ -s\theta & s\phi c\theta & c\phi c\theta \end{bmatrix}$: the rotation matrix to describe the orientation of the BF with respect to the IF.

$$(1) \Leftrightarrow \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \frac{T}{m} \begin{bmatrix} c\phi s\theta c\psi + s\phi s\psi \\ c\phi s\theta s\psi - s\phi c\psi \\ c\phi c\theta \end{bmatrix}$$

Note: φ: roll, θ: pitch, ψ: yaw

We also have:

$$\begin{cases} sin(\phi) \approx \phi, cos(\phi) \approx 1 \\ sin(\theta) \approx \theta, cos(\theta) \approx 1 \\ T = T_0 + \Delta T = T_0(1 + \frac{\Delta T}{T_0}) = mg(1 + \frac{\Delta T}{mg}) \end{cases} \text{With} \begin{cases} \phi \leq 10^\circ \\ \theta \leq 10^\circ \\ \frac{\Delta T}{mg} \ll 1 \end{cases}$$

Therefore,

$$\begin{cases} \ddot{x} = g(\theta.c\psi + \phi.s\psi) \\ \ddot{y} = g(\theta.s\psi - \phi.c\psi) \\ \ddot{z} = \dfrac{\Delta T}{m} \end{cases}$$

$$\Leftrightarrow \ddot{x}.c\psi + \ddot{y}.s\psi = g(\theta.c^2\psi + \theta.s^2\psi)$$

$$\Leftrightarrow \ddot{x}.c\psi + \ddot{y}.s\psi = g.\theta$$

$$\Leftrightarrow \theta = \frac{1}{g}(\ddot{x}.c\psi + \ddot{y}.s\psi)$$

Similar, we have:

$$\phi = \frac{1}{g}(\ddot{x}.c\psi - \ddot{y}.s\psi)$$

We have: $T = T_0 + \Delta T \Leftrightarrow \Delta T = T - T_0$

$$(3) \Leftrightarrow \ddot{z} = \frac{\Delta T}{m} \Leftrightarrow \ddot{z} = \frac{T - mg}{m} \Leftrightarrow T = m\ddot{z} + mg$$

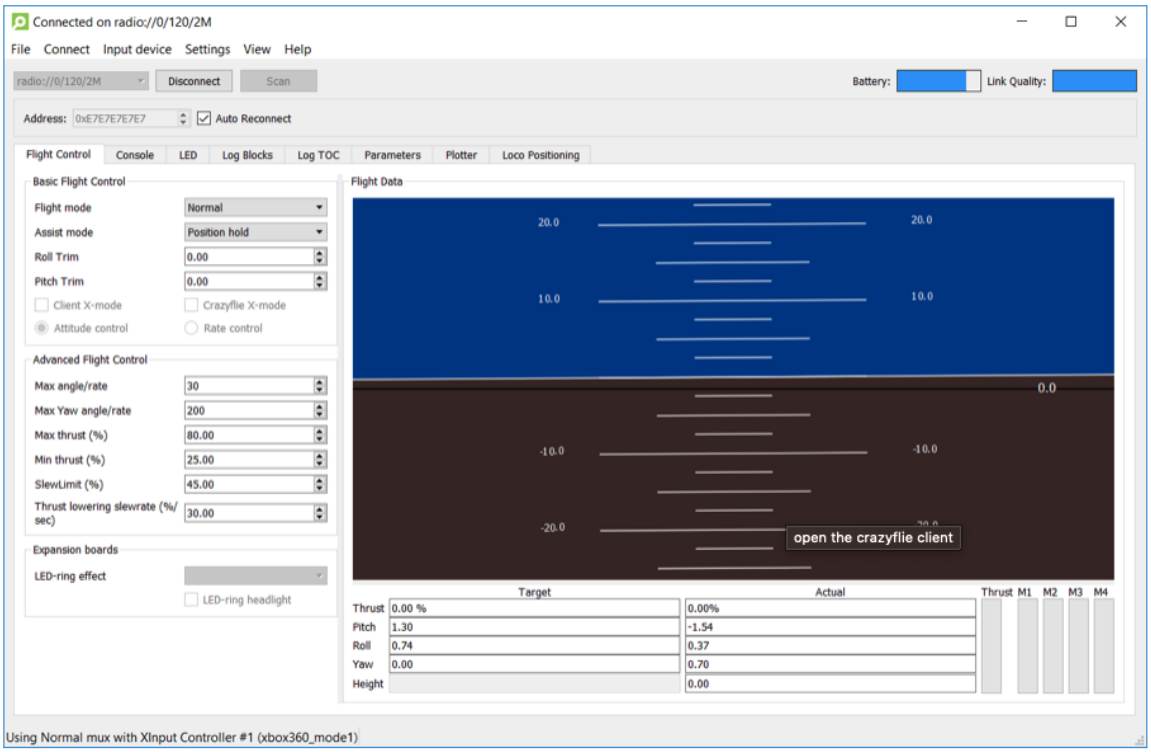
$$\Rightarrow\quad \begin{cases} \ddot{x} = g(\theta.c\psi + \phi.s\psi) \\ \ddot{y} = g(\theta.s\psi - \phi.c\psi) \\ \ddot{z} = \dfrac{\Delta T}{m} \end{cases}$$

