

Université de La Manouba
Ecole Nationale des Sciences de l'Informatique



Rapport de Projet de Conception et de Développement

Sujet

SYSTÈME DE NAVIGATION AUTONOME

Réalisé par

Trabelsi Mohamed

Abidli Abderrahmen

Trifi Amanallah

Encadré par
Dr.Chadlia Jerad

Année Universitaire : 2014/2015

Signature de l'encadrant

Remerciements

Nos remerciements les plus intenses sont adressés, au terme de ce modeste travail, à tous ceux qui de près ou de loin ont contribué pour que ce dernier puisse être mené à bien.

Nous tenons tout d'abord à exprimer notre gratitude pour nos familles pour leurs soutiens moraux aussi bien que financiers.

Notre gratitude s'adresse également à notre encadrante DR. Chadlia Jerad pour ses précieux conseils et sa grande disponibilité, aussi bien que pour le personnel de l'ENSI pour leurs conseils judicieux et leurs soutiens permanents.

Nous voudrions en faire part de notre amabilité à tous nos collègues de l'ENSI.

Nous tenons, aussi, à exprimer l'honneur que nous font les membres du jury pour avoir accepté de nous prêter leur attention et évaluer notre travail.

TABLE DES MATIÈRES

Introduction Générale	6
1 Présentation du Projet	7
1.1 Etude préalable	7
1.1.1 Plateformes aériennes	7
les dronnes	7
les véhicules aériens sans pilotes	8
1.1.2 Les planeurs	8
Principe de vol	8
Exemples des planeurs autonomes	10
1.2 Etat de l'art de l'asservissement	11
1.3 Etude technique	12
2 Analyse et Spécification des besoins	14
2.1 Spécification des besoins	14
2.1.1 Besoins fonctionnels	14
2.1.2 Besoins non fonctionnels	15
2.2 Description des fonctionnalités et scénarios	16
2.2.1 Diagramme de cas d'utilisation	16
2.2.2 Diagramme de séquence	16
3 Conception	19
3.1 Conception globale	19
3.2 Conception détaillée	20
3.2.1 Conception du planeur	20
Architecture matérielle	20
Architecture logicielle	21
3.2.2 Conception de l'application de contrôle	23
Diagramme de classes	24

3.2.3	Conception de simulateur	26
Description	26	
Diagramme de classes	27	
4 Réalisation		28
4.1	Réalisation de la partie matérielle	28
4.1.1	Choix de microcontrôleur	28
4.1.2	choix de capteurs	29
Capteurs d'orientation : Acceléromètre/ gyroscope/ Magnétomètre	29	
Capteur température :	30	
capteur de pression barométrique :	31	
4.1.3	Module de communication distante	31
4.1.4	Choix des actionneurs	32
Les servo-moteurs	32	
Moteurs Brushless	33	
4.2	Réalisation de la partie logicielle	33
4.2.1	Environnement et outils de développement	33
MATLAB	34	
SIMULINK	34	
La bibliothèque Aerospace Toolbox :	35	
La bibliothèque 3D Animation	36	
API Google Maps	36	
4.2.2	Description de l'application	37
4.2.3	Description du simulateur	42
Conclusion Générale		42
Netographie		43
Bibliographie		44

TABLE DES FIGURES

1.1	Forces agissant sur le planeur en équilibre	9
1.2	Profil d'une aile de planeur	9
1.3	actions subies par l'aile	10
1.4	Super-Dimona	10
1.5	DG-400	11
1.6	l'asservissement avec PID	12
1.7	Comparaison des microcontrôleurs	13
2.1	Diag des cas d'utilisations	16
2.2	Diagramme de séquence : pilotage manuel	17
2.3	Diagramme de séquence : pilotage automatique	18
3.1	schéma descriptif de l'architecture du système	20
3.2	Architecture matérielle	21
3.3	les tâches de la partie logicielle embarqué	22
3.4	Schéma descriptif de la tâche : Récuperation des données des capteurs	22
3.5	Schéma descriptif de la tâche : Calcul des angles	23
3.6	Diagramme de classes de l'application	25
3.7	Bloque-Diagramme de simulateur	26
3.8	Diagramme de classes de simulateur	27
4.1	Beaglebone Black	29
4.2	Accéléromètre lsm303DLHC	30
4.3	Gyroscope l3G4200D	30
4.4	Capteur de pression BMP085	31
4.5	Module Xbee Serie 1	32
4.6	Explorateur USB Xbee	32
4.7	Servo-moteur SG90	33

