

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES

PROJET ELECTRONIQUE

RAPPORT FINAL

Un drone au bout des doigts

Système de pilotage de drone à partir des gestes de la main

Auteurs :

BOUDET Arthur
AUSCHER Octave
PEREL Thibault
MALGORN Gaël
CAILLARD Valentin

BUTTIN Domitille
LE CALVEZ Antoine
NIESCIEREWICZ Benjamin
GADON Timothee

16 mai 2016



Remerciements

Nous tenons à remercier et à témoigner toute notre reconnaissance à l'équipe pédagogique du département Systèmes et Réseaux de Communication, pour le soutien et l'aide précieuse qu'elle nous a apportés dans la réalisation de ce projet.

Nous tenons également à la remercier pour avoir accepté d'investir dans le drone, sans lequel rien n'aurait pu aboutir.

Un grand merci à M. Gilles PICOUT pour ses conseils précieux, son investissement dans la commande du drone et dans la réalisation du prototype.

Merci à Mme Fabienne NOUVEL, M. Christophe LEMOINE et M. Yvon DUTERTRE pour leurs remarques qui nous ont permis d'affiner notre projet et ainsi faciliter son élaboration.

Nous remercions également Mme Fanny GOURRET pour son aide dans la planification des tâches de notre projet.

Table des matières

1 Présentation du projet	4
1.1 Introduction	4
1.2 Un drone au bout des doigts ?	5
1.2.1 État de l'art	6
1.2.2 Description du système	8
2 Mains	9
2.1 Reconnaissance des gestes	10
2.2 Transmission des données	13
2.3 Réalisation du prototype	14
2.4 Tests	17
3 Drone	19
3.1 Principe général	19
3.2 Combined Pulse Position Modulation (CPPM)	23
3.3 Contrôleur 328p Multiwii	24
3.4 Récepteur	25
3.5 Réalisation du prototype	27
3.6 Tests	27
4 Bilan	29
4.1 Bilan technique	29
4.1.1 Démonstration finale	29
4.1.2 Améliorations possibles	30
4.2 Gestion du projet	32
4.2.1 Répartition des tâches	32
4.2.2 Gestion du budget	33
4.2.3 Gestion du temps	34
4.2.4 Protection et brevetabilité	36
4.3 Bilan moral	38
A Annexes	42
A.1 Manuel d'utilisation	42
A.2 Poster de présentation	43

Table des figures

1	Drone sous-marin - ROV	5
2	Projet "Danse avec les drones"	6
3	Pilotage à l'aide d'une manette Wiimote	6
4	Manette de Wii	7
5	Multiwii	7
	a Arduino IDE	7
	b Interface logiciel MultiWii	7
6	Schéma de principe du système de pilotage	8
7	Système de pilotage	9
8	Solution 1 pour simuler une télécommande	10
9	Solution 2 pour simuler une télécommande	11
10	Code main non émettrice (main 1)	11
11	Code main émettrice (main 2)	12
12	Module d'émission NRF24L01	13
13	Code de l'émetteur	13
14	Librairies créées pour la réalisation des circuits imprimés	14
15	Routage top et bottom des 2 circuits imprimés	15
16	Boîtier, câble rj9 mâle/mâle et connecteur rj9 femelle	15
17	Batterie et câble utilisés pour l'alimentation	16
18	Prototype final du système de pilotage	16
19	Affichage du moniteur série de l'Arduino de la main 1	17
20	Signal PWM correspondant au throttle	17
21	Affichage du moniteur série de l'Arduino de la main 2	18
22	Schéma basique d'un drone de type quadcopter (4 hélices)	19
23	Schéma de fonctionnement simplifié d'un ESC	20
24	Schéma de branchement d'un ESC	21
25	Schéma d'une télécommande	21
26	Trames PPM/PPM et PWM	22
27	Trame d'un signal CPPM	23
28	Contrôleur MultiWii 328p	24
29	Throttle élevé (gauche) / Throttle faible (droite)	26
30	Routage top du récepteur	27
31	Premiers tests	27
32	Réglage des coefficients PID	28
33	Démonstration finale	29
34	Sonar HC-SR04	30
35	Exemple de répartition des tâches daté du 5 mai 2016	32
36	Liste des composants	33
37	Diagramme de Gantt prévisionnel	35
38	Diagramme de Gantt réel	35

1 Présentation du projet

1.1 Introduction

Dans le cadre du projet d'électronique de 4^{ème} année au sein du département Systèmes et Réseaux de Communication de l'INSA Rennes, notre équipe est constituée des étudiants suivants : Octave AUSCHER, Arthur BOUDET, Domitille BUTTIN, Valentin CAILLARD, Timothée GADON, Antoine LE CALVEZ, Gaël MALGORN, Benjamin NIESCIEREWICZ, Thibault PEREL. En raison de la mobilité internationale induite par la formation INSA, nous précisons que Valentin et Timothée ont été absents lors de la réalisation du projet mais que Gaël nous a rejoint lors de cette phase.

Notre idée pour ce projet est de piloter un drone seulement avec les gestes de la main. Le but est de rendre le pilotage plus ludique, intuitif et plus simple. La main remplace alors la télécommande et le pilotage devient ainsi accessible au plus grand nombre.

Ce projet s'étend sur toute l'année universitaire et s'articule autour de deux parties. Pendant le premier semestre, qui s'écoule de septembre à janvier, nous avons rédigé un cahier des charges de notre projet. Il présentait le plan d'action pour la deuxième partie du projet (deuxième semestre), c'est-à-dire la réalisation du prototype. L'objectif est de valider le concept de fonctionnement avec une démonstration finale le 20 mai 2016.

L'intérêt de ce projet est multiple. Il nous permet d'asseoir les compétences pratiques et théoriques acquises au sein du département SRC, mais aussi de nous auto-former plus spécifiquement sur des aspects techniques. Le but est également de nous former à notre futur métier d'ingénieur. En effet, il s'agit d'un travail en équipe, de la conception à la réalisation sur un sujet technique et innovant tout en respectant des contraintes budgétaires et temporelles.

Ce rapport s'articule autour de quatre grands axes : la présentation du projet, la récupération et l'envoi des données, le drone en lui-même et enfin un bilan technique et moral du projet.

La première partie présente l'état de l'art ainsi que la solution retenue pour notre prototype. Les deux parties suivantes (main, drone) présentent les solutions techniques réalisées. Enfin, dans la dernière partie, nous présenterons un bilan technique de notre prototype et les améliorations possibles de notre projet. De plus, nous aborderons la gestion de notre projet : répartition des tâches, gestion du budget et du temps, protection/brevetabilité de notre solution. Pour finir, nous effectuerons un bilan moral individuel sur la réalisation de ce projet.

1.2 Un drone au bout des doigts ?

Aujourd’hui, les drones sont très utilisés dans les domaines civil et militaire. Ainsi, les sapeurs pompiers de Paris ont testé en 2009 un drone permettant de compter le nombre de personnes présentes sur le lieu d’un incendie, sans avoir à y pénétrer. Depuis 2008, des drones sous-marins ont été créés pour pouvoir effectuer des travaux sous-marins. On les appellent les ROV (Remotely Operated Vehicle). C’est dans ce contexte que nous avons décidé de développer un nouveau système permettant de simuler une télécommande de drone.



FIGURE 1 – Drone sous-marin - ROV

Nous avons profité du projet d’électronique pour un nouveau mode de pilotage de drone. En effet, le pilotage classique nous paraissait peu intuitif.

Une télécommande de drone est composé de deux joysticks. Celui de droite permet à la fois de contrôler le roulis (Roll) et le tangage (Pitch). Le joystick de gauche sert à la gestion des gaz (Throttle) et à faire varier le lacet (Yaw). Nous avons donc pensé qu’il serait plus intuitif d’utiliser les mouvements de la main pour contrôler à la fois le roulis et le tangage pendant que l’autre main servirait à la gestion des gaz. C’est ainsi qu’est née l’idée de notre projet d’électronique.

1.2.1 État de l'art

Projets et systèmes existants

La plupart des projets existants sur le sujet sont des projets universitaires. Au cours de nos recherches, nous avons pu identifier quelques projets intéressants comme le projet "Danse avec les drones"^(Figure 2) développé par des élèves-ingénieurs de l'école Télécom Saint-Étienne.



FIGURE 2 – Projet "Danse avec les drones"

Ce projet permet à un drone de type quadcopter de pouvoir suivre les mouvements d'un danseur. Le système fonctionne grâce à un algorithme de suivi par caméra (en effectuant un traitement d'image). Ceci ne convient pas à notre projet à cause de la latence provoquée par le traitement d'image. Nous avons besoin d'un système réactif, sans aucune latence, pour pouvoir piloter correctement le drone.

Un deuxième projet développé par des étudiants de Polytech' Lille a beaucoup plus retenu notre attention. Il s'agit de piloter un drone à l'aide d'une télécommande de Wii (Wiimote)^(Figure 3). Ce projet présente de nombreuses similarités avec le nôtre. Il s'agit de récupérer les données du gyroscope de la manette, les interpréter et diriger le drone en conséquence. La limite principale de ce projet est la connectivité entre la manette et le drone, qui doit s'effectuer par l'intermédiaire d'un Smartphone Android connecté en Bluetooth.



FIGURE 3 – Pilotage à l'aide d'une manette Wiimote

Pour notre projet, nous souhaitons avoir une connexion directe entre les mains et le drone, encore une fois pour éviter toute latence inutile et éviter les intermédiaires entre le système de pilotage et le drone.

MultiWii

« MultiWii » est un projet arduino conçu pour contrôler des multicoptères (tels que des quadcoptères, hexacoptères, avions et même des hélicoptères). A l'origine, il a été développé pour les manettes portatives *Nintendo Wii*^(Figure 4).

Ce projet a évolué et dispose désormais d'un grand nombre d'options supplémentaires. Le code a également été adapté pour fonctionner avec la plupart des capteurs. Il fonctionne notamment avec le *MPU6050* qui dispose d'un gyroscope à trois axes ainsi que d'un accéléromètre à trois axes.

Un des éléments essentiels à noter est que le MultiWii est un projet Open-Source personnalisable. Il doit donc être configuré en fonction du matériel à utiliser et de l'utilisation prévue. Tout cela se fait au sein de l'environnement de programmation *Arduino IDE*^(Figure 5a).

La modification du code source, tout en assurant le support de la stabilisation et l'asservissement des moteurs, est gérée directement par le programme. C'est la raison pour laquelle nous nous sommes penchés sur cette solution car nous pouvons adapter le programme à nos besoins.

Le programme est également fourni avec un petit logiciel permettant le réglage de différents paramètres : armement du drone et de ses moteurs, réglages des seuils concernant les communications avec la télécommande (ou ici de notre système de pilotage)^(Figure 5b)... Il nous permet également de visualiser en temps réel les différentes entrées de notre système de pilotage, ce qui nous est très utile pour les tests.



FIGURE 4 – Manette de Wii

(a) Arduino IDE



(b) Interface logiciel MultiWii

FIGURE 5 – Multiwii

1.2.2 Description du système

Notre projet se découpe en 2 grandes parties : la reconnaissance des mouvements de la main et la réalisation d'un drone.

Le drone est constitué pour la partie électronique d'un module récepteur et d'un contrôleur de vol. Concernant la partie mécanique, il est constitué d'un cadre et de 4 moteurs.

La reconnaissance des gestes a pour but d'émuler une télécommande. Elle se fait par l'intermédiaire de 2 gyroscopes. Chaque gyroscope est relié à un Arduino. Les Arduinos peuvent interpréter directement les angles des mains. Ces angles sont ensuite centralisés, traités puis convertis par un programme se trouvant dans l'Arduino d'une des 2 mains, pour enfin les envoyer au drone via un module émetteur fonctionnant à 2,4 GHz.

