ENSTA BRETAGNE

Internship report

SAS Lab, Stevens Institute of Technology

1er octobre 2022

Florian Jerram
florian.jerram@ensta-bretagne.org

Résumé

En tant qu'étudiant en deuxième année d'école d'ingénieur à l'ENSTA Bretagne (Ecole Nationale Supérieure de Technique Avancées), j'ai du effectuer un stage de 14 semaines. Spécialisé en robotique autonome, j'ai ainsi fait le choix logique de prendre part à un projet d'un institut de recherche visant à construire et à programmer un drone de telle sorte qu'il soit totalement autonome. Ce stage à eu lieu dans un des laboratoires de l'université Stevens Institute of Technology à Hoboken, SAS LAb (Safe Autonomous Systems), dans le New Jersey aux Etats-Unis. En tant que premier voyage hors Europe, cette expérience fut aussi enrichissante d'un point de vue professionel que personnel, et c'est ce dont je traite dans ce rapport.

Abstract

As a second year student in the engineering school ENSTA Bretagne (Ecole Nationale Supérieure de Technique Avancées), I had to do a 14 weeks internship. Specialized in autonomous robotics, I logically chose to participate to a project in a research institute aiming at developing a custom-built quadrotor platform for AI and cyber-security research in the Safe Autonomous Systems Lab (SAS Lab). I must effectively create and program a drone so that it can fly by its own, which means that it does not need any human help. This one took part in one of the laboratories of the university Stevens Institute of Technology, in Hoboken, New Jersey, United-States.

Appreciations

I am very grateful to SAS Lab, which allowed me to have the greatest experience I have ever had, and also to the team which I worked with. Without them I would not have gotten this far. Please find below a picture of my appreciated work team.

Figure 1 – SAS Lab's work team

Contents

1 Introduction and context's presentation

The internship I have made was not in a company, it was in a university's laboratory. The university is called Stevens Institute of Technology, and the laboratory's name is SAS Lab (Safe Autonomous Systems). It took place in Hoboken in the New Jersey (United-States). This project was not for the country or the military, but for the laboratory itself, that is why it was a self-paid project.

The goal we aimed at, was developing a custom-built quadrotor platform for AI and cyber-security research in the Safe Autonomous Systems Lab. I had effectively to create and program a drone so that it can fly by its own, which means that it does not need any human help. This project required teamwork. We were effectively 9 students working on it. That is why, in order to not do the same tasks twice, we split the group into 3 smaller groups. To fulfill our mission, we needed effectively to work on the electrical, mechanical, and software field.

During those four months, I had the opportunity to discover, as well as to broaden, my knowledge about robotics, because SAS Lab is indeed a research laboratory specialized in robotics, and especially in drones. Not only have I learned some new abilities, such as the manner to build a drone from nothing, set its parameters in a ground control hardware (such as Mission Planner) and how to estimate its position and orientation thanks to a dozen of cameras, called Vicon Motion Capture. But I also found concepts that I had already seen, such as the manipulation of ROS, the programming in Python and C++, and the image processing thanks to OpenCv. The laboratory has several drones, and a sort of cage, equipped with the Vicon cameras, designed to test our programs in the robots. It also has some 3D printer, which were very useful to mount our drone in the desired way.

2 Problematization of the topic, purpose and stakes

2.1 The topic and purpose

The purpose of the internship was to develop a custom-built quadrotor platform for AI and cyber-security research in the Safe Autonomous Systems Lab (SAS Lab). I had effectively to create and program a drone so that it can fly by its own, which means that it does not need any human help. The plan was to make the drone fly indoors at the beginning, because we had a test cage at our disposal, and outdoors as soon as the drone could perfectly fly indoors.

To fulfill this mission, we were working in a team. This one was composed of 2 coordinators and 8 students. The other students were doing their "summer lessons" in this laboratory, and they were all American students. SAS Lab was funding this project, that is why, to complete our duty, we had plenty of equipment at our disposal. I am particularly thinking of the fact that they have got several NVIDIA cards to build the drone (a Jetson Nano and a Jetson Xavier). By the way, the drone burned once (before I arrived), so they had to buy every component again, and it did not seem to be a massive problem.

However, this work was not as easy as it looks like, especially because of our different backgrounds.

2.2 The stakes

2.2.1 The language barrier

Firstly, this project was representing a particularly important stake, because of the country barrier. I was effectively coming from another country and even from another continent, that is why nobody was certain that we were following the same courses or using the same methods and technics. Thus, I was working with natives and I was the only one being a foreigner. By the way, it was the first time that SAS Lab was working with a foreign intern. We were thus all dreading this situation at the beginning, but as soon as the project began, we all felt very comfortable with it. Although I was not that good in English, everyone was able to understand me and I was also able to understand everyone. At the beginning, as I cannot understand every single word that they were using in their sentences, I just had some difficulties to dissociate the useful content from the useless content. I was however doing my best to do so, and I learned this ability in a very short period of time, because I really needed it, in order to stop asking them to repeat what they just said. As we were a team, we were often communicating about what we did, and how we did it. Moreover, we had a meeting once a week to share with everyone the progress of our work. It was very rewarding for me, because I had to prepare and show some slides with the progress I have made and the topics I wanted to tackle for the next week. This taught me how to organize myself, and how to talk in front of an audience in a foreign language. Regarding all this communication that we had, it was thus really annoying if I had not developed any ability like this. For example, in the same way, another required skill was to understand a sentence or a whole point from understanding just one word or one group of words. It was as useful as the first skill I mentioned, because usually Americans are mincing their words (compared to the British for example).

2.2.2 Comparison our knowledge

Regarding the knowledge that we acquired during our respective lessons, I was surprised that we had quite the same knowledge than the Americans. Indeed, what could be more commonplace than thinking that we were working with different methods because we were working with different components? However, as we are using for example a lot of NVIDIA components in France, and as it is an American brand, at the end of the day, it was not surprising that I had to work with an NVIDIA as a companion computer.

I was however the only one who was able to manipulate ROS. It was very surprising, because at the first meeting, in which we had to explain our plans and how we will handle them, they did not mention ROS. I was thus surprised and asked why they did not want to use ROS, and it is not that they did not want to use ROS, it was just that they did not know this tool. I however took the lead and asked to use ROS during the project because I really wanted to strengthen my ROS skills during this internship, and also because I could not consider a more efficient tool than ROS to exchange data between the different components of the drone. And it was also a clever way to be useful to the team by teaching them how to manipulate ROS, which is very useful in robotics. As I was the only one knowing how to use ROS, I thus had to be the leader in the software part, in the way that I had to explain to everyone what they have to do to make things go further, and how their tasks work and how it is going to help us in our project. This was a very rewarding task, because I never had to lead a part of a project in another language before. I definitely developed my self-confidence in speaking in English as well as in giving directives.