$$\Leftrightarrow \begin{cases} \phi = \dfrac{1}{g}(\ddot{x}.c\psi - \ddot{y}.s\psi) \\ \theta = \dfrac{1}{g}(\ddot{x}.c\psi + \ddot{y}.s\psi) \\ T = m\ddot{z} + mg \end{cases} (*)$$

2. Crazyflie platform

Crazyflie is a small quadcopter that designed and developed by Bitcraze (www.bitcraze.io). Beside the Crazyflie quadcopter, they also create the open source platforms like the "Crazyflie PC Client Interface" and the "Loco Positioning System" (LPS).

Note: in this project, we work on the Crazyflie 2.1.



In this figure, we can see the software supported by Bitcraze. Via the firmware updated by manufacture, we can receive roll, pitch, yaw and thrust that are calculated from 6 DoF sensor on our drone.

3. PID controller

The idea is to discretize the drone's model and use the PID controller. The control loop is described as in this below figure:

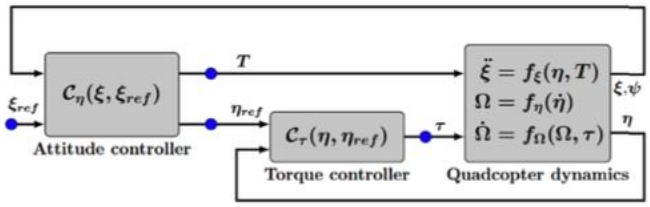


Figure 2: General control diagram

The study of a drone and a quadcopter is controlled by the use of two PIDs implemented in cascade. The first serves as regulations for the three axes of the drone (φ, Ψ, θ, and Thrust), and the second serves as regulation at the moment of force. In the control diagram, the position of the Crazyflie is retrieved by a tracking system (*not available at present but finding a solution*), then is calculated and sent to the Crazyflie while the moment of the Crazyflie is calculated in real time on the microcontroller implemented in the drone. As part of the project, we were interested in the development of the regulation of the three axes of the drone, namely the first PID.

Here, we want to regulate the error between the current position and the reference:

$$\begin{cases} e_x = x - x_{ref} \\ e_y = y - y_{ref} \\ e_z = z - z_{ref} \end{cases}$$

$$\Leftrightarrow \begin{cases} \ddot{e_x} = \ddot{x} - \ddot{x_{ref}} \\ \ddot{e_y} = \ddot{y} - \ddot{y_{ref}} \\ \ddot{e_z} = \ddot{z} - \ddot{z_{ref}} \end{cases}$$

We also have $\ddot{e}$ by the equation below:

$$\ddot{e} + K_p.e + K_d.\dot{e} + K_I.\int e = 0$$

$$\Leftrightarrow \begin{cases} \ddot{x} - \ddot{x_{ref}} + K_p.e_x + K_d.\dot{e_x} + K_I.\int e_x = 0 \\ \ddot{y} - \ddot{y_{ref}} + K_p.e_y + K_d.\dot{e_y} + K_I.\int e_y = 0 \\ \ddot{z} - \ddot{z_{ref}} + K_p.e_z + K_d.\dot{e_z} + K_I.\int e_z = 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} \ddot{x} = \ddot{x_{ref}} - K_p.(x - x_{ref}) - K_d.(\dot{x} - \dot{x}_{ref}) - K_I.\int (x - x_{ref}) \\ \ddot{y} = \ddot{y_{ref}} - K_p.(y - y_{ref}) - K_d.(\dot{y} - \dot{y}_{ref}) - K_I.\int (y - y_{ref}) \\ \ddot{z} = \ddot{z_{ref}} - K_p.(z - z_{ref}) - K_d.(\dot{z} - \dot{z}_{ref}) - K_I.\int (z - z_{ref}) \end{cases}$$

In the end, we have:

$$\begin{cases} \phi = \dfrac{1}{g}(z_1.c\psi - z_2.s\psi) \\ \theta = \dfrac{1}{g}(z_1.c\psi + z_2.s\psi) \\ T = mz_3 + mg \end{cases}$$