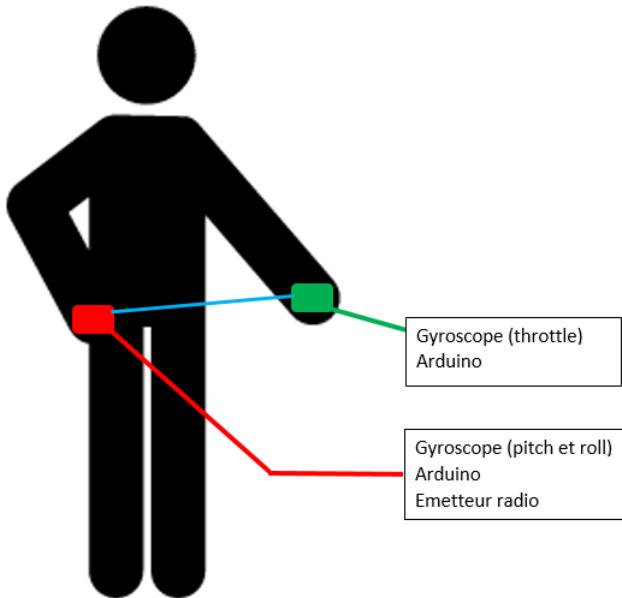


FIGURE 6 – Schéma de principe du système de pilotage

La liaison entre les 2 mains se fait par l'intermédiaire d'un câble RJ9, câble très répandu car utilisé sur les combinés téléphoniques filaires.

Comme sur la plupart des télécommandes, aucun retour d'information n'est mis en place. Nous avons donc une seule liaison, allant des mains vers le drone.

2 Mains

Cette partie est essentielle car c'est la réelle innovation du projet : piloter un drone avec le mouvement des mains. L'objectif est ici de présenter le fonctionnement du système de pilotage tout en mentionnant les difficultés que nous avons rencontrées et les modifications par rapport au cahier des charges. Ainsi, nous aborderons successivement les aspects conception, tests et réalisation du prototype.

La figure 7 rappelle le principe de fonctionnement du système de pilotage.

Le throttle contrôle les gaz. Le roll contrôle les mouvements gauche et droite. Le pitch contrôle les mouvements avant et arrière. Le yaw n'est pas utilisé car on se placera en mode « Headless ». Ce dernier permet au drone de garder le cap initial. Le système est compatible aussi bien pour les gauchers que pour les droitiers. Pour faciliter la compréhension, la main qui gère le throttle sera notée "main 1" et la main qui gère le pitch et le roll sera notée "main 2".

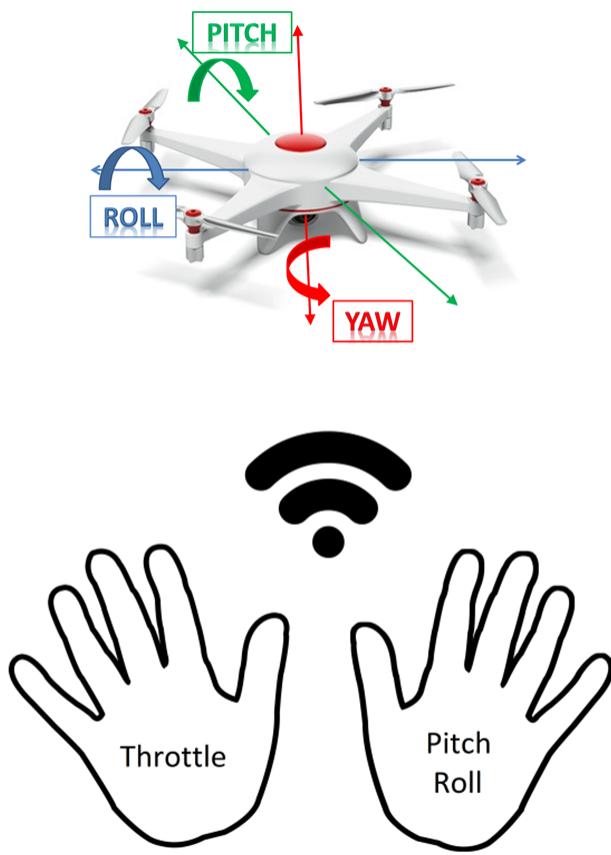


FIGURE 7 – Système de pilotage

2.1 Reconnaissance des gestes

Pour piloter le drone, il faut lui transmettre des valeurs de pitch, roll et throttle. Pour récupérer ces valeurs, la solution que nous avons envisagée est d'utiliser deux capteurs (gyroscopes) ainsi qu'un microcontrôleur (Arduino) : les gyroscopes permettent de mesurer les angles des mains et l'Arduino récupère et traite les données.

Voici le schéma initial envisagé :

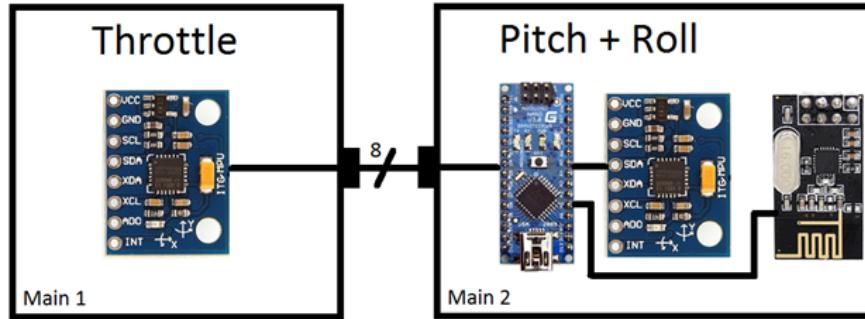


FIGURE 8 – Solution 1 pour simuler une télécommande

Le gyroscope que nous utilisons est le MPU6050. Ce capteur fonctionne en interruption et possède un DMP (digital motion processing) qui permet la conversion de données brutes en valeurs d'angles usuels. Il est normalement possible d'utiliser 2 MPU6050 avec un seul Arduino en utilisant la sortie « ADD ». En mettant la sortie ADD d'un capteur à la masse et celle de l'autre à VCC, il est théoriquement possible d'accéder aux valeurs mesurées par les 2 MPU6050. Cependant, nous avons rencontré des difficultés pour adapter le code initialement prévu pour un seul capteur. Il a donc fallu envisager une autre solution : utiliser un microcontrôleur par gyroscope.

L'objectif est d'envoyer le triplet throttle/pitch/roll au drone à l'aide d'un même module d'émission que nous présenterons par la suite. Ainsi, un seul Arduino est utilisé en émetteur et il doit avoir connaissance des 3 angles correspondant au triplet. Il faut alors faire communiquer les 2 Arduinos. Pour des raisons de simplicité, nous avons choisi l'Arduino de la main 2 en tant qu'émetteur. La communication entre l'Arduino de la main non émettrice et celui de la main émettrice s'effectue à l'aide d'un signal PWM dont le duty cycle correspond à la valeur du throttle. Le throttle correspond en fait au pitch de la main non émettrice, ce qui permet un fonctionnement similaire à une pédale d'accélération.

Les deux microcontrôleurs sont alimentés par la même batterie. 3 fils sont donc nécessaires pour relier les 2 mains (VCC, masse, PWM correspondant au throttle), contrairement à 8 fils dans la première solution. Le schéma modifié permettant de faire fonctionner 2 MPU6050 est présenté figure 9.

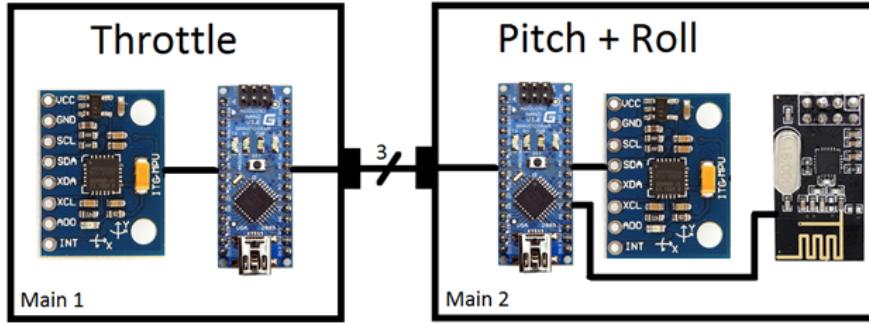


FIGURE 9 – Solution 2 pour simuler une télécommande

La modification de notre schéma de principe initial implique l'utilisation d'un Arduino supplémentaire ainsi que la mise en place d'une communication entre les Arduinos. En revanche, cela nous a permis de réduire le nombre de fils nécessaires entre les 2 mains.

La figure 10 présente une version simplifiée du code de la main non émettrice.

```
#include "MPU6050_6Axis_MotionApps20.h" // Inclusion de la librairie MPU6050

MPU6050 mpu; // Déclaration d'une instance de la classe MPU6050
float ypr[3]; // Tableau contenant les valeurs : [yaw, pitch, roll]

void setup() {
    mpu.initialize(); // Initialisation du MPU
    Serial.println(mpu.testConnection() ? F("MPU6050 connection successful") : F("MPU6050 connection failed")); // Test de connexion
    mpu.dmpInitialize(); // Initialisation du DMP (Digital Motion Processor)
}

void loop() {
    mpu.dmpGetYawPitchRoll(ypr, &q, &gravity); // Récupération des valeurs [yaw, pitch, roll] dans le tableau ypr
    analogWrite(pin, map(ypr[2] * 180/M_PI, 90,-90,0,255)); // Transmission de l'angle Roll à la main émettrice
    // (signal PWM correspondant au throttle)
}
```

FIGURE 10 – Code main non émettrice (main 1)

La fonction dmpYawPitchRoll nous permet de récupérer les valeurs brutes des angles de la main gauche et de les stocker dans un tableau de flottants appelé ypr. "mpu" est une instance de la classe MPU6050 dont le fonctionnement est transparent pour nous. Cette instance nous permet d'interagir avec le composant. Par la suite, nous effectuons la conversion des valeurs issues du pitch uniquement en valeurs d'angles usuels que nous envoyons sous forme de signal PWM vers « pin », c'est à dire la broche numéro 10 de l'arduino. En effet, nous voulons que la main 1 agisse comme une pédale d'accélération, avec une position haute ($\theta = 90^\circ$ et rapport cyclique de 0 %) et une position basse ($\theta = -90^\circ$ et rapport cyclique de 100%).

La figure 11 présente une version simplifiée du code de la main émettrice permettant de récupérer tous les angles nécessaires. La partie du code en lien avec l'émission sera présentée par la suite.

```

#include "MPU6050_6Axis_MotionApps20.h" // Inclusion de la librairie MPU6050

MPU6050 mpu; // Déclaration d'une instance de la classe MPU6050
float ypr[3]; // Tableau contenant les valeurs : [yaw, pitch, roll]

void setup() {
    mpu.initialize(); //Initialisation du MPU
    Serial.println(mpu.testConnection() ? F("MPU6050 connection successful") : F("MPU6050 connection failed")); // Test de connexion
    mpu.dmpInitialize(); //Initialisation du DMP (Digital Motion Processor)
}

void loop() {
    mpu.dmpGetYawPitchRoll(ypr, &q, &gravity); // Récupération des valeurs [yaw, pitch, roll] dans le tableau ypr
    thr = map(pulseIn(sortie, HIGH),0,2000,-90,90); // Récupération de la valeur de throttle envoyée par la main non émettrice
}

```

FIGURE 11 – Code main émettrice (main 2)

L'Arduino de la main 2 récupère les valeurs du pitch et du roll du MPU6050 qui lui est associé. Il récupère également la valeur du throttle sur « sortie », c'est à dire la broche n°9, grâce au duty cycle du signal PWM émis par la main 1. La largeur de l'impulsion du signal reçu est une valeur entre $0\mu s$ (position haute de la main gauche, rapport cyclique de 0%) et $2000\mu s$ (position basse de la main gauche, rapport cyclique de 100%). Nous voulons que ces valeurs soient de nouveau des valeurs d'angles, c'est pourquoi nous utilisons la fonction map qui se charge d'effectuer la correspondance entre les deux. A présent, l'Arduino émetteur a connaissance du triplet throttle/pitch/roll. A ce stade, nous récupérons donc toutes les données nécessaires au pilotage du drone. Il faut maintenant aborder la partie transmission des données au drone.