2.2.3 Having someone with a different background on the team

At the beginning of the internship, and even before, I did not really understand why they accepted to take me as an intern. It was indeed a lot of paperwork, and they even paid me for my work. I was also thinking that anybody else could have done the work I performed. However, after my arrival, I realized that this was quite important to have someone in the team who has a different background. Indeed, I had more an external opinion on the subject, and I could propose some other technics or methods that they did not think about it, which could possibly be more efficient (the use of ROS for example). That is why, neither the university nor me have some regrets about it, because in one hand, I brought a lot of knowledge for the team, and in another hand, I really improved my English and robotics skills.

It was really important to me to do my internship in an English-speaking country, because all my scholarship, my first foreign language was German, that is why I can speak above average in this language but that is also why I had a certain backwardness in English. It was a little bit awkward, but as I had a lack of practice in this language, I was a little bit shy to speak in English in front of some other people. But now, this is not the case anymore, because I am not behind anymore and I have got a better accent. It is actually the opposite: I only ask some opportunities to speak English and strengthen my skills to sound like a native !

2.2.4 Professional and personal abilities

Traveling into the United-States is also a very good way to mobilize and enhance my open mind. People from the other side of the earth have effectively a different living-routine. First of all, they are speaking in another language, which is a huge gap. Moreover, they

are not working with the same methods and software than us. Everything is different. I did have to adapt myself to these new ways of living and working. It was however with a real pleasure, because strengthening this ability of adapting myself to a completely new environment could be mobilized in companies. Each company may work with its own methods and 'language', that is why we can associate a company to a little country. I will be thus able to transfer this ability to companies in which I will work, so that I could adapt myself much sooner.

Not only is doing an internship in a foreign country a professional support, but it is also personally rewarding. As it was my first time outside Europe, I was quite intimidated by this whole new system and people, with different backgrounds and culture. For example, Americans are in general not shy at all: they are all speaking very loud and they do not hesitate to take the floor. In a personal way, I am particularly thinking about the sport. One of the most popular sport in the United-States is the basketball, that is why you can find plenty of public courts on the streets. And a very common thing in this country, is to challenge some other players you do not know. In this way, you are playing with and against unknown people. Beside the fact that it is a very good way to meet new people, it is also a very intimidated step to do. Indeed, playing with your friend is way more soft and less stressful than playing with unknown people, because here you do not want to disappoint the people you asked to play with. It was exactly the same process for the volleyball. During my stay in New York, I wanted to do a lot a sports. I thus ran and played basketball and beach volleyball. However, the sport I practiced the most was the beach volleyball. Indeed, there are some courts in Central Park. I went there almost every day. It was my kind of "afterworks". As I made my very best friends on those courts, we can understand that it is very necessary to act as a native and to ask random people to play with and against. It was not that easy at the begining, because it is not common in France to act like that, but as soon as you meet new people, you feel way more confident. That is why, the first time you ask is the hardest. At the end of my internship I knew almost everyone, and I was like a referent because I knew all the specific rules regarding the Central Park's courts. Here is a picture of a few players I met.

Figure 2 – Beach volleyball team

To show that we also had some different backgrounds, let's look at the nationality of those

players. From left to right, we were: a French, a Brazilian, a Nigerian, two Americans, a Russian and a Korean.

In a professional context, it was the same thing for our weekly meeting. For an unknown reason, we only had 40 minutes of meeting. In order to let everyone speak, we had not to monopolize the floor and to speak in a concise way. However if someone was not respecting those rules, we had to interrupt him (politely), and take the floor in order to let everyone knows the work you did and the work you planned for the next few days. That is why I had to do the same thing than the native and to overstep my bounds, otherwise I would have outshined me. Being introverted by nature, I thus came back in France way less shy than I was before.

2.2.5 Physical architecture

During my scholarship and even during my second-year internship, I had to have a global and accurate view of the project, so that I can better know what I will do, and in which order. A useful tool is the physical architecture. This has indeed two main goals: in one hand it allows us to know the system's purpose and requirements, and in a second hand it shows that the system is conceptually realizable. This ability can be utilized everywhere. This method is internationally widespread. At the other side of the earth, I effectively had to deal with it, as you can see in 3. This was very useful for the whole team because we were coming from everywhere, with different backgrounds.

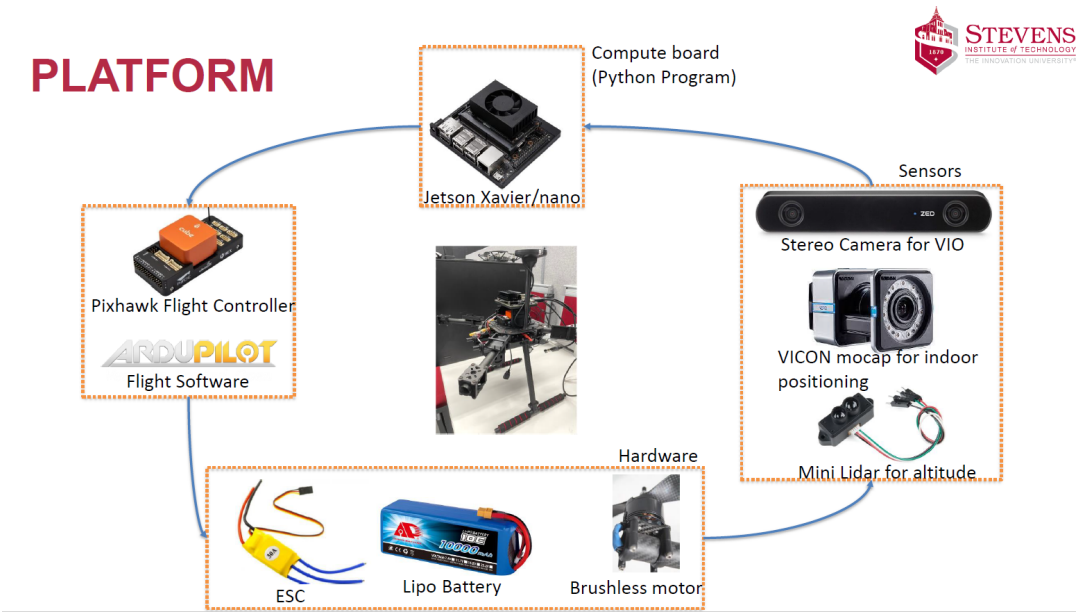

Figure 3 – Physical architecture of the drone

3 Internship's activities and results

3.1 Description of the drone and its environment

As I said before, the mission we had to fulfill was to build and to program a drone in order to make it fly autonomously. But before entering in the details, it is really important to describe our drone and its environment, in order to know what we are talking about.

3.1.1 The drone

We had a lot of equipment at our disposal. I am particularly thinking about the components of the drone. This one was composed of Lipo batteries, motors and ESC, a ZED 2 camera, a Pixhawk and a Jetsnon Nano at the begining and a Jetson Xavier afterwards.