4.8 Moteur Brushless	33
4.9 Simulink 3D Animation Tools	36
4.10 Menu de l'application	37
4.11 Interface : Coordinates visualisation	38
4.12 Interface : Gyroscopique Visualisation	38
4.13 Interface : Cockpit Instruments	39
4.14 Interface : RealTime Analysis	40
4.15 Interface : FlighGear Simulator 1	41
4.16 Interface : FlighGear Simulato 2	41
4.17 Interface : Virtual simulation	42



« Le plus grand plaisir dans la vie est de réaliser ce que les autres vous pensent incapables de réaliser. »

Walter Bagehot

INTRODUCTION GÉNÈRALE

L'informatique et la robotique sont devenues aujourd'hui comme l'une des sources les plus importantes d'innovations. Les avancées technologiques ont en effet favorisé la miniaturisation des composants électroniques et ont permis d'embarquer de l'intelligence dans tout type d'objet, donnant naissance à une nouvelle aire dans le progrès technologique, celui des systèmes embarqués.. Les systèmes embarqués sont le fruit d'une combinaison du matériel (composants électroniques et processeurs) et du logiciel (systèmes d'exploitation réduits et programmes) conçus pour répondre aux nouveaux besoins (humains, environnementaux, etc.). Ils devront, selon leurs usages et leurs principales vocations, répondre aux contraintes suivantes : réactivité, ressources limitées (mémoires, capacité de calcul), consommation et dissipation énergétiques, autonomie, mobilité, sûreté de fonctionnement, etc. Plusieurs applications basées sur la notion des systèmes embarqués, ont apparu dans des domaines divers tels que le domaine du transport (Automobile, Aéronautique), le domaine de l'astronomie (fusée, satellite artificiel, sonde spatiale), le domaine de la télécommunication (Set-top box, téléphonie, routeur, etc.). Ces applications cherchent à automatiser certaines tâches simples (appareils électroménagers, distributeur automatique) ou complexes (robot humanoïde). Dans le cadre de notre projet nous nous sommes intéressés aux systèmes embarqués appliqués à l'aéronautique. Plus spécifiquement notre projet est situé dans le cadre du développement de plates-formes robotiques volantes (drones) qui connaissent un essor croissant depuis quelques années dû non seulement à la miniaturisation permanente et plus poussée des capteurs et des actionneurs, mais aussi à la possibilité d'embarquer des cartes de commande toujours plus performantes et rapides. Ces dernières ont la capacité d'exécuter une masse de calcul considérable nécessaire au contrôle de ces engins volants. Jusqu'à présent, plusieurs plates-formes robotiques volantes ont été réalisées.

1

PRÉSENTATION DU PROJET

1.1 Etude préalable

Pour aboutir aux résultats escomptés, il faut commencer par une étude théorique approfondie. Pour ce faire, une certaine recherche s'impose dans le but de définir les différents concepts des avions. Nous allons nous intéressé dans ce premier chapitre à donner une vue générale sur les différents plates-formes aériennes et une vue particulière sur les planeurs et leurs mode de fonctionnement.

1.1.1 Plateformes aériennes

L'aéronautique est un très vaste domaine qui date depuis 1980. Ils incluent des diverses sciences par exemple l'aérodynamique, la mécanique, l'électronique et l'informatique.

les drones

Depuis plus d'un demi-siècle, des drones ont été développés dans le domaine militaire. Après quelques expériences anecdotiques, ils ont été utilisés systématiquement lors des conflits «modernes», depuis les années 80, principalement par les israéliens et les américains, et par les forces alliées lors des deux guerres du Golfe. Ces engins ont notamment été utilisés pour des missions de reconnaissance et de surveillance. Néanmoins certains peuvent être armés, de missiles ou de mini-drones. Ces avions possèdent une charge utile importante et sont équipés de caméras (visible et infrarouge) et de radars divers. Il sont reliés au sol via des communications haut-débit, habituellement via des satellites.