2.2 Transmission des données

Lorsque les 3 angles qui correspondent au triplet throttle/pitch/roll sont récupérés par l'Arduino émetteur, il faut les transmettre au drone. Le module que nous avons choisi est le transceiver NRF24L01 (Figure 12) en raison de sa faible consommation, son prix réduit et sa grande utilisation au sein de la communauté Arduino. Celui-ci a une fréquence de fonctionnement de 2.4GHz et utilise le protocole ESB (Enhanced Shockburst).

Il est important de noter que la communication entre les mains et le drone est unidirectionnelle. Ainsi, le module NRF24L01 situé sur la main 2 fonctionnera uniquement en tant d'émetteur. Rappelons par ailleurs que le fonctionnement du module (protocole) est totalement transparent pour nous. Nous allons uniquement lui fournir le triplet à émettre. Le module de réception situé sur le drone sera présenté ultérieurement.

La figure 13 présente une version simplifiée du code de l'émetteur.

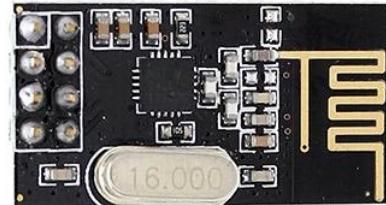


FIGURE 12 – Module d'émission NRF24L01

```
char data[12];
char buf[12];

pitch = map(pitch, -90, 90, 100, 280);
roll = map(roll, -90, 90, 100, 280);
thr = map(thr, -90, 90, 100, 280);

itoa(thr,buf,10);
data[0] = buf[0];
data[1] = buf[1];
data[2] = buf[2];

Mirf.send((byte *) &data);
```

FIGURE 13 – Code de l'émetteur

De la même manière que précédemment, nous interagissons avec le module d'émission grâce à une instance de la classe Mirf. La méthode send() prend comme paramètre un tableau de caractère, ce qui impose une conversion préalable des valeurs d'angles. La fonction itoa() convertit une valeur entière en chaîne de caractère mais ne gère pas les nombres négatifs. Pour remédier à ce problème, nous avons décidé d'appliquer un offset aux valeurs d'angles afin que celles-ci soient positives, quelle que soit l'inclinaison des mains. Par soucis de clarté, nous présentons une version simplifiée du code, où thr est la seule valeur traitée. Il faut imaginer que l'opération de conversion est effectuée de la même manière pour le pitch et pour le roll.

2.3 Réalisation du prototype

Les premiers tests étaient effectués avec des breadboards et des fils à breadboard. Cela était suffisant pour les aspects fonctionnels mais l'objectif était de réaliser un prototype plus robuste, compact et ergonomique pour la démonstration. Pour cela, nous avons réalisé des circuits imprimés que l'on a intégrés dans des boîtiers en plastique. De plus, nous avons travaillé sur l'optimisation de la liaison physique entre les 2 mains.

Réalisation des cartes électroniques

Une fois les tests fonctionnels validés, nous avons entrepris la réalisation de circuits imprimés. Il s'agissait de router deux cartes électroniques (une pour chaque main) de tailles réduites. Pour cela, nous avons réalisé les librairies^(Figure 14) des composants nécessaires sous Pads.

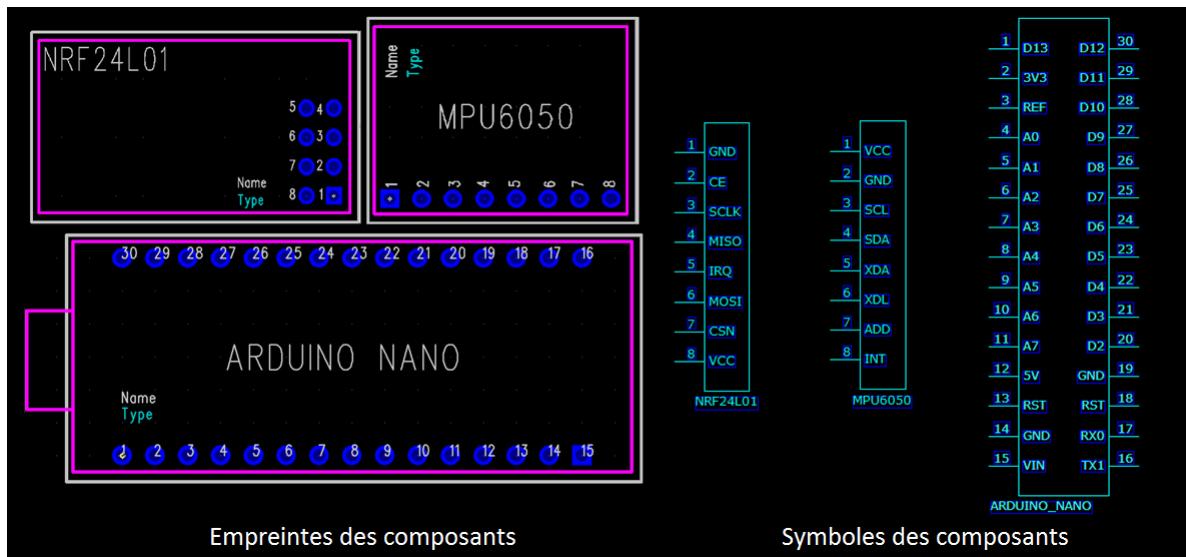


FIGURE 14 – Librairies créées pour la réalisation des circuits imprimés

L'étape suivante a été de réaliser les schémas puis le routage. La figure 15 présente les rendus finaux des 2 circuits imprimés (Top et Bottom).

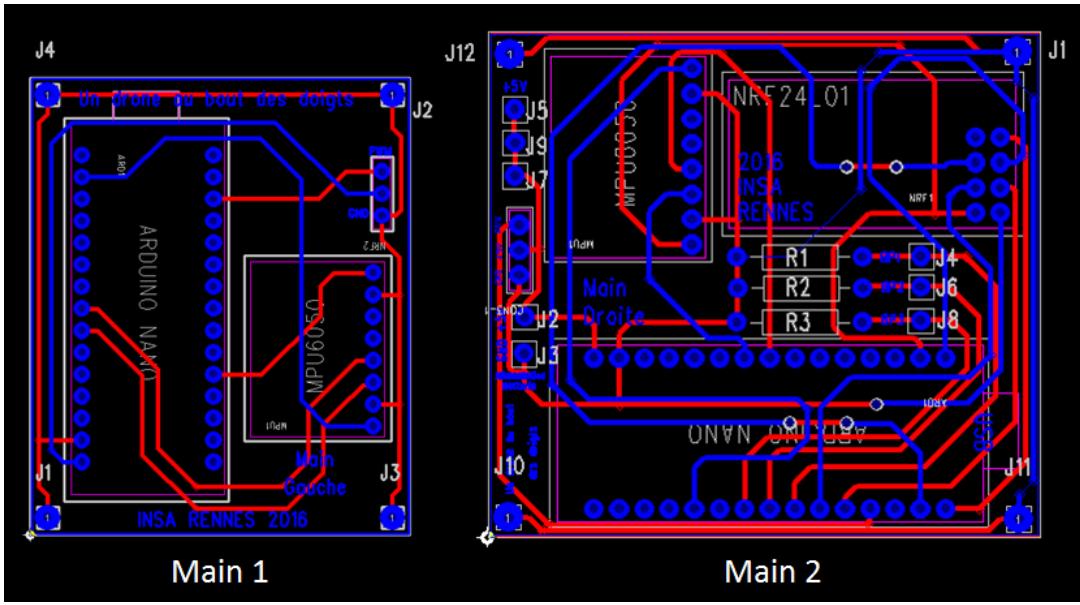


FIGURE 15 – Routage top et bottom des 2 circuits imprimés

Il faut noter que certaines soudures ont été difficiles à réaliser car nous avons routé en bottom et en top alors que les technologies disponibles ne permettaient pas de souder des éléments en top. Il aurait donc fallu router la totalité en bottom.

Liaison entre les 2 mains et mise en boîte

Une fois les circuits imprimés réalisés et les composants soudés, nous avons pu tester le prototype. Son bon fonctionnement nous a conduit à travailler sur l'intégration de ces cartes dans des boîtiers afin d'améliorer la robustesse et l'ergonomie du produit final. Cette tâche a été réalisée en parallèle de la recherche d'un câble pour lier les deux mains. Les boîtiers doivent être de tailles réduites pour s'intégrer facilement sur les mains. Concernant le câble, il doit garantir une liberté de mouvements sans risquer le débranchement de celui-ci. Le boîtier "Atlantic composants", le câble téléphonique à spirale rj9 (4 fils) mâle-mâle ainsi que le connecteur rj9 femelle associé sont présentés figure 16. La spirale garantit des mouvements amples sans problèmes. Les connecteurs sont intégrés dans les boîtiers.



FIGURE 16 – Boîtier, câble rj9 mâle/mâle et connecteur rj9 femelle

Alimentation

Pour alimenter les 2 microcontrôleurs, il suffit d'en alimenter un seul car les 2 alimentations des Arduinos sont reliées par le câble rj9. Une batterie externe 5V et un câble usb mini^(Figure 17) sont suffisants pour notre prototype.



FIGURE 17 – Batterie et câble utilisés pour l'alimentation

Rendu final

Voici le prototype final du système de pilotage (sans la batterie) présenté lors de la démonstration :



FIGURE 18 – Prototype final du système de pilotage

2.4 Tests

Cette partie a pour objectif de présenter les différents tests que nous avons effectués. Ces tests nous ont permis de valider le fonctionnement de notre système de pilotage. Les tests concernant la transmission des données seront détaillés dans la partie consacrée au drone.

Acquisition du throttle

Tout d'abord, on s'intéresse à la récupération du throttle qui correspond à l'angle pitch de la main 1. La figure 19 présente le moniteur série de l'Arduino de la main 1. On affiche les 3 angles yaw/pitch/roll mais seul le pitch nous intéresse pour la main 1.

```

COM3
Envoyer
Initializing I2C devices...
Testing device connections...
MPU6050 connection successful
Initializing DMP...
Enabling DMP...
Enabling interrupt detection (Arduino external interrupt 0)...
DMP ready! Waiting for first interrupt...
ypr    -4.85  19.64  25.19
ypr    -4.83  19.58  25.22
ypr    -4.80  19.52  25.27
ypr    -4.78  19.46  25.32
ypr    -4.74  19.39  25.36
ypr    -4.71  19.34  25.41
ypr    -4.67  19.30  25.44

```

Défilement automatique Pas de fin de ligne 115200 baud

FIGURE 19 – Affichage du moniteur série de l'Arduino de la main 1

Signal PWM

Une fois que l'Arduino de la main 1 a récupéré l'angle pitch correspondant au throttle, il faut transmettre cette valeur à l'Arduino de la main 2. Pour rappel, cette transmission s'effectue avec un signal PWM dont le duty cycle est modulé par la valeur du throttle. La figure 20 présente les oscillosogrammes correspondants à 3 situations : throttle minimum, throttle intermédiaire et throttle maximum.