3.1.2 The environment

As we all know, the GPS signals are not efficient at all inside a building, that is why we could not rely on the GNSS to tell the drone where it was in the cage. But we could get round this obstacle thanks to the cameras we installed in the test cage, called Vicon cameras. We could indeed follow the drone's position by following it with all the cameras. To do so, we just had to put some Vicon tracker on it. In order to satisfy this demand, we divided the team in three smaller teams. We thus created the electrical, mechanical and software teams. As you can see in the 4, we were in each team respectively two, four and two.

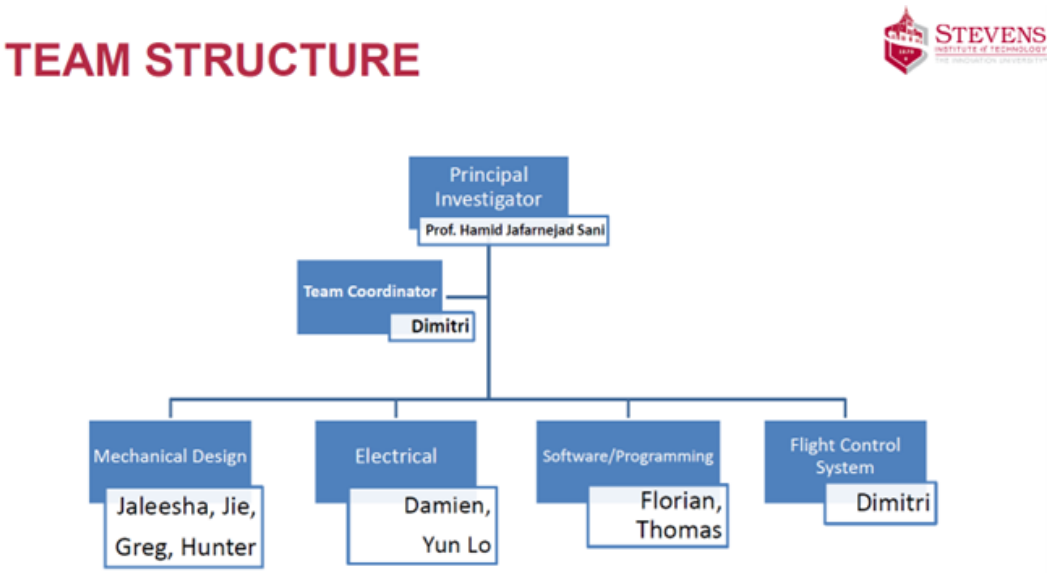

Figure 4 – Team structure of my internship

We chose this configuration considering our backgrounds, but we decided it all together. As it was an important task, we set our aims with seriousness. Beside we split the tasks, we still have to communicate with each other to know what everyone had done, and thus to know how to modify our work in consequences. This is also a way to control the well execution of all the

tasks, beside we also have weekly meetings for that.

Another skill I developed during my internship is to delegate the work. As we are two students working on the drone's software, we had to be very careful not doing the same code twice. That is why we were always communicating and delegating the work to prevent from this waste of time. Although we were in a team, we still had to work with the other teams to know what they did, because a shift in a team might causes a shift in the other teams. This was the main goal of our weekly meetings, in which each team was telling the other what they have done and how. For example, as I was the only one who knew how to use ROS, I was obviously in the software team, but I still helped a little bit in the two other teams. I was however feeling well in the software team, because in our school, we are more specialized in software than in electrical or mechanical, although we have got some lessons about it.

The logical way to show my internship's results is to split them in three parts, one part corresponding to one team.

3.2 Electrical part

The first part was the electrical one. An important thing in order to make the motors spinning, is to associate each motor to an ESC. An Electronic Speed Control (ESC) is an electronic circuit that controls and regulates the speed of an electric motor. After powered up those ESC thanks to the LiPo battery, which is wired to the blue board (which was acting as a power distributor), we had to connect the ESC to the motors, in order to make them work, as you can see in 5.

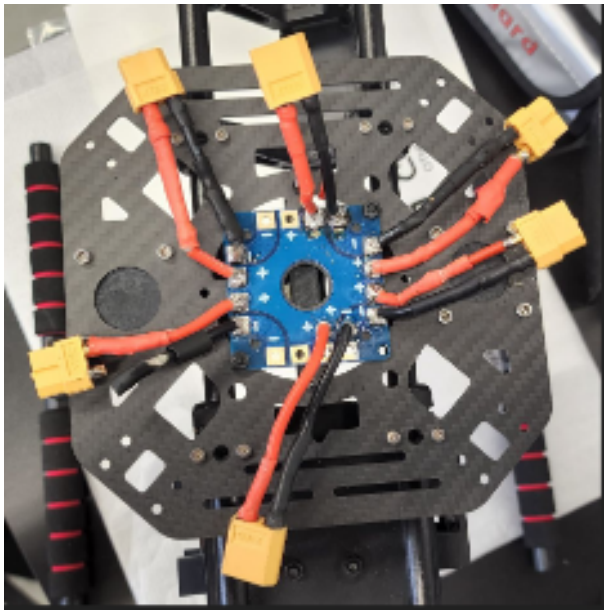

Figure 5 – How the ESC and motors are powered

Afterwards, we calibrated the ESC, otherwise the motors would not have worked. At first, we tried to calibrate them all at once, but it did not work (and we could not figure out why), so we attempted to do a semi automatic calibration, which is a little bit more fastidious because you have to calibrate ESC-by-ESC, but it worked. However, the all

at once calibration is more accurate than the semi automatic one. To do so, we used a Pixhawk and Mission Planner. Pixhawk is an independent open-hardware project providing readily-available, and high-end, autopilot hardware designs to the academic, hobby and industrial communities. And Mission Planner is a full-featured ground station application for the ArduPilot open source autopilot project for Plane, Copter and Rover. It is compatible with Windows only. Mission Planner can be used as a configuration utility or as a dynamic control supplement for autonomous vehicles. I explain a little bit bellow, in 3.5.1, how we configured them.

Moreover, an important fact that we should not forget, is that the motors must not rotate all in the same sens. As we can see on the photo 6, the crossed motors are both going clock wise or both going counter clock wise.