With:

$$\begin{cases} z_1 = \ddot{x_{ref}} - K_p.(x - x_{ref}) - K_d.(\dot{x} - \dot{x}_{ref}) - K_I.\int(x - x_{ref}) \\ z_2 = \ddot{y_{ref}} - K_p.(y - y_{ref}) - K_d.(\dot{y} - \dot{y}_{ref}) - K_I.\int(y - y_{ref}) \\ z_3 = \ddot{z_{ref}} - K_p.(z - z_{ref}) - K_d.(\dot{z} - \dot{z}_{ref}) - K_I.\int(z - z_{ref}) \end{cases}$$

And the saturations as follows:

$$\begin{cases} \phi_{sat} = max(\min(\phi_{max}, \phi), \phi_{min}) \\ \theta_{sat} = max(\min(\theta_{max}, \theta), \theta_{min}) \\ T_{sat} = max(\min(T_{max}, T), T_{min}) \end{cases}$$

i. Improve the regulation along the axis x and y

We are now looking to improve the regulator on the x and y axes as a first step. To do this, we first calculate the error between the desired and actual position. Then, this error becomes the speed setpoint. That is, when this error is large, the drone has to move faster to get to the desired point. Otherwise, if the error is small, the drone is close to the desired point, the set point for the velocity will also be small. Thus, we get the following commands:

$$\begin{cases} \phi = \frac{1}{g}(z_1.c\psi - z_2.s\psi) \\ \theta = \frac{1}{g}(z_1.c\psi + z_2.s\psi) \\ T = mz_3 + mg \end{cases}$$

With:

$$\begin{cases} z_1 = K_{px}[(x_{ref} - x) - v_x] + K_{dx}\frac{d}{dt}[(x_{ref} - x) - v_x] + K_{ix}\int[(x_{ref} - x) - v_x]dt \\ z_2 = K_{py}[(y_{ref} - y) - v_y] + K_{dy}\frac{d}{dt}[(y_{ref} - y) - v_y] + K_{iy}\int[(y_{ref} - y) - v_y]dt \\ z_3 = K_{pz}[(z_{ref} - z) - v_z] + K_{dz}\frac{d}{dt}[(z_{ref} - z) - v_z] + K_{iz}\int(z_{ref} - z)dt \end{cases}$$

In order to calculate the speeds (v_x, v_y, v_z) we use the following derivative formula:

$$v_x = \frac{\Delta x}{\Delta t}, \qquad v_y = \frac{\Delta y}{\Delta t}, \qquad v_z = \frac{\Delta z}{\Delta t}$$

And the saturations as follows:

$$\begin{cases} IntX_{sat} = max(\min(IntX_{max}, IntX), IntX_{min}) \\ IntY_{sat} = max(\min(IntY_{max}, IntY), IntY_{min}) \\ IntZ_{sat} = max(\min(IntZ_{max}, IntZ), IntZ_{min}) \end{cases}$$

With:

$$\begin{cases} IntX = \int[(x_{ref} - x) - v_x]dt \\ IntY = \int[(y_{ref} - y) - v_y]dt \\ IntZ = \int(z_{ref} - z)dt \end{cases}$$

Finally, we got:

$$\begin{cases} \phi_{sat} = max(\min(\phi_{max}, \phi), \phi_{min}) \\ \theta_{sat} = max(\min(\theta_{max}, \theta), \theta_{min}) \\ T_{sat} = max(\min(T_{max}, T), T_{min}) \end{cases}$$

ii. Improve the regulation along axis z and develop a regulation for yaw:

We then take the same idea as in formula n3 to improve the regulation on the z axis and, we develop a regulation on the yaw (angle Ψ).

$$\begin{cases} \phi = \frac{1}{g}(z_1.c\psi - z_2.s\psi) \\ \theta = \frac{1}{g}(z_1.c\psi + z_2.s\psi) \\ T = mz_3 + mg \\ \Psi_{Rate} = K_p(\psi_{ref-\psi}) \end{cases}$$

With:

$$\begin{cases} z_1 = K_{px}[(x_{ref} - x) - v_x] + K_{dx}\frac{d}{dt}[(x_{ref} - x) - v_x] + K_{ix}\int[(x_{ref} - x) - v_x]dt \\ z_2 = K_{py}[(y_{ref} - y) - v_y] + K_{dy}\frac{d}{dt}[(y_{ref} - y) - v_y] + K_{iy}\int[(y_{ref} - y) - v_y]dt \\ z_3 = K_{pz}[(z_{ref} - z) - v_z] + K_{dz}\frac{d}{dt}[(z_{ref} - z) - v_z] + K_{iz}\int(z_{ref} - z)dt \end{cases}$$