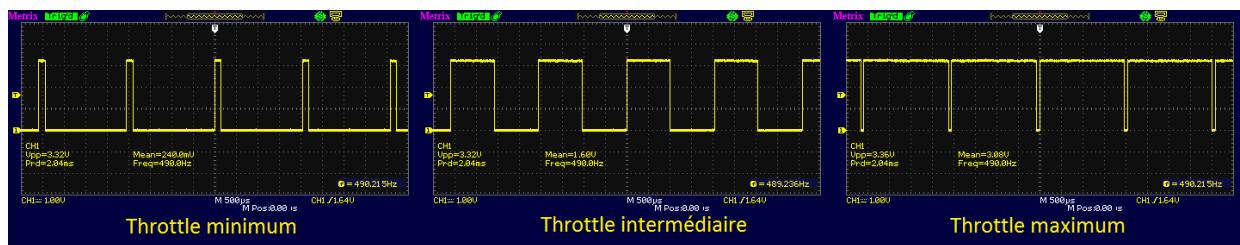
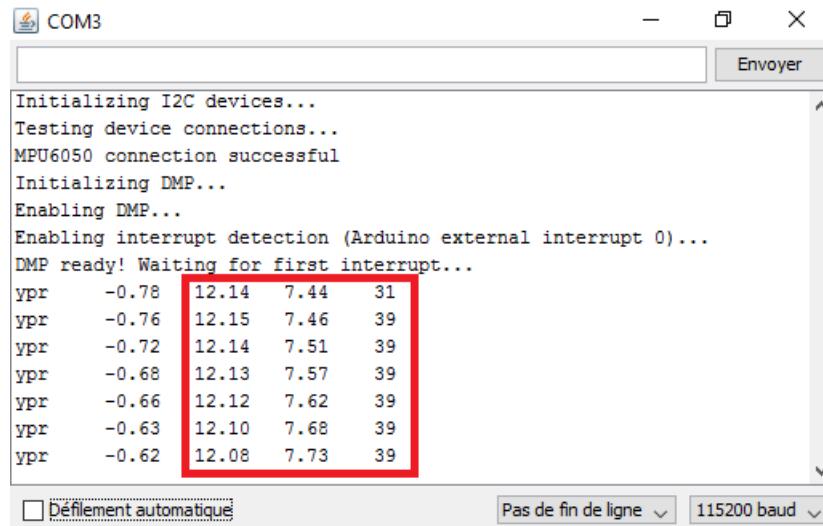


FIGURE 20 – Signal PWM correspondant au throttle

Acquisition des triplets pitch/roll/throttle

On cherche à valider la récupération des 3 valeurs pitch/roll/throttle de la main émettrice. La figure 21 présente l'affichage du moniteur série de l'Arduino de la main 2.



The screenshot shows the Arduino Serial Monitor window titled "COM3". The window displays the following text:

```
Initializing I2C devices...
Testing device connections...
MPU6050 connection successful
Initializing DMP...
Enabling DMP...
Enabling interrupt detection (Arduino external interrupt 0)...
DMP ready! Waiting for first interrupt...
```

Below this, there is a table of data:

ypr	-0.78	12.14	7.44	31
ypr	-0.76	12.15	7.46	39
ypr	-0.72	12.14	7.51	39
ypr	-0.68	12.13	7.57	39
ypr	-0.66	12.12	7.62	39
ypr	-0.63	12.10	7.68	39
ypr	-0.62	12.08	7.73	39

At the bottom of the monitor window, there are three buttons: "Défilement automatique" (unchecked), "Pas de fin de ligne" (selected), and "115200 baud".

FIGURE 21 – Affichage du moniteur série de l'Arduino de la main 2

3 Drone

3.1 Principe général

Dans cette partie, nous allons expliquer le fonctionnement du drone utilisé dans le cadre de ce projet. Il est important de noter que ce qui va suivre est valable pour tout type de drone, peu importe sa configuration initiale (quadcopter, tricopter, hexacopter, etc...). Voici le schéma de fonctionnement du drone :

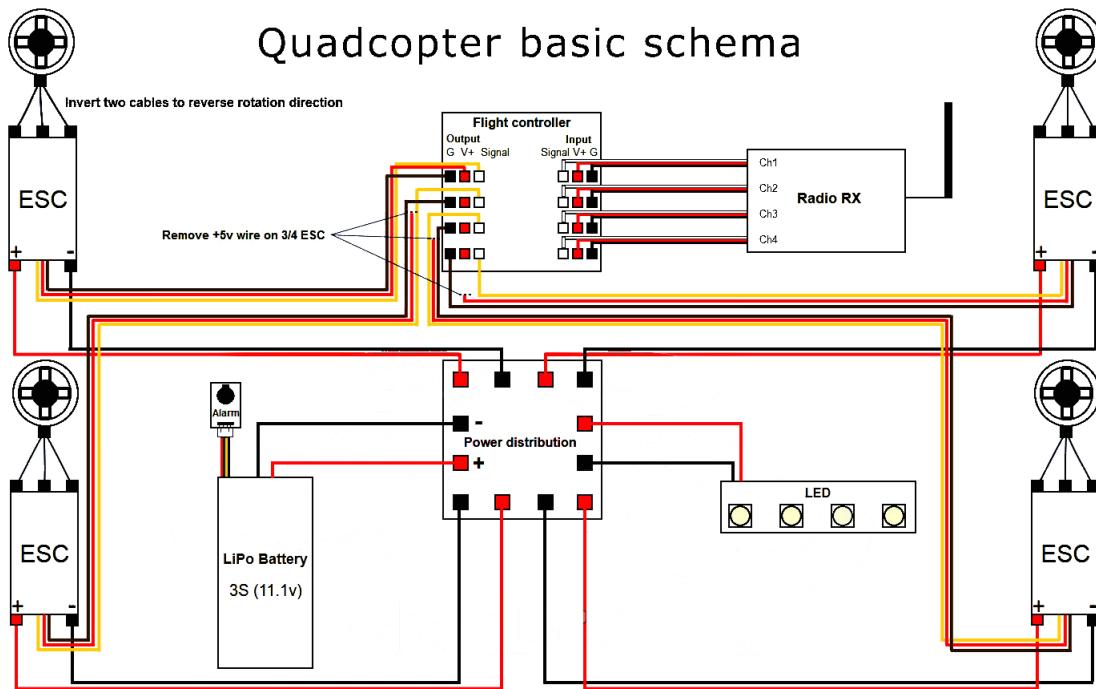


FIGURE 22 – Schéma basique d'un drone de type quadcopter (4 hélices)

Dans un premier temps il est important de présenter les composants indispensables au bon fonctionnement du quadcopter :

Flight Controller (FC)

Le contrôleur de vol (ou Flight Controller) est l'élément le plus important de l'architecture du drone. Il agit comme un cerveau et exécute l'algorithme qui va équilibrer le drone en temps réel. Il reçoit des commandes issues du récepteur et envoie les informations de stabilisation aux moteurs par le biais des ESCs. Il en existe de toute forme pour différentes utilisations : cinéma, courses de drones ou autre. Dans la plupart des cas il s'agit d'une carte de type Arduino (généralement équipé d'un microcontrôleur ATmega328) avec plusieurs capteurs intégrés (accéléromètre, gyroscope, magnétomètre, baromètre, sonar, GPS, etc). Nous détaillerons dans la partie suivante le FC utilisé dans le cadre du projet qui repose sur une carte de type Arduino également.

La batterie LIPO (LIthium POlymer)

C'est un élément qui peut sembler futile mais qui est en réalité très important pour le bon fonctionnement du drone.

En effet il faut une batterie avec un très fort courant de décharge pour pouvoir alimenter les 4 moteurs qui peuvent tirer jusqu'à 20A chacun. Ces batteries sont caractérisées par leur coefficient de décharge C . Il suffit ensuite de multiplier ce coefficient par sa capacité (qui se mesure en mAh , milliampère-heure) pour obtenir le courant maximum que peut débiter la batterie. A titre d'exemple une batterie de 2000mAh avec un coefficient C de 45 pourra débiter au maximum un courant de 90A. Une batterie LIPO est également caractérisée par son nombre de cellules, chacune d'entre elle possédant une tension nominale de 3,7V. Dans le cadre du projet nous disposons d'une batterie à 3 cellules (fournissant ainsi une tension nominale de 11.1V) et d'une capacité de 1000mAh.

Electronic Speed Controller (ESC)

Il est évident que dans le cadre du projet, le procédé qui suit est totalement transparent et intégralement géré par le FC et son algorithme de stabilisation. Il est néanmoins intéressant de comprendre son fonctionnement.

Les ESCs (Electronic Speed Controller) permettent de faire la liaison entre les moteurs et les commandes issues du FC. Pour faire simple, on peut assimiler un ESC au circuit suivant :

Le FC n'ayant pas assez de puissance pour alimenter directement les moteurs, il est nécessaire de passer par un transistor MOSFET qui va agir comme un interrupteur commandé en tension. Il suffit d'appliquer un signal (5V correspondant au niveau haut logique) sur l'entrée "Enable" du MOSFET pour fermer le circuit entre les moteurs et la batterie (cf figure 23).

Dans la plupart des cas, les ESCs sont équipés d'un BEC (ou régulateur de tension) qui permet également d'alimenter le FC en 5V. Dans la pratique il s'agit alors d'utiliser l'un des 4 BECs pour alimenter le FC.

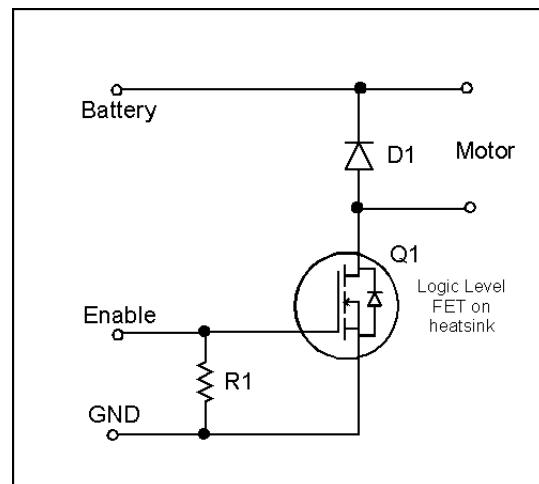


FIGURE 23 – Schéma de fonctionnement simplifié d'un ESC

Les ESCs sont donc connectés à la fois à la batterie, au FC ainsi qu'aux moteurs selon le schéma de la figure 24.

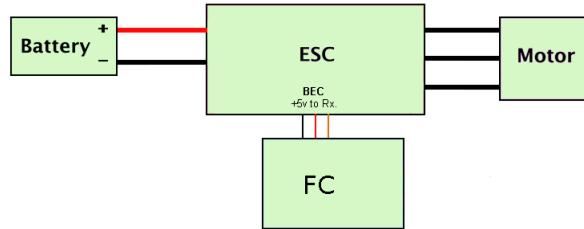


FIGURE 24 – Schéma de branchement d'un ESC

Il est important de préciser qu'au delà de l'alimentation, les ESCs et le FC communiquent via un signal de type PWM. En effet le FC fait varier sur une de ses broches son duty cycle pour indiquer à l'ESC la vitesse que doit avoir le moteur pour se stabiliser. Par exemple lorsque le duty cycle généré par le FC est à 50%, l'ESC doit faire tourner le moteur à la moitié de sa vitesse maximale.

Les moteurs

Les moteurs utilisés sur les quadcopters, en général, sont brushless (ou moteurs "sans-balais"). Ces moteurs sont triphasés d'où les 3 câbles de la figure 24. Ils sont caractérisés par leur KV (Motor Velocity Constant). Le KV est déterminé par le nombre de spires et le diamètre du fil de cuivre utilisé pour le bobinage des dents du moteur. Pour faire simple, un moteur à faible Kv est dit "coupleux" et un moteur à fort Kv est prédestiné à la vitesse.

Émetteur radio (télécommande) et notion de canaux

Avant d'aborder le fonctionnement du récepteur, il faut expliquer comment fonctionne la télécommande qui pilote le drone. Une télécommande possède plusieurs sticks pouvant agir sur les différentes orientations du drone conformément à la figure 25. Le stick de gauche contrôle les gaz (throttle) et le lacet (yaw), celui de droite contrôle le roulis (roll) et le tangage (pitch). Ces informations sont envoyées au récepteur sur différents canaux, chacun correspondant à une orientation particulière du drone. Il y a donc 4 canaux à communiquer au récepteur : Throttle, Roll, Pitch et Yaw.