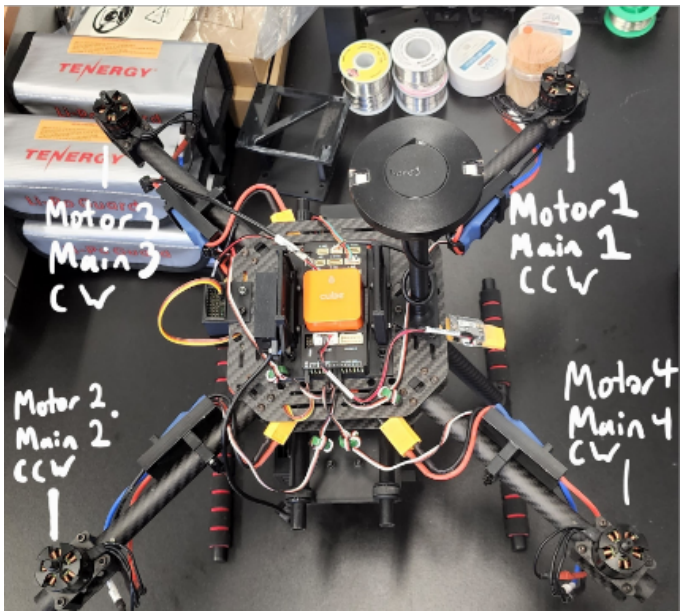

Figure 6 – Motors' spinning sens

Furthermore, to know how high the drone was, we wanted to use a Lidar. This is a detection system which works on the principle of radar, but uses light from a laser. That is why we needed to wire it on the Pixhawk, our flight controller. We wired it thus trough the Serial 4/5 port. We encountered however a problem, because Mission Planner was giving a "bad lidar health" warning, and we did not know why. After thinking that it was because of the source that was powering the lidar, we figured out that the solution was to resolder the wires, but in the correct order, as showed in the figure 7, because they were not. To be sure that we soldered the wires well, we checked the lidar values in Mission Planner under the statues named "rangefiner1" (cm) and "sonarrange" (m).

Another problem we encountered was that one motor was stalling at low throttle speeds. We tried to change which ESC was connected to certain MAIN ports on the Pixhawk, as well as to change the motor, but it did not solve the issue. The issue was in fact due to Mission Planner, in its manner to coded the drone.

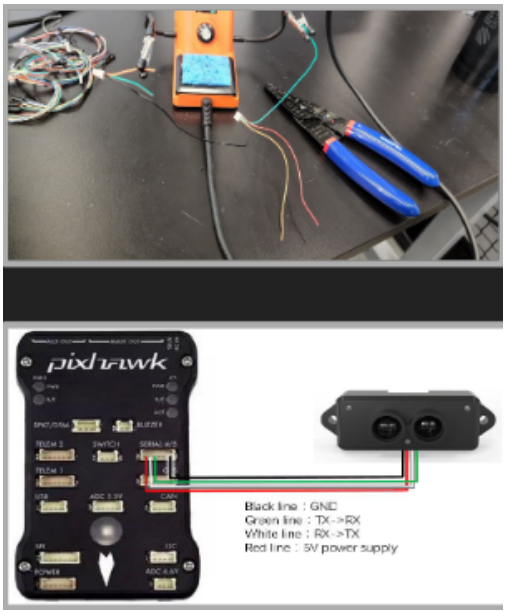

Figure 7 – Rewiring the lidar to the Pixhawk

3.3 Mechanical part

The goal of this team was to 3D print some "mounts" for the drone. We needed indeed one for the camera, because we wanted it to be stabilized. It looked like on the figure 8: on the left side we can see the prototype of this holder, and on the right we have an overview of the real aspect of them, mounted on the drone.

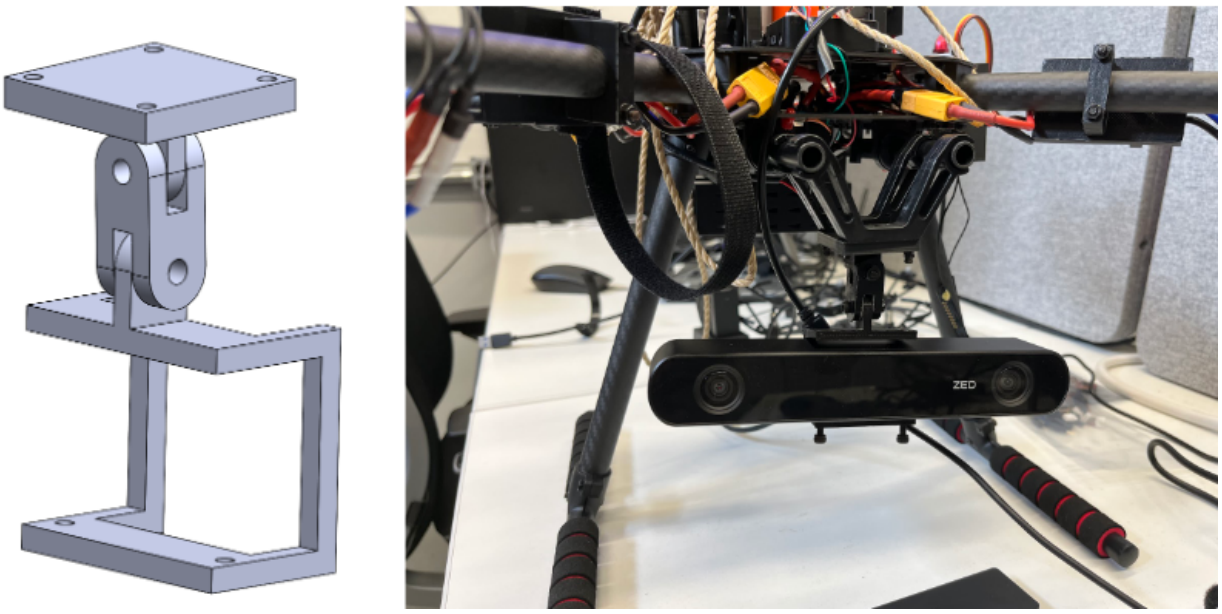

Figure 8 – Camera holder

Furthermore, we also needed one for the Vicon sensors. Vicon stands for Vicon motion system, which is a Motion Capture (MoCap). This is the process of recording the movement of objects or people. Here is a picture showing the protection and test cage with the Vicon

cameras all around above.

Figure 9 – Vicon cameras in the cage

The cameras are able to detect the drone, under the condition that the drone holds the Vicon sensors. As previously mentioned, they look like small white spheres, and they have to be stable on the drone to be as accurate as possible. On this figure 10, we can see on the left the prototype of those tracker holders, and on the right we have an overview of the real aspect of them, mounted on the drone.