And the saturations as follows:

$$\begin{cases} IntX_{sat} = max(\min(IntX_{max}, IntX), IntX_{min}) \\ IntY_{sat} = max(\min(IntY_{max}, IntY), IntY_{min}) \\ IntZ_{sat} = max(\min(IntZ_{max}, IntZ), IntZ_{min}) \end{cases} \quad With: \begin{cases} IntX = \int[(x_{ref} - x) - v_x]dt \\ IntY = \int[(y_{ref} - y) - v_y]dt \\ IntZ = \int(z_{ref} - z)dt \end{cases}$$

Finally, we got: $\begin{cases} \phi_{sat} = max(\min(\phi_{max}, \phi), \phi_{min}) \\ \theta_{sat} = max(\min(\theta_{max}, \theta), \theta_{min}) \\ T_{sat} = max(\min(T_{max}, T), T_{min}) \end{cases}$

iii. Kalman filter

We will use Kalman filter to reduce noise on the speeds. Via this filter, we hope we can optimize drone's smooth.

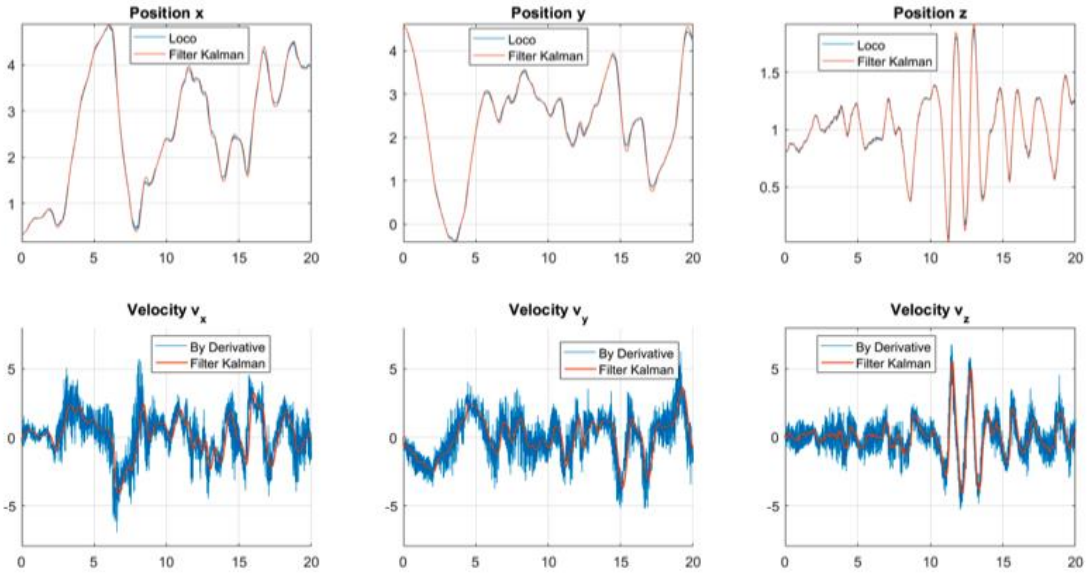


Figure 3: Example of estimate the speed by Kalman filter

III. Tracking system for drone(discussing)

In this section, we will list some ways which are possible to track and control drone in outdoor environment.

What do we need?

We need a system which can track accuracy drone's position in outdoor environment. For more detail, we need a real-time system which can update continuously drone's position. With these real positions and the position reference, we can use controller to automatic update the parameters. It's better for drone's performance. However, we have some ways to track accuracy position in indoor but there're not too many ways for outdoor environment. We can list some ways as follows:

1. GPS with 4 satellites

This method is the simplest way but it requires some conditions:

- Find a GPS module which can receive signal from at least 4 satellites. From that, we can calculate 3D position.

Drawbacks:

- Slow result(perhaps) because it can make this system having long sample time.
- In some cases, GPS tracking data can be compromised by high building, heavy tree cover, …

2. GPS with 3 satellites

   With signals from 3 satellites, we can only have 2D position. Therefore, we need one more thing to record z-axis. One of the solutions is ToF sensor which can calculate the distance to the ground.

   However, this method has some big problem:

- The accuracy of ToF sensors, which are sold in the market, isn't good enough. The maximum distance which is mentioned by dataset is around 4m, but the actually maximum distance is only around 2.5m.
- Long sampling time.
- Can be make compromised by some higher surface than ground.