FIGURE 25 – Schéma d'une télécommande

Le récepteur radio

Le rôle du récepteur radio est, comme son nom l'indique, de récupérer le signal issu de la télécommande. C'est un élément très important dans ce projet car il s'agit de l'un des modules que nous allons devoir concevoir pour être capable de réceptionner les informations transmises par l'émetteur sur les mains. Il est donc primordiale de connaître le type de signal en sortie d'un récepteur pour pouvoir le "copier".

Il existe plusieurs manières, pour un récepteur radio, de communiquer avec le FC pour lui transmettre les commandes envoyées par le pilote. On distingue principalement 2 types de récepteurs : les récepteurs PPM/CPPM (Pulse Position Modulation/Combined PPM) et PWM (Pulse Width Modulation).

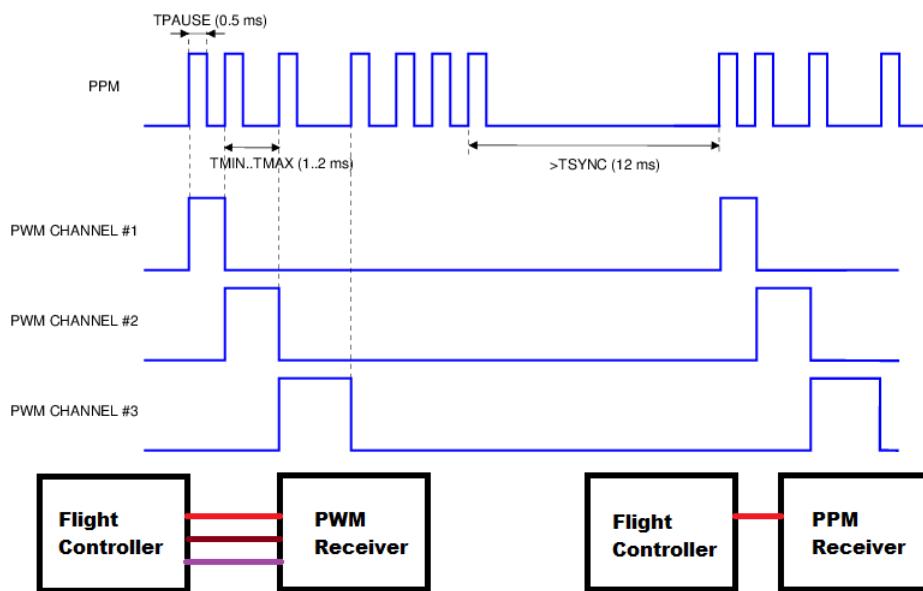


FIGURE 26 – Trames PPM/CPPM et PWM

La principale différence entre ces 2 types de récepteurs est le nombre de câbles utilisés pour la connexion au FC (cf figure 26). Par exemple dans le cas d'un récepteur PWM, les différents canaux sont séparés en sortie et doivent donc être connectés séparément au FC. A l'inverse, dans le cas d'un récepteur PPM/CPPM, un seul signal est généré, ce qui simplifie le branchement au FC car la totalité des canaux sont maintenant "mélangés" sur un seul câble. Dans le cadre du projet nous nous sommes focalisés sur ce type de récepteur, plus simple à réaliser car ce système n'utilisera qu'une broche de l'Arduino. Il s'agit maintenant de bien comprendre ce que représente ce signal CPPM pour mieux le "copier" en y intégrant les valeurs d'angle reçues.

3.2 Combined Pulse Position Modulation (CPPM)

Maintenant que nous savons comment fonctionne le récepteur, il s'agit de bien étudier le signal que nous allons envoyer au FC. C'est donc un signal de type CPPM (Combined Pulse Position Modulation).

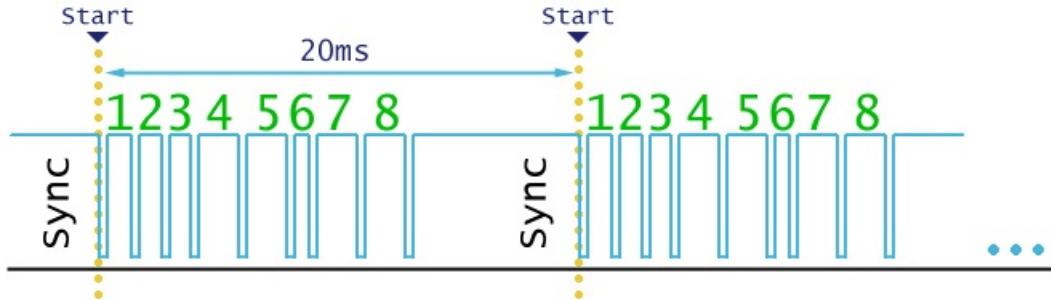


FIGURE 27 – Trame d'un signal CPPM

Comme on peut le voir sur la figure 27, le signal se décompose en période de 20ms. Cette valeur peut changer suivant les différentes marques de récepteurs mais elle importe peu tant que l'ordre de grandeur est respecté (entre 18ms et 22ms). Sur chaque période on distingue plusieurs impulsions de largeurs différentes qui modélisent les informations liées au pitch, roll, yaw et au throttle. Ces impulsions peuvent aller de $1000\mu\text{s}$ jusqu'à $2000\mu\text{s}$. Les 4 autres canaux servent d'entrées auxiliaires pour des boutons (changer le type de vol, activer diverses options, etc).

On remarque aussi une impulsion de synchronisation en fin de période, indispensable pour que le FC interprète correctement la trame.

Le but de la manœuvre est donc de récupérer les angles et de les adapter de manière à respecter la structure du signal CPPM. Cette partie sera détaillée par la suite.

3.3 Contrôleur 328p | Multiwii

Dans le cadre de ce projet, nous avons décidé d'utiliser un contrôleur de vol prenant en charge le code MultiWii.

Cela nous permet d'avoir une interface facilement configurable et offrant une large gamme de capteurs tel qu'un baromètre, un magnétomètre, plusieurs gyroscopes/accéléromètres, etc. De plus, le code est open source et on peut le modifier directement depuis l'IDE Arduino.

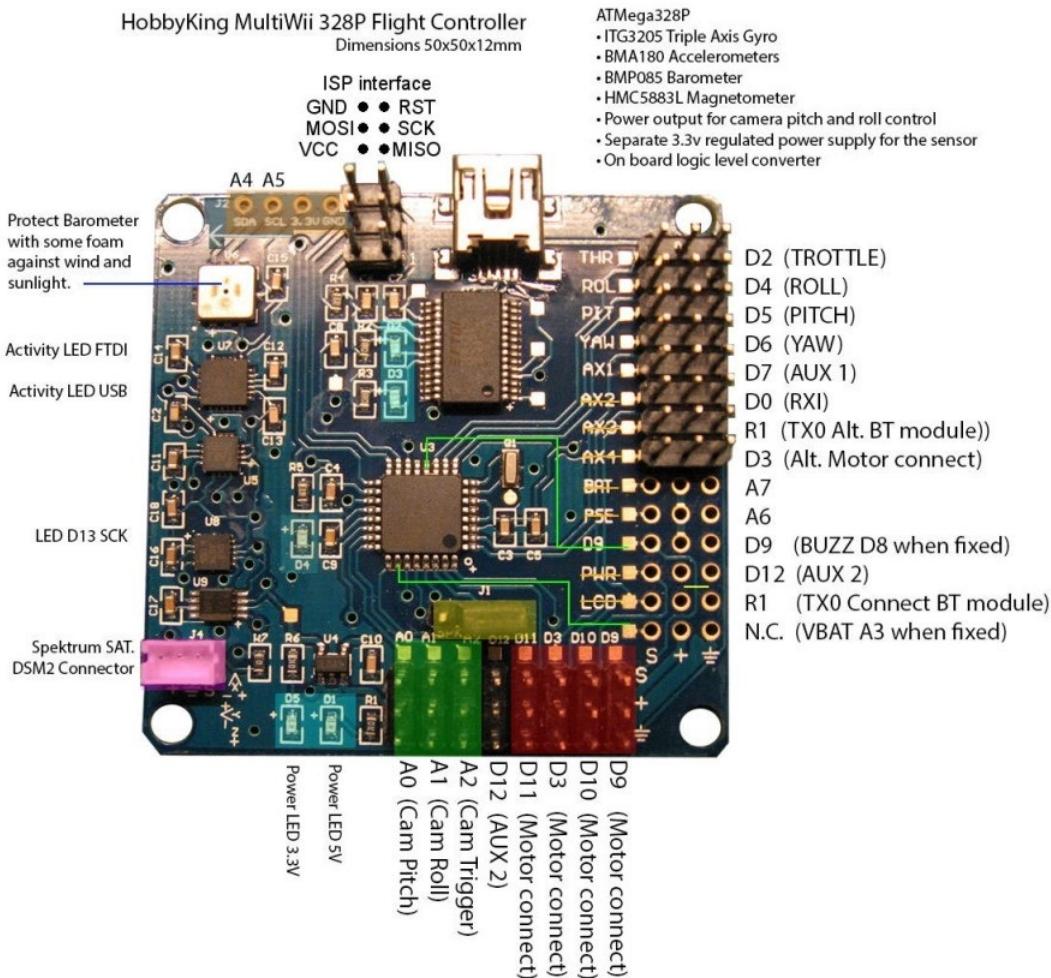


FIGURE 28 – Contrôleur MultiWii 328p

On peut voir sur la figure 28 que ce FC possède de nombreuses entrées/sorties. Il permet de brancher jusqu'à 4 moteurs, ce qui correspond à notre besoin (quadcopter). Dans une éventuelle poursuite du projet, il est également possible de rajouter d'autres modules tel qu'une caméra, un sonar, un GPS, un module Bluetooth ou autre.

3.4 Récepteur

Comme nous l'avons vu précédemment, le contrôleur de vol Multiwii attend un signal CPPM pour commander les moteurs. L'objectif même de notre système est d'imiter le signal envoyé depuis une télécommande classique.

Placée sur le drone, la carte de réception est composée d'un module NRF24L01 (le même que celui utilisé pour l'émission des données). Ce module reçoit les données envoyées par l'émetteur : il s'agit des valeurs d'angles des mains en degrés.

A ce niveau il faut donc utiliser ces valeurs d'angles afin de créer le signal CPPM qui permettra au drone de réagir aux mouvements des mains ; c'est le rôle de l'Arduino qui est placé en réception. Nous avons vu que le "temps haut" de chaque channel du signal CPPM doit être compris entre 1000 et 2000 microsecondes afin d'être compris par le code Multiwii. Les valeurs des angles sont comprises entre -90° et 90° degrés pour le pitch, le roll et le throttle. Par une simple règle de trois, nous pouvons translater les valeurs de l'intervalle $[-90; 90]$ vers l'intervalle $[1000; 2000]$. Il existe déjà une fonction Arduino permettant de réaliser cette opération : la fonction map(). Dans notre cas, on a les lignes de codes suivantes :

```
rpitch = map(mpitch, -90, 90, 1000, 2000);
rroll = map(mroll, -90, 90, 1000, 2000);
rthr = map(mthr, 90, -90, 1000, 2000);
```

Ainsi, l'inclinaison de nos mains génèrent des valeurs d'angle en degré. Ces dernières sont ensuite transformées en valeurs qui correspondent au temps haut (en microsecondes) de chaque channel du signal CPPM. On rappelle qu'une channel est dédiée pour le pitch, une autre pour le roll, etc.