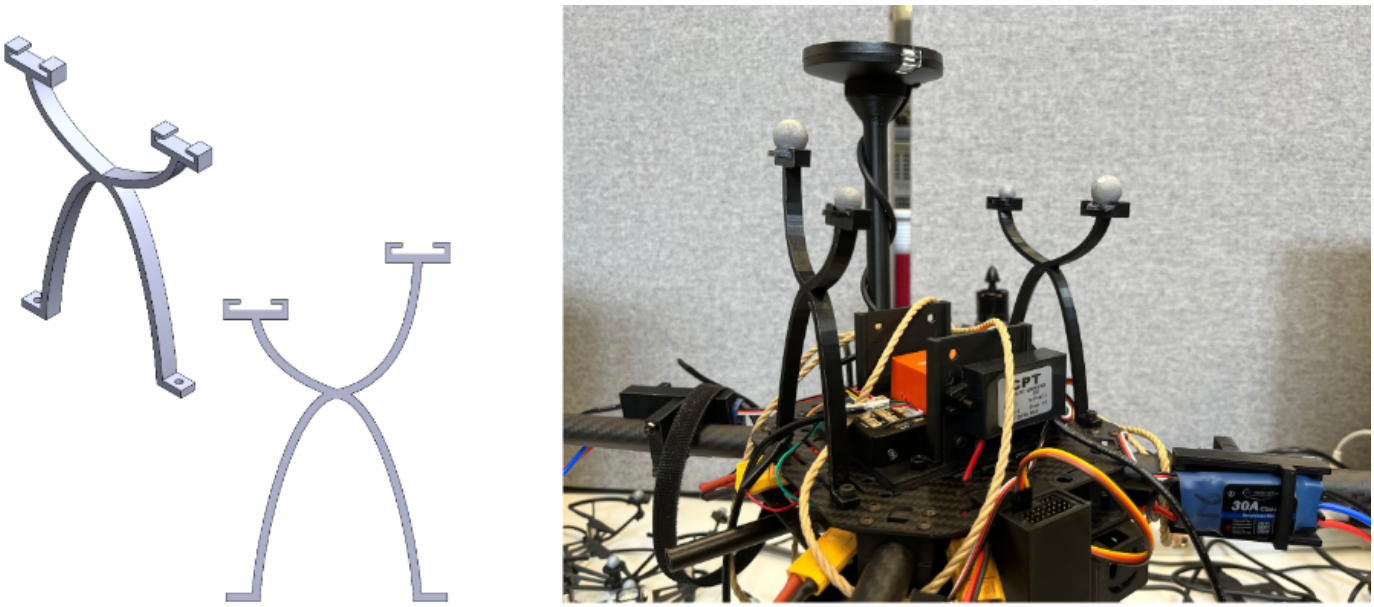

Figure 10 – Vicon tracker holders

All the prototypes of the holders were built thanks to Solidworks, which is a solid modeling

Computer-Aided Design (CAD) and Computer-Aided Engineering (CAE) application. Below is a screenshot of what the final Solidworks project looked like.

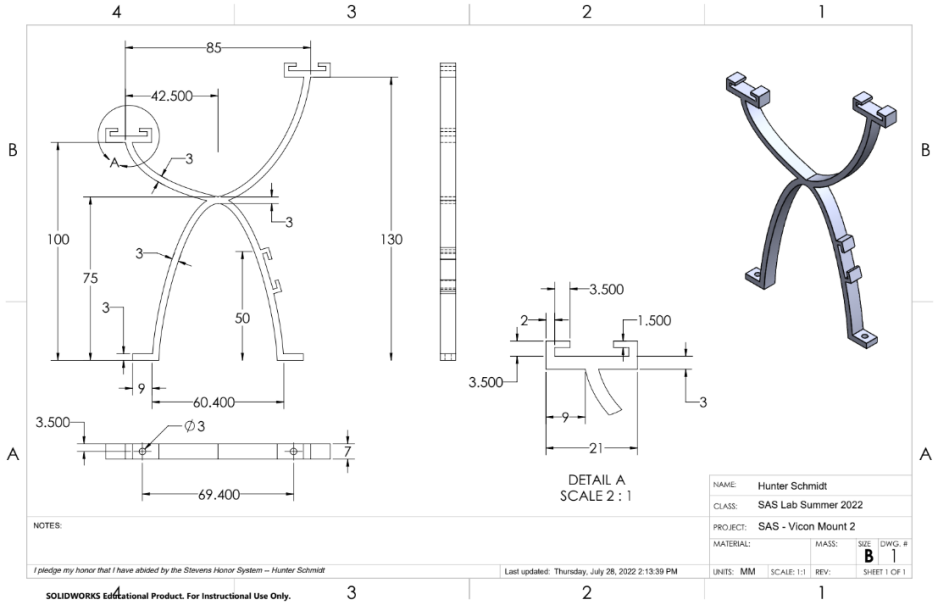

Figure 11 – Solidworks final look of the project

3.4 Software part

3.4.1 Image processing

The goal of this team was probably the most tedious. The first purpose we aimed at, was to focus on the ZED camera. The ZED is a passive stereovision based camera that reproduces the way human vision works. Using its two "eyes" and through triangulation, the ZED understand its surroundings and create a three-dimensional model of the scene it observes. Thanks to the ZED-SDK, we thought that it would be very easy to do some edge recognition or some collision avoidance, because the open source was sharing the scripts. Indeed, the ZED-SDK is well-documented and easy-to-use C++ and Python API for building real-time space-aware applications on desktop and embedded platforms. It was however harder than expected.

Indeed, The ZED-SDK's version that we can download on our Jetson Nano is the jetpack 4.6 version, which is compatible with the Jetson Nano running CUDA 10.2. CUDA (or Compute Unified Device Architecture) is a parallel computing platform and application programming interface (API) that allows software to use certain types of graphics processing units (GPU) for general purpose processing. However, our Jetson Nano is unable to run CUDA 10.2. The only version that our Jetson Nano can run, is CUDA 11.0, which is currently installed on the Nano. The reason is that the architecture of the Jetson Nano ("arm") is not compatible with the architecture that Nvidia proposes for the ZED-SDK ("amd64"). As we were privileging having CUDA downloaded than having the SDK downloaded, the consequences of that, is that we cannot use the ZED-SDK. We thus had to write our own programs in order to do some object recognition or obstacle avoidance.

As we did it, we were able to do some real-time edge recognition thanks to the Canny filter in OpenCv, which can be used for obstacle avoidance for example. OpenCv (Open Source

Computer Vision Library) is an open source computer vision and machine learning software library.Here is how the python files image_publisher.py and image_subscriber.py work.

Thanks to OpenCv, the drone can record its environment in real-time, which allows it to save the frames as OpenCv images. We cannot however send some OpenCv images with the open source ROS (Robot Operating System), that is why we have to convert the OpenCv images into ROS images. Thanks to the OpenCv module named CvBridge, it is very easy to convert them. After converted them, we were now able to send them into the "video_frames" ROS topic. After received those frames in the other node, which is the image processing file, we used the Canny filter. To use it, we had to set the low and the high treshhold. Usually, the high one is between 2 and 3 times as much as the low one. Here, thanks to the experiments, we have estimated a 2.7 rate as the best results rate, and the low treshhold as 60. Thus, the Canny algorithm is selecting the color between 60 and 162 (in the image converted into the gray scale). To be more accurate, all candidate edge pixels below the lower threshold are labeled as non-edges and all pixels above the low threshold that can be connected to any pixel above the high threshold through a chain of edge pixels are labeled as edge pixels. As you can see in the picture 12, this is working quite well.