les véhicules aériens sans pilotes

Les véhicules aériens sans pilote (UAV) ou les drones sont des aéronefs sans pilote humain à bord. L'apparition de ces machines capables de décoller et d'atterrir sans pilote remonte à la préhistoire de l'aéronautique. La première tentative a eu lieu en Italie en 1849 suite à des bombardements sur Venise moyennant des ballons sans pilote munis de bombes à retardement. Dès cette tentative des nombreuses améliorations ont suivi et ont donné naissance à plusieurs types de drones. Ces machines sont généralement utilisées pour accéder à des zones à risque voir même inaccessible par les humains ou par les véhicules ordinaires : réaliser des missions de renseignement stratégique, des missions de détection, de surveillance ou de patrouille maritime. Cependant les drones ne sont pas cantonnés à des fins militaires, ils peuvent également être utiles pour un usage civil pour réaliser des prises de vues aériennes ou par exemple la surveillance du trafic routier. Nous distinguons différentes catégories de drones suivant leurs dimensions et leurs missions : Les drones stratégiques comme (drone MALE) représente un drone volant à moyenne altitude inférieur à 1000km et à grande autonomie. Traditionnellement utilisés pour un renforcement de renseignement (écoute des signaux électromagnétiques), la recherche ou le sauvetage (SAR). Quant au drone HALE, volant à haute altitude et à grande autonomie, il est destiné pour réaliser des missions de renseignement stratégiques ou à la détection de missiles balistiques grâce à une alerte avancée. Les drones tactiques, comme Sagem, sont utilisés essentiellement au profit des forces terrestres, pour des missions de sécurité et pour la localisation des cibles pour l'artillerie. Les drones de combat UCAV : Ce type de drone est programmé pour effectuer des missions de combat sans intervention humaine.

1.1.2 Les planeurs

Un planeur est un aérodyne qui est par définition dépourvu de moteur il existe toutefois des versions dotées d'un moteur d'appoint escamotable appelé moto-planeur, la pratique du planeur est le vol à voil. D'une manière générale ,l'une L'une des caractéristiques principales du planeur est sa finesse qui se traduit, par la capacité de parcourir une grande distance en perdant un minimum d'altitude.

Principe de vol

Le planeur est plus lourd que l'air. Pour qu'il vole, il faut faire apparaître une force qui s'oppose à son propre poids. Cette force est une résultante de la résistance de l'air qui se développe sur les surfaces portantes lorsque le planeur adopte une trajectoire descendante appropriée Imaginez une bille posée sur une

planche. Inclinez la planche et la bille, par son poids, roule. Pour un planeur, le principe est le même ; il lui faut un certain angle à piquer.

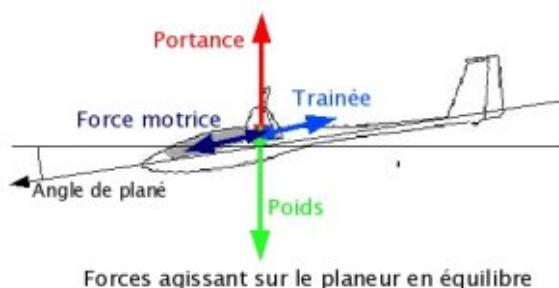


FIGURE 1.1 – Forces agissant sur le planeur en équilibre

En plus de ça le planeur est un avion qui vole grâce à la force des courants atmosphériques qui agit énormément au niveau de ailes :

Profil d'une aile de planeur



FIGURE 1.2 – Profil d'une aile de planeur

Le schéma montre en coupe le profil d'une aile. La partie supérieure est l'extrados, la partie inférieure est l'intrados. La forme du profil est définie par calcul et par expérience en soufflerie en fonction de l'usage et du type de performances souhaitées.

En vol, sous l'effet de sa vitesse, le planeur est soumis à un flux d'air appelé vent relatif. Ce flux se sépare de part et d'autre de l'aile. On voit sur le schéma que les particules d'air qui contournent l'aile par l'extrados parcourent un trajet plus long que celles qui passent sous l'intrados.