Maintenant que nous disposons de ces valeurs en microsecondes, nous devons mettre en forme le signal CPPM. C'est nous qui choisissons quelle channel va correspondre à quelle orientation. Par exemple nous décidons que la première channel correspond au roll, la deuxième au pitch... Il faut être vigilant à bien faire cette correspondance dans le code Multiwii. Dans le fichier de configuration de Mutliwii, on trouve la ligne suivante :

```
#define SERIAL_SUM_PPM ROLL,PITCH,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11
```

Ici on voit que le premier élément est le ROLL, cela signifie que la voie 1 correspond au ROLL. Les termes AUX-X correspondent à des entrées auxiliaires (des boutons se trouvant sur une télécommande par exemple). Dans une éventuelle poursuite du projet, ces canaux nous permettraient d'ajouter de nouvelles fonctions de pilotage (atterrissement d'urgence, retour à la maison...).

Revenons à la mise en forme du signal CCPM : il faut émettre des impulsions dont la valeur va de 1000 à $2000\mu\text{s}$. Les canaux sont tous transmis les uns à la suite des autres avec une pause entre chaque. Pour cela, on utilise des fonctions à base de compteurs qui nous permettent de créer le signal voulu. On obtient par exemple le résultat suivant :

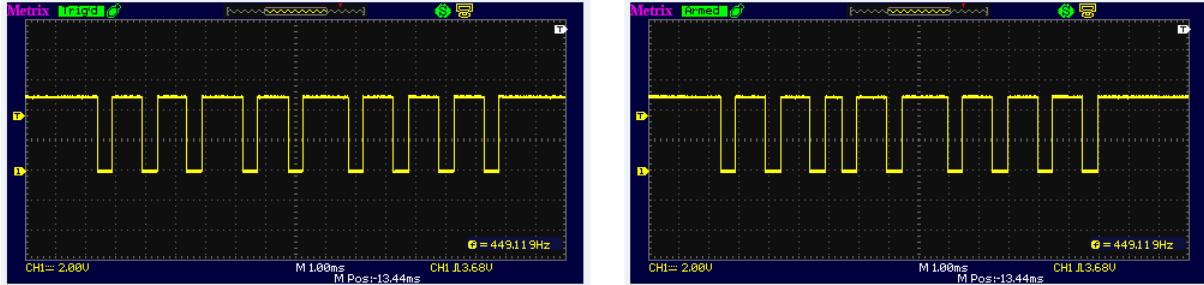


FIGURE 29 – Throttle élevé (gauche) / Throttle faible (droite)

Dans notre cas, le throttle est représenté par la voie 3 du signal CPPM. On voit bien que dans le cas d'un throttle élevé (qui correspond à une inclinaison de la main plus importante), la durée de l'impulsion est plus importante que dans le cas d'un throttle faible.

C'est de cette façon que l'on parvient à décrire de façon exacte l'inclinaison des mains. Le pilotage et le contrôle du drone s'effectuent ainsi très précisément.

3.5 Réalisation du prototype

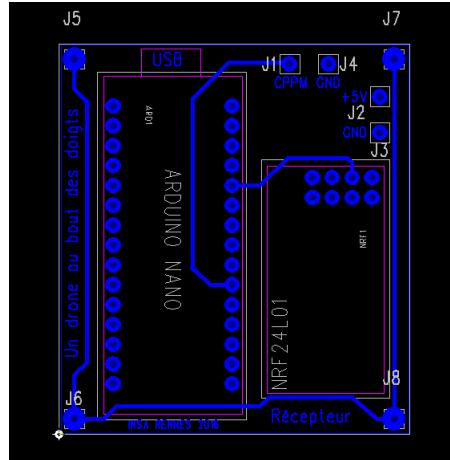


FIGURE 30 – Routage top du récepteur

Le récepteur est composé de 2 éléments : l'Arduino et le module NRF24L01. Sachant que cette carte doit être montée sur le drone, cela impose des contraintes de dimension de la carte. Cette carte est placée sur le haut du drone. En effet il est fondamental de faire attention au centre du gravité du drone qui pourrait entraîner une éventuelle instabilité. En prenant en compte le fait qu'il puisse y avoir d'éventuels chocs, il est nécessaire de protéger le récepteur en le plaçant dans une boîte. Cette dernière est collée directement sur le cadre du drone.

3.6 Tests

Après avoir vérifié que notre récepteur recevait bien toutes les données émises, il était nécessaire de tester le comportement du drone en conditions réelles.



FIGURE 31 – Premiers tests

Suite à une première série de tests (cf figure 31) accompagnée de plusieurs crashs, nous nous sommes penchés sur un élément important que nous avions omis de configurer : les coefficients PID.

Coefficients PID (*Proportional Integral Derivative*)

Comme un drone a de multiples raisons de quitter la trajectoire prévue il faut donc un mécanisme très rapide et efficace pour corriger les écarts par rapport à celle ci.

	P	I	D
ROLL	4.0	0.030	23
PITCH	4.0	0.030	23
YAW	8.5	0.045	0
ALT	1.6	0.015	7
Pos	0.11	0.0	
PosR	2.0	0.08	0.045
NavR	1.4	0.20	0.080
LEVEL	7.0	0.010	100
MAG	4.0		

FIGURE 32 – Réglage des coefficients PID

Ainsi, on retrouve 3 types de coefficients à régler pour agir sur l'asservissement du drone.

- La correction P (proportionnelle a l'écart) : plus l'écart est grand plus on corrige.
- La correction D (dérivée) vise à "atterrir" en douceur (sans sur-correction) près de la valeur cible.
- La correction I (intégrale) vise à corriger des dérives lentes et persistantes.

On peut trouver des valeurs de base sur internet comme on peut le voir sur la figure 32. Ces valeurs par défaut suffisent pour avoir un bon contrôle du drone. On pourrait imaginer par la suite une optimisation de ces valeurs.

Pour se faire il est nécessaire de réaliser une autre batterie de tests en vol pour bien évaluer les différents points à corriger. En pratique il s'agit de procéder par tâtonnement afin de trouver les meilleures réglages. Pour modifier les coefficients, il faut utiliser le logiciel Multiwii.

4 Bilan

4.1 Bilan technique

4.1.1 Démonstration finale

Notre démonstration finale aura pour but de faire voler notre drone à l'aide des 2 boîtiers de contrôle.

Comme évoqué dans le cahier des charges, nous effectuerons notre démonstration dans la halle Francis Querne de l'INSA de Rennes. Nous avons choisi cet espace car il est haut de plafond et nous laisse un espace de vol confortable. Il nous permet donc d'évoluer dans des conditions similaires à un vol en plein air, sans les conditions défavorables (vent, pluie, etc) qui pourraient nuire à la démonstration et rendre le vol impossible.



FIGURE 33 – Démonstration finale

Le pilote, muni des 2 boîtiers de contrôle, se placera à quelques mètres du drone. Une seconde personne sera chargée de mettre sous tension le drone. Concernant la démonstration en elle-même, nous effectuerons plusieurs petits vols afin de montrer le bon fonctionnement du projet ainsi que les différentes fonctions implémentées. Les vols auront pour but de faire circuler le drone dans la halle, d'un point A à un point B sur une trajectoire rectiligne ou non et de faire décoller et atterrir le drone à un même endroit, en ayant effectué quelques commandes de base (déplacement latéral, d'avant en arrière, à altitude variable).

Le jury pourra ensuite, s'il le souhaite, s'essayer au contrôle du drone.

4.1.2 Améliorations possibles

Lors de la réalisation du cahier des charges, nous avions envisagé un certain nombre de fonctionnalités plus ou moins évidentes à implémenter. Nous étions conscients que certaines options ne pourraient être réalisées. C'est pourquoi nous avons établi un cahier des charges plutôt vaste et ambitieux, afin de ne pas se limiter simplement à l'émulation d'une télécommande de drone.

Nous avons envisagé l'ajout éventuel de boutons poussoirs sur lequel n'importe quelle fonction peut être implémentée (changement de mode de vol par exemple). Leur emplacement est déjà prévu sur le PCB des cartes de contrôle. Comme nous l'avons vu précédemment, notre signal CPPM comporte encore de nombreux canaux non utilisés. Il suffira de préciser dans le code Multiwii l'apparition de ces nouvelles fonctions. De plus, comme il s'agit de boutons poussoirs (des actions "tout ou rien"), l'étape de mapping ne sera pas nécessaire. L'impulsion de la channel sera de $1000\mu\text{s}$ quand le bouton n'est pas pressé, elle sera de $2000\mu\text{s}$ si on appuie dessus.

Nous souhaitions également ajouter des fonctions de sécurité comme un atterrissage d'urgence au cas où l'utilisateur perdrat le contrôle. Cependant, nous avons prévu uniquement le cas où il n'y aurait plus de batterie dans les mains. Il y a alors coupure automatique des moteurs. Cette fonctionnalité peut également être utilisée en cas de perte de contrôle du drone, par simple déconnexion du fil situé entre la batterie et les mains. Il serait alors plus judicieux de gérer cette fonction à l'aide des boutons poussoirs évoqués au paragraphe précédent.

Nous avons envisagé d'intégrer un sonar HC-SR04^(Figure 34) au drone. Le sonar permettrait de détecter le sol et à une certaine distance, le programme entraînerait un ralentissement progressif des moteurs afin d'effectuer un atterrissage automatique. Cette fonctionnalité permet donc de préserver l'intégrité du drone lors de l'atterrissement. Cependant, la principale difficulté réside dans le fait que la version de multiwii utilisée pour le projet n'intègre pas de gestion de sonar. Il existe cependant des versions alternatives de multiwii qui permettent l'intégration d'un sonar. Nous n'avons pas utilisé ces versions car ce sont des versions modifiées et donc instables.



FIGURE 34 – Sonar HC-SR04

Malgré de nombreux efforts sur les différents codes, nous n'avons pas pu améliorer la latence comme nous le souhaitions. Nous observons toujours un petit laps de temps entre la commande effectuée par la main et le mouvement réel du drone. Une étape cruciale serait de réduire ce temps de sorte que la latence ne se remarque pas pour l'utilisateur.

Il serait également possible de supprimer la liaison filaire entre les 2 mains afin de gagner en liberté de mouvement. Cependant, nous avons rapidement écarté cette solution, car il y avait un grand risque d'augmenter la latence du système. De plus, cela aurait induit des coûts supplémentaires car nous aurions eu besoin d'une batterie supplémentaire, ainsi que de 2 modules bluetooth.

Une amélioration, assez simple à faire, serait de déporter la diode de verrouillage dans un endroit plus visible (sur le dessus du drone par exemple). Cette diode indique si le drone a été verrouillé par l'utilisateur en plaçant la main des gaz à la verticale et autorise alors la mise en marche du drone. Cependant, nous n'avons pas pris le risque d'effectuer cette manipulation car elle est soudée en CMS, ce qui demande une grande précision et une certaine maîtrise au fer à souder.

La dernière étape serait de miniaturiser les cartes que nous avons conçus. Pour cela, il faudrait éventuellement optimiser le routage. Mais il faudrait surtout souder directement la puce arduino en CMS sur les plaques et ne pas utiliser de carte arduino préfabriquée. Toutefois, l'avantage de ces cartes est qu'elles sont facilement intégrables et plus pratiques pour concevoir un prototype.