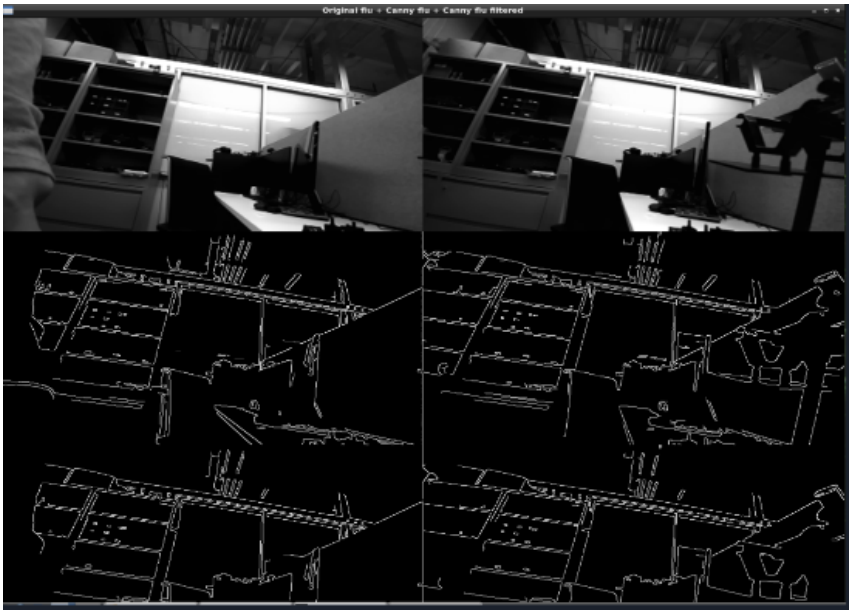

Figure 12 – OpenCv edge recognition

But before going further in the collision avoidance, we first have to make the drone fly autonomously, otherwise the algorithm alone would not be very useful.

3.4.2 Vicon cameras and MavRos

Our second interest is thus to locate the drone thanks to the Vicon cameras, in order to know where the drone is, which will be the steppingstone of the autonomous flight. Here is how it works:

Thanks to ROS, the Jetson Nano is able to receive the Vicon data, which are the drone's position and orientation. The Vicon tracker is effectively sending the Vicon data in the topic named "vicon/SAS_Drone/SAS_Drone". After being able to receive those data, an important thing is to send them from the Jetson to the Pixhawk. Once again, ROS (and more precisely

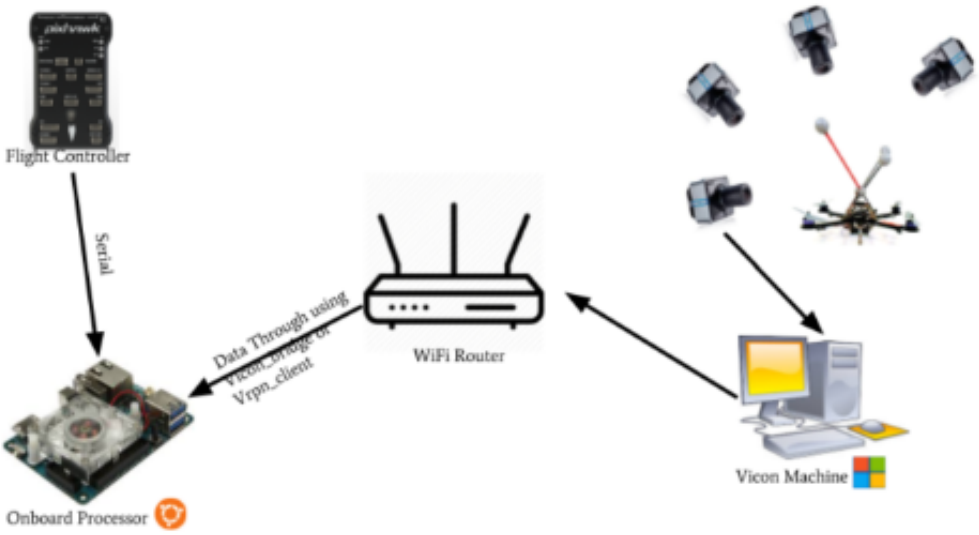

Figure 13 – Communication between the devices

MavRos (and MavRos_extras)) is the perfect tool to use. MavRos is a package that provides communication driver for various autopilots with MAVLink communication protocol. It also provides MAVLink bridge for ground control stations. MAVLink is a very lightweight messaging protocol for communicating with drones (and between onboard drone components). Indeed, publishing the right data in the right topic permits the data to be sent from the Jetson to the Pixhawk. Here, in order to make the bridge between our two devices, we can publish the Vicon data in the “mavros/vision_pose/pose” topic. Here are the results we got. On the picture 14 we can see what we obtained while launching the vicon_bridge file, and on the right, figure 15, we can see the results of the computation we made to estimate the position and orientation of the drone thanks to the Vicon cameras.

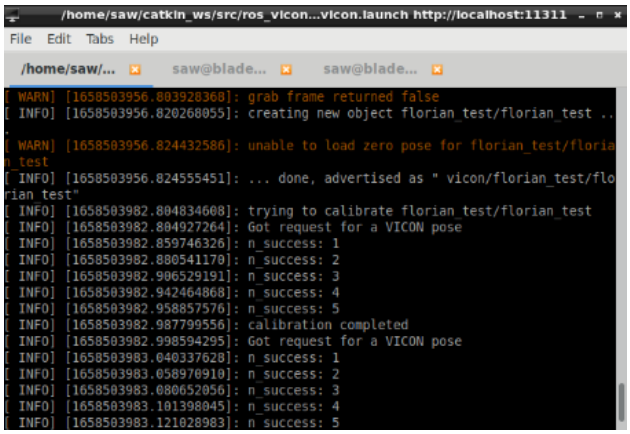

Figure 14 – Vicon bridge

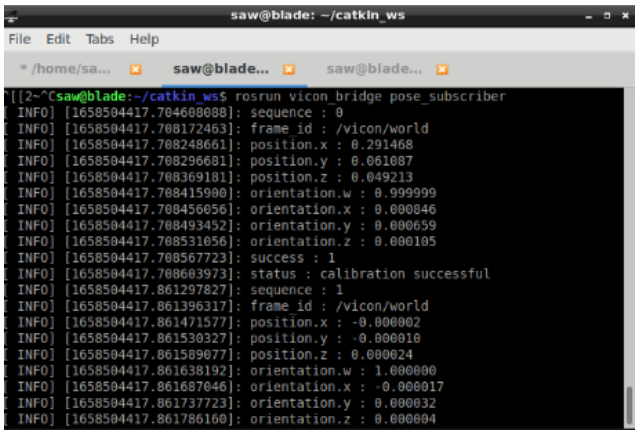

Figure 15 – Drone’s position and orientation

On the first screenshot, we can see that the Vicon Control app created the object to follow, which is called here "florian_test/florian_test", because I was the one who did this. After that, we can see that the following lines show that we tried to calibrate the position of the drone. The first step, which is calibrating the zero, represents the fact that the Jetson Nano is not right

on the ground, but it is here 17cm high. Knowing this data is important for the computation. Indeed, the controller needs this data in order to better know where the drone is and where it should go. However, not only does the controller require the zero of the drone, but for the same reasons it also needs the drone's position and orientation. That is why we can see the "trying to calibrate florian_test/florian_test" and the "Calibration completed" lines. The calibration file leans on calling ROS services, already existing in the "vicon_bridge.cpp" file, which relies on the "tf::Transform.setOrigin" (resp. "tf::Transform.getOrigin") and "tf::Transform.setRotation" (resp. "tf::Transform.getRotation") functions.