Ces particules seront donc accélérées, une dépression se crée sur l'extrados et aspire l'aile vers le haut. Dans le même temps, l'air qui s'écoule sous l'intrados exerce une pression qui soulevera l'aile.

La force engendrée par la dépression sur l'extrados est beaucoup plus importante que celle générée par la pression sur l'intrados. Même si ces deux forces s'exercent dans la même direction et contribuent ensemble à créer la portance, le

planeur est plus aspiré que porté par la résistance de l'air.

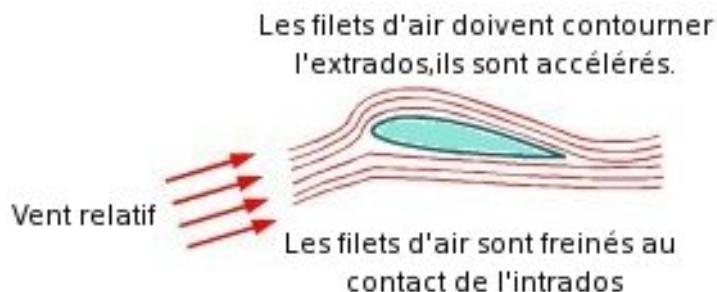


FIGURE 1.3 – actions subies par l'aile

Exemples des planeurs autonomes

Super Dimona, est un moto planeur chez BMI, en version électrique accus 4000mAh 4S avec moteur de 600 w hélice 11.7 poids en ordre de vol env 2 Kg, surface alaire 40,34 dm², fuselage fibre, aile coffré.



FIGURE 1.4 – Super-Dimona

Le Glaser-Dirks DG-400 est un motoplaneur monoplace fabriqué par Glaser-Dirks entre 1981 et 1990. Il a été le premier motoplaneur à dispositif rétractable produit à grande échelle, il est caractérisé par un moteur rotax 505 de puissance 32 kW et il peut avoir 380 km d'autonomie en vol en dauphin



FIGURE 1.5 – DG-400

1.2 Etat de l'art de l'asservissement

Dans le domaine de l'aéronautique , on veille toujours à améliorer le temps d'executions des consignes ainsi que la précision des actions souhaité . En outre , notre travail nécessite de concevoir un régulateur pour la trajectoire rectiligne de l'aéronef . Notre besoin alors est de trouver un mode d'asservissement fiable et non complex pour le commande en alttitude et en rotation angulaire de l'aeronef . En effet , Il existe différentes techniques pour synthétiser les régulateurs. La technique industrielle la plus largement utilisée est le régulateur PID qui calcule une action Proportionnelle, Intégrale et Dérivée en fonction de l'erreur consigne/mesure. Cette technique permet de satisfaire la régulation de la majorité des procédés industriels. La commande à modèle interne , généralisation des régulateurs PI ou PID avec prédicteur de Smith , offre beaucoup plus de possibilités et est également répandue. Des techniques avancées se basent sur la commande par retour d'état (ou commande par retour d'état reconstruit par un observateur). Ces types de commande peuvent être conçus par placement de pôles ou (pour ce qui concerne les systèmes d'état) par minimisation d'un critère quadratique prenons en guise d'exemple la commande LQ ou LQG. Deplus on trouve La commande prédictive se basant sur l'utilisation d'un modèle dynamique du système pour anticiper son comportement futur et la commande robuste permettant de garantir la stabilité par rapport aux perturbations et aux erreurs de modèle . Notre choix semble décisif dans le choix du régulateur , l'asservissement PID est le moins complex à implémenter et présente une varieté d'utilitaire sur marché qui permette d'optimiser notre travail tel que les PID tuners .

Le correcteur PID agit de trois manières :

- ◊ action proportionnelle : l'erreur est multipliée par un gain G ;

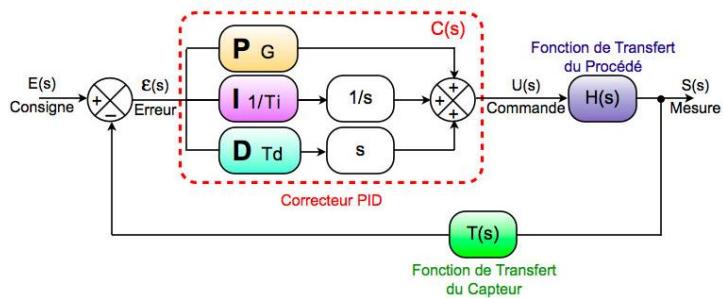


FIGURE 1.6 – l’asservissement avec PID

- ◊ action intégrale : l’erreur est intégrée et divisée par un gain T_i ;
- ◊ action dérivée : l’erreur est dérivée et multipliée par un gain T_d .