4.2 Gestion du projet

4.2.1 Répartition des tâches

Pour avancer dans le projet, nous nous sommes répartis le travail de la façon suivante :

Partie	Tâches	Description	Etat	mise à jour le :	02/05/2016
				Ressources	
Gant	Routage	prévoir : 1PCB (MPU/NRF/Arduino -> Alimentation) + 1PCB (MPU throttle)	Terminé	Antoine/Benjamin	
	Nettoyage code émetteur	Supprimer les parties de code inutiles (laisser la partie Serial pour debug)	Terminé	Arthur/Benjamin	
	Intégrer code de filtrage de Gael	Voir si le code est intégrable	En cours	Arthur/Gael	
	Mettre le 2ème MPU	Notion d'adressage	Terminé	Antoine/Gael	
	Design du gant	Penser à la connexion entre le MPU du throttle et le reste (nappe?) + brassard ?	Terminé	Domitille/Thibault	
Drone	Routage (NRF/Arduino)	1 PCB (NRF/Arduino) (prévoir un pin CPPM pour aller au contrôleur et 2 pins pour l'alimentation)	Terminé	Antoine/Benjamin	
	Gestion de la sécurité	bouton atterrissage (voir les fonctionnalités du code Multiwii)	En cours	Antoine/Octave	
	Tutoriel fonctionnement du drone	Poster détaillé + Manuel + vidéo ?	En cours	Domitille/Thibault/Octave	

FIGURE 35 – Exemple de répartition des tâches daté du 5 mai 2016

Nous étions généralement deux personnes par tâche. Ceci nous permettait de confronter nos idées et d'avancer plus efficacement. Lors de ce projet, nous avons utilisé la plateforme de partage Google Drive. Nous avons ainsi pu mettre en commun tous les documents liés au projet et y avoir accès rapidement. Le tableau de répartition du travail était ainsi à la disposition de tous, permettant à chacun de suivre l'avancement des différents binômes.

De plus, nous nous sommes réunis très régulièrement au cours du second semestre. Au moins une fois par semaine, nous faisions un rapide résumé de l'état d'avancement de chaque binôme, de ce qu'il restait à faire et des difficultés rencontrées. Cette bonne communication (et bonne entente) au sein du groupe a été un moteur pour le projet et nous a permis de travailler efficacement, sans prendre de retard.

4.2.2 Gestion du budget

Le département SRC nous a alloué un budget de 120€ pour réaliser un prototype. La justification des différents composants choisis a été détaillée précédemment dans ce rapport.

Voici la liste des composants achetés, classés par fournisseur :

Nom composants	Nb pièces requis	Nb pièces achetés	Fournisseur	Prix
Kit drone + contrôleur Multiwii	1	1	HobbyKing	170€
Transceiver NRF24L01	2	5	Bangood	4,75€
Arduino Atmega328P	2	2	Bangood	5,69€
Capteur MPU	2	3	Amazon	9,21€
Sonar	1	1	Bangood	1,69€
Arduino Atmega328P (2e commande)	1	1	Bangood	2,76€
Batterie Giant Power 7.4V 2S 600mAh 25C	1	1	Bangood	8,30€
Connecteurs batterie	2	2	Bangood	0,91€
Hélices de rechange	au moins 4	12	Bangood	5,67€
Frais de port Bangood				5,57€
Elastiques + ficelle			Carrefour	3,00€
Boîtiers noirs	3	3	Atlantique Composants	8,60€
Connecteur RJ9	2	2	Atlantique Composants	3,23€
Coût total		229€		
Montant pris en charge par le département		170€		
Coût du projet		59€		

FIGURE 36 – Liste des composants

Le budget total dépasse les 120€ initiaux. Cependant, l'équipe pédagogique a accepté d'acheter le drone. C'est un investissement pour le département SRC puisque ce drone n'a pas de code propriétaire et pourra donc être utilisé par d'autres étudiants ou chercheurs pour d'autres projets.

Pour plusieurs raisons, nous avons fait le choix de ne pas suivre la liste des fournisseurs conseillés par le département. La première est que certains composants n'étaient disponibles qu'en CMS, or le département ne dispose pas du matériel adéquat pour intégrer ces composants sur une carte. Nous aurions pu tenter de les souder nous-mêmes, mais le risque à prendre était trop important.

De plus, le site Bangood où nous avons acheté la majorité des composants propose des composants à prix défiant toute concurrence. Mr Gilles PICOULT nous a alors donné son feu vert. Nous savions que ce site était fiable car nous avions passé des commandes personnelles auparavant. Cette démarche est en accord avec notre futur métier d'ingénieur qui est de trouver les meilleurs composants au meilleur prix.

Pour rappel, notre budget prévisionnel s'élevait à un montant de 37,84€. Aujourd'hui, le prix de notre prototype est de 59 €. Entre temps, nous avons pensé à des fonctions de sécurité pour le pilotage et avons acheté un sonar. Nous avons également acheté des composants de rechange (hélices, connecteurs...) ainsi qu'un deuxième Arduino pour la deuxième main et des boîtiers pour protéger les cartes électroniques et rendre le prototype plus ergonomique.

Finalement, nous avons tout à fait respecté la contrainte budgétaire.

4.2.3 Gestion du temps

Concernant la gestion du temps, nous n'avons pas eu de retards particuliers. Le diagramme de Gantt prévisionnel nous a permis de vérifier régulièrement si nous étions dans les temps.

Voici quelques modifications qui ont été apportée à ce diagramme :

- l'absence de Gilles Picoult courant mars pendant deux semaines et demi nous a encouragé à terminer le routage des cartes. Nous avons dû avancer cette étape d'une quinzaine de jour. Nous avions pris de l'avance sur cette étape, cela n'a pas été un problème majeur.
- la soutenance du projet a été avancée de cinq jours. Nous avons été prévenus suffisamment longtemps à l'avance, d'où un impact moindre.
- le premier vol du drone a été réalisé le 23 mars ce qui est environ trois semaines avant la date prévue initialement.

Voici les diagrammes de Gantt prévisionnels^(Figure 37) et effectifs^(Figure 38) de notre projet (pour la période de janvier à juin 2016).

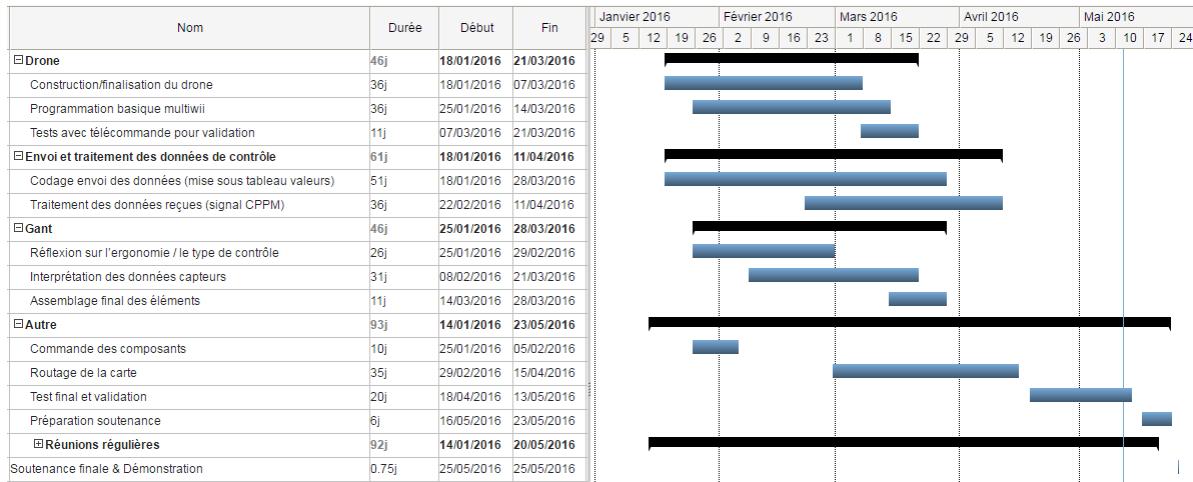


FIGURE 37 – Diagramme de Gantt prévisionnel

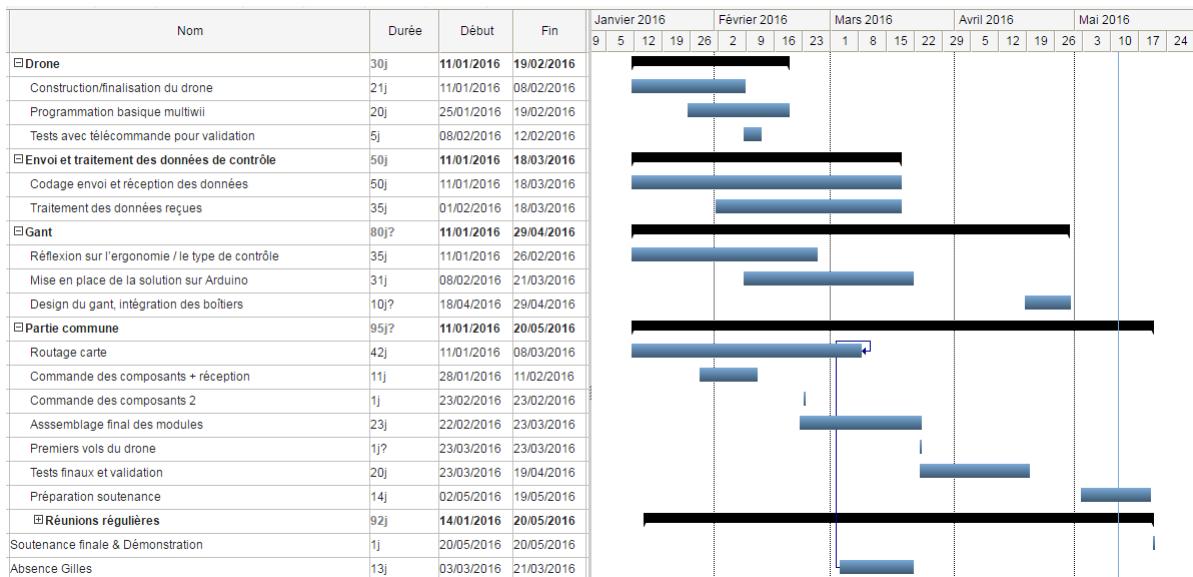


FIGURE 38 – Diagramme de Gantt réel

4.2.4 Protection et brevetabilité

Suite aux conseils de nos encadrants, nous avons contacté puis rencontré Adrien Guilhaire, chargé d'affaires chez OUEST VALORISATION. L'objectif de cette rencontre était de se poser les bonnes questions pour appréhender la notion de protection/brevetabilité de notre système de pilotage de drone. Voici une synthèse de la démarche adoptée lors de l'entretien.

Quels critères doit respecter notre système ?

Différents critères doivent être considérés pour envisager la brevetabilité :

- Nouveauté : le critère de la nouveauté impose que l'objet sur lequel repose la demande du brevet soit jusqu'à présent inconnu dans le monde entier. L'inventeur, lorsqu'il fait sa demande de brevet, doit être le premier à revendiquer la technologie qu'il expose.
- Non-évidence/inventivité : suivant le critère de non-évidence, une invention doit à la base résulter d'un processus de créativité pour être brevetable. Cette créativité peut s'exprimer de différentes manières ce que la loi ne restreint aucunement. L'invention qui découle de manière évidente de l'évolution d'une technologie ne fait pas intervenir un degré suffisant de créativité pour engendrer l'octroi d'un brevet.
- Application industrielle : l'objet de l'invention doit pouvoir être fabriqué ou utilisé dans l'industrie : il doit pouvoir être reproduit sous peine de nullité.

Le système répond bien au critère d'application industrielle. Pour appréhender la notion d'inventivité de notre projet, il faut le conceptualiser de façon abstraite et ne pas considérer le matériel en tant que tel (arduino, NRF, MPU). On considère ainsi le système dans sa globalité (capteurs, traitement des données, algorithmes). D'un point de vue abstrait, notre projet se résume à récupérer des valeurs d'angles et à les rendre interprétables par un drone pour en assurer le pilotage.

Quels systèmes existent ? Quel est le système existant le plus proche du nôtre ?

Actuellement, la majorité des drones sont contrôlés par une télécommande à l'aide de joysticks. Les drones peuvent également être pilotés par des smartphones grâce aux capteurs qui y sont intégrés. Cette façon de piloter semble la plus proche de notre système (récupération d'angles et envoi au drone).

Qu'est ce qui différencie notre système d'un pilotage par smartphone ?

- Utiliser 2 mains : séparer les gaz des autres données, pouvoir écarter les mains (ne pas avoir les deux mains sur la télécommande ou sur le smartphone).
- Récupérer les gaz sous la forme d'un angle : principe d'une pédale d'accélération.