Moreover, let's describe the functioning of the ViconToMocap file. This one is creating a node, named ViconToMavros. This node is subscribing to the "vicon/SAS_Drone/SAS_Drone" topic. The received message is a GeometryTransformStamped one, which expresses a transform from the coordinate frame header to the coordinate frame child. After receiving it, this script is processing it. The main goal of this file is to convert the position and orientation contained is this GeometryTransformStamped message, into roll, pitch and yaw data. To do so, we used the quaternions. Indeed spatial rotations in three dimensions can be parametrized using both Euler angles and unit quaternions. To convert the quaternions into Euler angles, we used this formula:

$$\begin{pmatrix} phi \\ theta \\ psi \end{pmatrix} = \begin{pmatrix} \arctan(\frac{2(q_0q_1+q_2q_3)}{1-2(q_1^2+q_2^2)}) \\ \arcsin 2(q_0q_2 - q_1q_3 \\ \arctan(\frac{2(q_0q_3+q_1q_2)}{1-2(q_2^2+q_3^2)}) \end{pmatrix}$$

After translating this formula into a C++ function, we were able to convert the quaternions into Euler angles. This is very useful because we could now store those data into a Geometry-PoseStamped message, which contains the drone's orientation and position, and which will be sent from the Jetson to the Pixhawk thanks to the mavros topic "mavros/vision_pose/pose", as I previously said. Here is the results of the roll, pitch and yaw conversion.

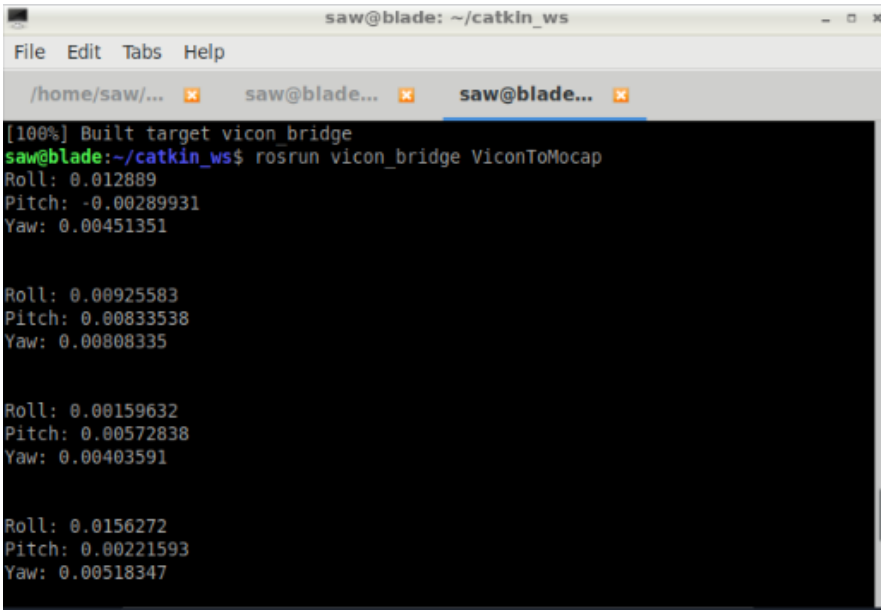

Figure 16 – Drone's roll, pitch and yaw

By the way, the figure 17 below is showing what we can see on the computer running windows. This is a display of the Vicon Control app, the yellow lines representing the links between the

drone's Vicon tracker.

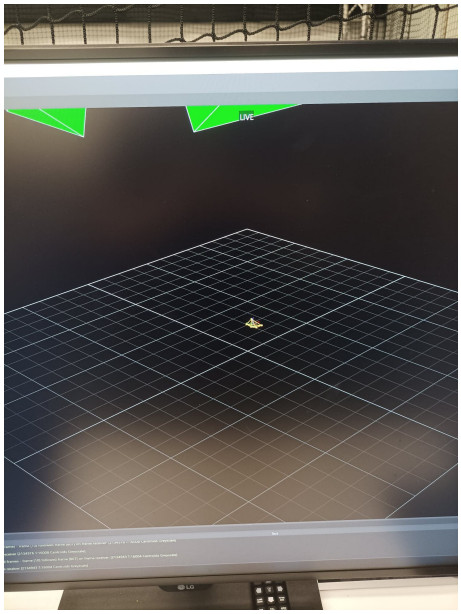

Figure 17 – Display of the Vicon control app

It is however important to notice that we were publishing the Vicon data in the "mavros/vision_pose/pose" topic and not in the "mavros/mocap/pose", because this would send a MavLink command which is not compatible with Ardupilot. To explain that, let's compare the Pixhawk with the Ardupilot.

Px4 has a LPE (Local Position Estimator) and an EKF (Extanded Kalman Filter), while Ardupilot has only an EKF that fills both roles. The local position estimator allows the drone to estimate where it is, and understand its position. However, a separate LPE is only available in Px4. So this MoCap topic, although not labeled as such, is meant to be used with Px4 and not Ardupilot. This is important as data in the ATT_MOCAP_ESTIMATE message is only used by the LPE. The EKF does not support/expect to see this message, so it will ignore it.

That is why, we must interact with the EKF, which is a complex filter that takes in all sensor data and "fuses" them together, determining which sensor is the most reliable and accurate, and will then control the motors speed accordingly. This filter takes all sensor data, and then determines a "guess" as to its current attitude, position and speed. So if the GPS suddenly has bad reception, EKF will switch to the IMU temporarily to prevent a crash. This explains some flight behavior in the lab, as the EKF will prefer GPS data if it has it, and tends to use IMU only when it is unavailable. When running MavRos, the terminal will show updates from various systems, including the EKF. Eventually, the EKF will send a message, explaining that it has swapped between the IMU and GPS. This is the main driver for Ardupilot, and all sensors flow through it (while Px4 has a separate EKF and LPE, Ardupilot only has an EKF that handles everything).

So as we could not use the MoCap plugin, we targeted a different plugin (one that was not "designed" for our Vicon data), which is what we have done with the topic called "mavros/vision_pose/pose".