1.3 Etude technique

Notre choix de microcontrôleur c'est établie sur le Beaglebone Black rev C. Ce choix est justifié par sa performance ; ce microcontrôleur possède un processeur Sitara ARM Cortex- A8 de fréquence 1GHz 2000 mips et une mémoire DDR3L de 512Mega-octet. Le Sitara AM3358BZCZ100 dispose aussi de deux PRU 32bits Risc doté d'une fréquence de 200Mhz chacune qui est un atout pour les besoins de programmation temps réel . Le beaglebone black offre une variété de périphériques. Il inclut une multitude d’interfaces de communication dont six UART, deux SPI, deux interfaces CAN , et deux interfaces USB .Déplus Le BeagleBone est un environnement fertile pour les projets basé sur les système linux embarqué .En outre L’architecture interne du Beaglebone supporte plusieurs système d’exploitation comme Debian GNU/linux et freeRT .Les dimensions réduites ainsi que le poids du Beaglebone sont adéquate pour nos besoin de minimisation de la masse totale de l'aéronef . Somme toute , ces performances inégales ainsi que l’existence de plusieurs produits de développement et de recherche supportant ce microcontrôleur comme Matlab a renforcé notre choix : en effet la nature de notre projet exige un contrôle en temps réel et une exécution de plusieurs tâches en parallèle .

	Processor	SDRAM	Flash	Périphérique	Indicateur	Puissance	Environnement	
<u>Mega2560</u>	-Microcontrôleur ATmega2560	SRAM 8ko	-256 ko	14PWM 16IA 4UART SPI,I ² C	- 101,6x53,3 mm - 60 g	-5-3,3VDC -Up to 200mA	Arduino Software Development Environment (IDE)	
<u>STM32F4 Discovery</u>	-STM32F407VG-T6 ARM CORTEX M4-based MCU + FPU core 168Mhz/210DMIPS	192+4 Ko	-1Mo	17 TIMS,3ADC,15 interface de communication , LIS302DL ou LIS3DSH ST MEMS 3-axis accelerometre, UART,SPI,I ² C,DCMI PWM,I ² S audio ...	-97x 66 mm -60g	-5-3,3VDC -44mA	Tools : IAR EWARM Keil MDK-ARM™ GCC-based IDE OS : ChibiOS/RT	
<u>Ni myRio-1900</u>	-Xilinx Z-7010 667 MHz 2 core	512Mo DDR3 533MHz	-256 Mo Non volatile	Accéléromètre UART SPI,I ² C,PWM	- 136,6x88,6 mm - 193g	-5VDC up to 15 VDC - 32..500mA (2,6..14W)	FPGA Xilinx Z-7010 RT programming	
<u>Raspberry Pi B+</u>	-700 MHz ARM1176JZF-S core (ARM11)	512Mo (intégré avec GPU)	-MicroSD	17 x GPIO,UART,I ² C bus,SPI bus avec deux chip select,I ² S audio	-85,60 x 53,98 mm -45g	-5VDC - up to 600 mA (3W)	Debian ,GNU/linux ,RaspbianOS, Fedora, Kali linux,Android ..	
<u>CubieBoard</u>	-ALLWINNERA10 ARM Cortex-A8 1 GHz CPU -Mali400, OpenGL ES GPU	512 Mo ou 1 Go DDR3 480 MHz	-4 GoNAND Flash - SDHC - SATA II	96 extend pin interface, including I ² C, SPI, RGB/LVDS, CSI/TS, FM-IN, ADC, CVBS, VGA, SPDIF-OUT, R-TP,...	-100x60 mm	-5VDC -up to 2A	Android ,GNU/linux (Cubian,Debian, Ubuntu ...) ..	
<u>Beaglebone Black rev C</u>	-Sitara AM3358BZCZ100 1GHz 2000MIPS -"2xPRU pour RT"	512Mo DDR3L 800MHz	-4Gb EMMC -MicroSD	4xUART 8x PWM, LCD, GPMC, MMC1, 2xSPI, 2xI ² C, A/D Converter, 2xCAN Bus , 4 Timers	-86,63x 53,34mm -39,68g	-5VDC - 210-460mA	Debian GNU/linux,windows Android,freeRTOS..	