Cependant, au regard des conditions de brevetabilité, les traitements des signaux issus des capteurs mis en œuvre dans la solution proposée, ne se différencient pas suffisamment pour permettre de valider les critères de nouveauté et d'activité d'inventive, par rapport à ce qui peut être fait au sein d'une télécommande gestuelle ou d'un smartphone.

Conclusion

Cet entretien nous a permis d'avoir l'avis d'une personne extérieure qualifiée dans le domaine de la protection industrielle. De plus, cela nous a amené à nous poser des questions pertinentes et à obtenir des réponses objectives. Cette rencontre apporte une réelle finalité à notre projet.

4.3 Bilan moral

Octave

Cette expérience m'a permis de renforcer ma capacité d'adaptation qui est une compétence essentielle pour un ingénieur. Tout d'abord, nous avons dû nous renseigner sur le principe de fonctionnement d'un drone à quatre hélices, ce qui a suscité ma curiosité. Par la suite, nous avons travaillé sur des codes en langage C déjà existants que nous avons modifiés en fonction de notre besoin puis intégrés à notre système. Nous serons éventuellement amenés à procéder de la même manière en entreprise. Enfin, le passage de la conception sur ordinateur à la réalisation concrète du prototype m'a particulièrement marqué car j'ai pu constater l'importance d'intégrer des contraintes techniques. Je tiens à remercier Gilles Picoult pour la patience avec laquelle il a su nous faire bénéficier de son expertise.

Un grand merci à tous les membres du groupe pour leur implication dans le projet, et avec qui j'ai eu grand plaisir à travailler. Merci encore au département SRC pour l'achat du drone, la mise à disposition du FabLab et du matériel, sans quoi le projet n'aurait pas pu voir le jour.

Thibault

Je remercie le département SRC ainsi que son équipe pédagogique de nous offrir la possibilité de réaliser un projet technologique dans son intégralité. Ce projet m'a fait prendre conscience des nombreux enjeux auxquels je pourrais être confronté dans ma future carrière d'ingénieur. Réaliser un cahier des charges respectant les contraintes imposées était un véritable challenge, mais le plus stimulant pour ma part était la réalisation technique du projet en elle-même. Je remercie l'intégralité des membres du groupe pour leur motivation, leur présence et pour les nombreux échanges que nous avons eus, toujours dans le plus grand des respects. C'était également un réel plaisir de travailler avec Gilles Picoult lors de cette seconde phase, de profiter de ses nombreux conseils et de son savoir-faire, notamment en observant la fabrication des circuits imprimés.

Gaël

Pour ma part, ce projet a été très enrichissant et m'a permis de voir les aspects théoriques d'un projet (après avoir vu les aspects du cahier des charges lors du projet drone compteur de foules). J'ai donc pu me confronter au problème que nous rencontrerons lors de notre futur métier d'ingénieur. Il m'a aussi permis de me renforcer dans la capacité d'auto-formation qui sera très importante durant ma carrière d'ingénieur. Je remercie l'intégralité des membres du projet pour leur motivation et leurs échanges sur ce projet. Je remercie également le département SRC et son équipe pédagogique de nous permettre de réaliser un projet innovant et particulièrement Gilles Picoult pour son aide lors de la réalisation du projet.

Valentin

En ce qui me concerne, travailler sur ce projet a été une expérience enrichissante autant du point de vue de l'apport de connaissances que de l'organisation de l'équipe. Creuser ce vaste sujet qu'est le monde des drones m'a permis de m'ouvrir sur un domaine totalement nouveau, et par lequel, je dois le dire, j'ai été séduit ! C'est, d'après moi, une très bonne façon de s'approcher de notre futur métier d'ingénieur. Avec tout de même un petit regret de ne pas avoir été présent pour la partie pratique...

Timothée

Ce projet a été l'occasion de mettre en pratique nos compétences techniques et de les approfondir. Je n'ai pas pu assister à la réalisation du prototype mais j'ai quand même pu suivre l'avancement et effectué un vol avec notre système. Le domaine des drones, que j'affectionne particulièrement, est un domaine très vaste qui laisse place à la créativité et à l'innovation. Merci aux membres de l'équipe qui ont mené à bien le projet.

Domitille

Je rejoins les autres membres de l'équipe pour dire que ce projet a été une très bonne expérience au cours de cette deuxième année de cycle ingénieur SRC. Je suis très satisfaite d'avoir fait partie de ce projet et d'en avoir suivi toutes les étapes, de la conception à la réalisation. J'ai particulièrement apprécié prendre part à la gestion du projet : comme organiser le planning ou suivre le budget. Ce sont les aspects d'un projet qui me plaisent et dont je suis prête à gérer dans ma future carrière d'ingénieure.

Arthur

Pour ma part, ce projet constitue une expérience unique pendant laquelle il m'a été possible de renforcer mes compétences dans le domaine de l'électronique/programmation et d'avoir pu travailler en équipe au sein d'un groupe motivé et envieux de mener à bien ce projet jusqu'au bout, jusqu'à même se poser la question d'une éventuelle brevetabilité, apportant une véritable finalité au projet. Ce fut également très intéressant d'avoir été soumis aux mêmes contraintes que nous pourrions retrouver plus tard dans le métier d'ingénieur (mise en place d'un cahier des charges ou encore la gestion du budget). Je souhaite remercier l'INSA de Rennes pour toutes les ressources mises à notre disposition et spécialement Gilles Picoult pour sa disponibilité et pour ses nombreux et bons conseils.

Antoine

Cette expérience a été enrichissante et a permis de balayer les différents aspects d'un projet tel qu'on le trouvera en tant qu'ingénieur au quotidien (autoformation technique, gestion du temps, gestion du budget, gestion des ressources humaines). En partant d'une idée, nous avons su concevoir, valider puis réaliser un prototype. La cohésion au sein du groupe et nos compétences diverses nous ont permis de résoudre les problèmes auxquels nous étions confrontés. J'ai vraiment pris du plaisir à travailler en collaboration avec Gilles Picoult qui nous a fait part de son expérience et de ses astuces.

Deux étapes m'ont particulièrement marqué :

- Assister au process complet de réalisation d'un circuit imprimé et en comprendre les contraintes technologiques.
- Rencontrer Adrien Guilhaire pour échanger autour de la protection/brevetabilité de notre système.

Je remercie mes partenaires ainsi que toute l'équipe pédagogique du département SRC sans qui le projet n'aurait pas abouti.

Benjamin

La réalisation de ce projet tout le long de cette année a été pour moi d'une grande satisfaction. Je suis ravi d'avoir travaillé dans un groupe dont la motivation et la bonne humeur ont nécessairement été des éléments clés à la réussite de ce projet !

Références

- [1] **Mapassiondumodelisme**,<http://mapassiondumodelisme.jimdo.com/technique/quadrifopt%C3%A8re/fonctionnement-d-un-quadrifopt%C3%A8re>
Fonctionnement d'un quadcopter / Notions de base d'aéronautique
- [2] **GeekMag**,<http://www.geekmag.fr/quadrifoptere-fonctionnement/configuration-et-pilotage-rc/>
Fonctionnement et configuration d'un drone
- [3] **Forum RCGroups**,<http://www.rcgroups.com/forums/index.php>
Forum référence en modélisme
- [4] **MultiWii**,<http://www.multiwii.com/>
Téléchargement du firmware / Quelques tutoriels
- [5] **Hobbyking**,<http://www.hobbyking.com/hobbyking/>
Achat du drone en kit / Achat du contrôleur de vol multiwii 328p
- [6] **Hobbyking**,<http://www.hobbyking.com/hobbyking/store/uploads/463565377X19082X13.pdf>
Réglage des PID
- [7] **DIY Hacking**,<http://diyhacking.com/arduino-mpu-6050-imu-sensor-tutorial/>
Fonctionnement du MPU6050
- [8] **Tiptopboards**,http://tiptopboards.free.fr/arduino_forum/viewtopic.php?f=2&t=28
Exemples de codes Arduino pour MPU6050
- [9] **Itechnofrance**,<https://itechnofrance.wordpress.com/2013/05/24/utilisation-du-module-nrf24l01-avec-larduino/>
Utilisation du NRF24L01 avec Arduino
- [10] **DiyDrones**,<http://diydrones.com/profiles/blogs/why-frsky-cppm-signal-is-so-disappointing/>
Signal CPPM

A Annexes

A.1 Manuel d'utilisation

The image shows two pages of a manual titled "Manuel d'Utilisation" for a drone controlled by hands. Both pages feature the INSA Rennes logo at the top.

Page 1 (Left):

- Section:** Un drone au bout des doigts
- Title:** Manuel d'Utilisation
- Image:** A silhouette of two hands holding a quadcopter drone between them.
- Text:** "Un drone au bout des doigts:
Projet d'Innovation de 4^{ème} année
Département Systèmes et Réseaux de Communication – INSA Rennes"

Page 2 (Right):

- Section:** Un drone au bout des doigts
- Section:** Au préalable
- Text:** Assurez-vous que les 2 batteries (drone et télécommande) soient bien chargées. Placez le drone au sol, posé en équilibre sur ses 4 pieds. Assurez-vous de disposer d'un endroit dégagé, peu exposé au vent et aux intempéries pouvant entraîner des difficultés de pilotage. Assurez-vous également que toute personne ou animal ne soit mis en danger à cause du drone.
- Section:** Installation
- List:**
 - Connectez la batterie au drone: une diode bleue située sur le contrôleur de vol doit se mettre à clignoter.
 - Reculez-vous de quelques mètres.
 - Enfilez les boîtiers de contrôle sur chaque main.
 - Reliez ensuite les 2 boîtiers entre eux (câble RJ9) et connectez la batterie au premier boîtier (câble USB).
- Section:** Armement du drone
- Text:** Placez la main des gaz à la verticale, vers le haut. La diode bleue doit alors être allumée en continu. Si la diode continue de clignoter, veuillez effectuer cette procédure à nouveau.

The image shows two continuation pages of the manual, both featuring the INSA Rennes logo at the top.

Page 3 (Left):

- Section:** Un drone au bout des doigts
- Section:** Pilotage
- Text:** La main des gaz fonctionne comme une pédale d'accélération. Avec l'autre boîtier, vous pouvez diriger le drone dans l'espace, d'avant en arrière et de gauche à droite par de simples inclinaisons de la main.
- Section:** Mise à l'arrêt
- Text:** Lorsque vous posez le drone au sol, débranchez la batterie des boîtiers de contrôle, puis la batterie du drone. Vous pouvez ensuite ranger votre drone.
- Text:** En cas de problème (drone incontrôlable, obstacle imprévu, etc.), débranchez la batterie des boîtiers de contrôle. Cela entraîne une coupure instantanée des moteurs. Cette procédure peut cependant entraîner une casse partielle ou totale du drone.

Page 4 (Right):

- Image:** A silhouette of two hands holding a quadcopter drone between them.
- Section:** Un drone au bout des doigts
- Text:** By :
 - Auscher Octave
 - Boudet Arthur
 - Buttin Domitille
 - Caillard Valentin
 - Gadon Timothée
 - Le Calvez Antoine
 - Malgom Gael
 - Niescierewicz Benjamin
 - Perei Thibault
- Text:** INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

A.2 Poster de présentation

Un drone au bout des doigts

Projet électronique 2015-2016

A. Boudet, D. Buttin, A. Le Calvez, T. Perel, O. Auscher
B. Niescierewicz, G. Malgorn, V. Caillard, T. Gadon

Conseils et réalisations techniques : Gilles Picoult

 **Innovation**
Un nouveau mode de pilotage de drone

 **Ergonomie**
Une prise en main rapide intuitive et sécurisée

 **Distribution** Un système entièrement open-source



Transformez vos mains en une véritable télécommande de drone !



Département Systèmes et Réseaux de Communication

INSA