3.4.3 An important step: a simple take off

Furthermore we also wrote a program which permits the drone to do a 2 meters high take off (entirely vertically), and then to land fully autonomously on the ground. To do so, merging the bridge which is able to receive the Vicon's data with the one which can send those data from the Jetson to the Pixhawk through ROS is really useful. Indeed, thanks to MavRos and MavRos_extras, it is possible to set a position that the drone will reach automatically. That is why we published the desired position in the topic named "mavros/setpoint_position/local", in order to make the drone goes there, thanks to a simple controller (which also uses the drone's current position published in the "mavros/vision_pose/pose" topic). To do so, we entered in the GeometryPoseStamped message the data "msg.pose.position.z = 2", and we let the position and orientation in the other direction to 0. We understand thus well that the drone is moving along the z axis until it reaches 2 meters. Then, we sent a few setpoints before starting, otherwise it will not work. And finally, we set the drone's mode to offboard, in order to make it fly, and the previously mentioned controller will do the work to reach the desired position.

However, the best (and funniest!) thing to do would have been to write our own controller, because we could have made it more complex, at a higher level.

3.4.4 A useful tool: Rosbag

Thanks to the Rosbag tool from ROS, we could record all the messages that have been sent from the different topics during an attempt, in order to play them back or visualize them. As you need neither to launch all the files nor to have your jetson nano plugged with any other device (which includes the fact that the Vicon cameras can be turned off), it is very convenient for the tests, because it spared us a lot of time. Indeed, as we could remove the Jetson Nano from the drone, the mechanical or electrical group could work on it, and meanwhile, the software team could program on the Jetson Nano.

3.5 Mission Planner

Without the ground control Mission Planner, we could not do anything. All the parameters need indeed to be set in this ground control. As it is totally new for me, my opinion is that it is useful to talk about how we set those parameters. We used this tool especially for configuring the ESC, arming the motors and configuring the compass.

3.5.1 Configuring the ESC

In order to configure the ESC, we had a few steps to realize. The first one was to connect to the autopilot from a ground station such as the Mission Planner and set the ESC_CALIBRATION parameter to 3, which is used for the start-up and for automatically calibrate the ESC. Afterwards, we disconnected and reconnected the battery and USB cable so that the autopilot powers down and on. Then, we pressed the safety button of the Pixhawk until it displayed solid red and until we heard a musical tone. The final step was then to disconnect and reconnect the battery again, and test them. To test them the steps were first to ensure the transmitter's flight mode switch was set to "Stabilize Mode", to arm the drone, and then to give a small amount of throttle. All motors should spin at about the same speed and they should start at the same time. If the motors did not all start at the same time and spin at the same speed, it means that the ESC are still not properly calibrated.

However, a steppingstone of this configuration was to arm the drone, and I did not know how to do this. That is why I will talk about it on the next section.

3.5.2 Arming the motors

Arming the vehicle allows the motors to start spinning, otherwise it will not spin. It is quite easy though, because all the pre-arm checks will run automatically and if any problems are found the RGB LED will blink yellow and the failure will be displayed on the ground station. We however needed a remote control for some steps. And then, we had to set the flight mode to Stabilize and to press the safety button until it goes solid red again. The next procedure needed the remote controller, and was to arm the motors by holding the throttle down, and rudder right for about 5 seconds. Once the LED was solid and the propellers were spinning, we could raise the throttle to take-off.

To desarm it, it was even more easy and we also needed the remote controller. First, we held the throttle at minimum and the rudder to the left for 2 seconds, and then pressed the safety button until the LED were flashing.

3.5.3 Configuring the compass

We realized an "Onboard Calibration", which is a calibration routine that runs on the autopilot. To configure the compass we only needed the ground control Mission Planner. Under the setup in mandatory hardware we selected the compass and clicked on the start button for the onboard calibration. Nothing surprising until here. The next step was however to hold the vehicle in the air and rotate it so that each side (front, back, left, right, top and bottom) points down towards the earth for a few seconds in turn, which means a full 360-degree turn with each turn pointing a different direction of the vehicle to the ground. To put it in a nutshell, it resulted in 6 full turns plus possibly some additional time and turns to confirm the calibration. Upon successful completion, three rising tones has been emitted and a "Please reboot the autopilot" window appeared and we rebooted the autopilot in order to be able to arm the vehicle. Otherwise (it could be because of magnetic disturbances from electronics from our pockets for example), we would have heard an "unhappy" failure tone, the green bars might reset, and the calibration routine may restart (depending upon the ground station), and Mission Planner would have automatically retried.

3.5.4 Some additional parameters

Furthermore, we set some other parameters in order to allow the drone to flight indoors, but without GPS. Here is a picture of those parameters set by default above, and set by us for the indoor flight under.

Default			
Parameter	Value	Comment	Description
AHRS_GPS_USE	1	Use GPS for DCM position	GPS Activity
BATT_FS_LOW_ACT	0	None	Low Bat FailSafe
FS_THR_ENABLE	0	Disabled	Throttle FailSafe
FENCE_ENABLE	0		N/A

Indoor Specfic			
Parameter	Value	Comment	Description
AHRS_GPS_USE	0	Disabled	GPS Activiy
BATT_FS_LOW_ACT	0	None (Could Change to Land [1]	Low Bat FailSafe
FS_THR_ENABLE	0	Disabled	Throttle FailSafe
FENCE_ENABLE	0		N/A

Figure 18 – Some additional parameters

4 Undone activities

4.1 Electrical and mechanical parts

As the electrical and mechanical parts were fully finished, we did not have any other task to complete.

4.2 Software part

However, in this team there is still a lot of work to achieve. In deed we stopped at a point where the drone was able to take off and land autonomously. The next step was to test the pre-installed controller in order to follow a path during an unlimited time. It should not be tedious, because we just have to enter a variable that depends of time instead of a time independent variable in the "mavros/setpoint_position/local". For example we could have chosen a constant altitude, but a various position along the x and y axis (as the Lissajous function). To make things go even further, we would have written an extended kalman filter combined to a PID. Thus, we could have followed a path with our own controller. Moreover, we should have implemented a collision avoidance algorithm thanks to the camera in order to prevent from an undesired object, which would be on the drone's path defined in the PID.

5 Conclusion

To put it in a nutshell, I would say that I learned numerous of new abilities during my internship, which are professional as well as personal. In my opinion it is very important to discover new skills in internship, because during our courses in the school, we do not have the time to cover every single point about robotics. It is thus very rewarding in order to explore some other robotics notion, and to know which one we like the most. However, if we already know which branch of robotics we like the most, it is better to find an internship which requires skills in this field, in order to deepen our knowledge. We understand thus that the internship choice is a real stake that anyone should take careful and cleverly. Beyond the professional choice, the personal choice is not something to be trifled with neither. It was indeed a real experience for me to travel outside Europe and to meet all those people I am still in touch with. That is why I would never forget this professional as well as personal rewarding experience, even though I am still a little frustrated that I did not have the time to finish the project.