Please take a look to these documents below, they also describe about GPS for drone tracking:

- [Integrating GLONASS with GPS for Drone Orientation Tracking]

(https://www.cc.gatech.edu/~dhekne/drone_orientation_glonass_gps.pdf)

- [GPS Guided Autonomous Drone]

(https://www.evansville.edu/majors/eecs/downloads/projects2016/CameronRobertsReport.pdf)

3. Wi-fi positioning

Though GPS is the most popular positioning system at present it does not perform well in indoor environments and metropolitan city areas. Wi-Fi positioning has received much attention due to its advantages with respect to indoor positioning and the wide spread of the Wi-Fi access points (APs). Its performance in an outdoor environment is also of interest as a Wi-Fi based positioning system can overcome the shortcomings of GPS.

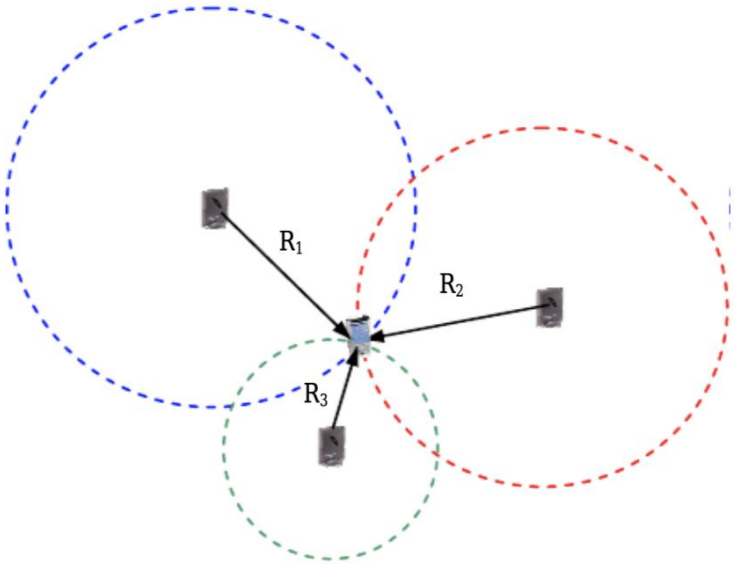


Fig. 1 Trilateration approach using Wi-Fi

Obviously, Wi-Fi is not designed or deployed for the purpose of positioning. However, the measurements of signal strength (SS) of the signal transmitted by either AP or station imply the possibility of finding the location of the mobile user (MU). In fact, several SS based techniques have been proposed for location estimation in indoor environments in which Wi-Fi is deployed (Bahl and Padmanabhan, 2000; Li et al., 2005). There are essentially two categories of such techniques.

- One uses a signal propagation model and information about the geometry of the building to convert SS to a distance measurement. 'Trilateration' can then compute the location of the MU (Li, 2006). This approach is simple to implement; however, it does have difficulties in building a sufficiently good model of signal propagation that is adequate for real world applications since so many factors affect the signal propagation.

- The other category of Wi-Fi positioning is 'Location Fingerprinting'. This class of technique has received more attention recently as it is able to address some of the problems related to non-line-of-sight (NLOS) and multipath propagation (Haldat, 2002). The basis of location fingerprinting is first to establish a database that contains the measurements of wireless signals at some reference points (RPs) in the area of Wi-Fi coverage. Then the location of the MU can be identified by comparing its SS measurements with the reference data (Ladd et al., 2002; Li et

al., 2005). The disadvantages of this approach are the database generation and maintenance requirements.

Please find more detail via this link:
https://www.researchgate.net/publication/251404219_On_outdoor_positioning_with_Wi-Fi

In this paper above, we can find some outdoor positioning technologies using Wi-Fi such as trilateration, fingerprinting as well as Wi-Fi plus GPS. Via both of those methods, we can try to find a solution to solve our tracking issue.

4. Use AI for semi-controller

The main idea of this method is to use AI for sending signal control for drone depends on the actual situation of drone.

Depends on the reference position and joystick signal control, we can train an AI model that can send control signals to drone (similar to joystick controller). That's reason why we call it is semi-controller.

In the other ways, we can train a model that can concern more about the wind speed, camera tracking or signal strength (Wi-Fi or other local signal) to increase the accuracy control of drone.

The advantages:

- We don't need to set up a system for 3D tracking.

Drawbacks:

- There're lots of unexpected external forces that can make wrong position update.
- Hard to collect control signal for training.
- Limited working area.

Please take a look to this paper:

[An Indoor and Outdoor Positioning Using a Hybrid of Support Vector Machine and Deep Neural Network Algorithms] (https://www.hindawi.com/journals/js/2018/1253752/)

In this paper, we also find an idea how to apply AI to outdoor positioning