FIGURE 1.7 – Comparaison des microcontrôleurs

2 | ANALYSE ET SPÉCIFICATION DES BESOINS

2.1 Spécification des besoins

Cette partie sera consacrée à l'analyse et à la spécification des besoins, les fonctionnalités dont nous avons besoin pour définir le système seront décrites au terme de besoins fonctionnels. Nous allons aussi illustrer les champs d'amélioration possible de notre système dans la partie spécification de besoins non fonctionnels.

2.1.1 Besoins fonctionnels

Afin de satisfaire les exigences du pilote, le drone doit effectuer un vol stable qui suit d'une manière exacte un itinéraire rectiligne en prenant comme consigne la destination et l'altitude de vol . Le drone devra répondre en temps réel aux commandes envoyées par le pilote. Dans le cas d'absence des signaux de commande, le plateforme volant doit conserver son itinéraire initial avec une conservation d'altitude . Si le délai de la mission encourue dépasse le délai théorique déjà calculé le drone devrait effectuer un atterrissage d'urgence . Comme application assuré par le drone, la mesure de la pression ainsi que la température le long de sa trajectoire avec de potentiel capture . La sécurité et la tolérance aux pannes devront être prise en compte dans la construction de l'engin . La station de commande devrait se disposer d'un mécanisme spécifique responsable à l'abondement de toute mission . Ainsi, les besoins fonctionnels pour le pilote se résume en cinq points :

Besoins du planeur :

- ◊ Le planeur doit effectuer un vol stable.
- ◊ La réponse du système aux commandes et le traitement des données captées doivent être en temps réel.

- ◊ La communication entre le planeur et la station de commande devrait être sécurisée .
- ◊ Le planeur devras peut etre commandé manuellement à l'aide d'une manette .

Besoins de l'application :

- ◊ L'utilisateur doit introduire le point de départ et de destination du mouvement du système .
- ◊ l'application permet de visualiser le mouvement de système à l'aide d'une animation en 3D.
- ◊ l'application doit afficher les données reçues des capteurs moyennant une interface " Cockpit instruments".
- ◊ l'utilisateur peut suivre les inclinaison effectué par les ailes , pitch et yaw du planeur via cette application

2.1.2 Besoins non fonctionnels

Outre que les besoins fonctionnels cités, notre plateforme volant doit assurer de nouveaux services ayant pour objectif la suivie de l'état et la position du drone. Dans ce cadre, nous comptons introduire une interface de visualisation temps réel de l'évolution cinématique de l'aéronefs autonomes. En effet nous espérons nous profiter des données récupérées par l'IMU et le capteur GPS pour déduire les déplacements du drone selon les différent axes spacieux-temporels des repères relatifs et absous fixées à l'avance . ainsi les besoins demandés.

Besoins du planeur :

- ◊ Le système doit résister contre certaines constraintes de l'environnement externe.
- ◊ le planeur ne doit pas depasser la zone de couverture de la communication distante.
- ◊ le poids du planeur ne doit pas gêner le mouvement du système.

Besoins de l'application :

- ◊ l'application doit rester fonctionnel indépendamment du mode de pilotage du planeur.
- ◊ Les interfaces de l'application doivent avoir une clarté pour bien renseigner l'utilisateur.
- ◊ L'application doit notifier l'utilisateur en cas de perte de communication avec le planeur .

2.2 Description des fonctionnalités et scénarios

2.2.1 Diagramme de cas d'utilisation

Le diagramme de la figure 2.1 décrit les différents services offerts à l'utilisateur et les ensembles de fonctionnalités qui peuvent être passées à l'autopilote.

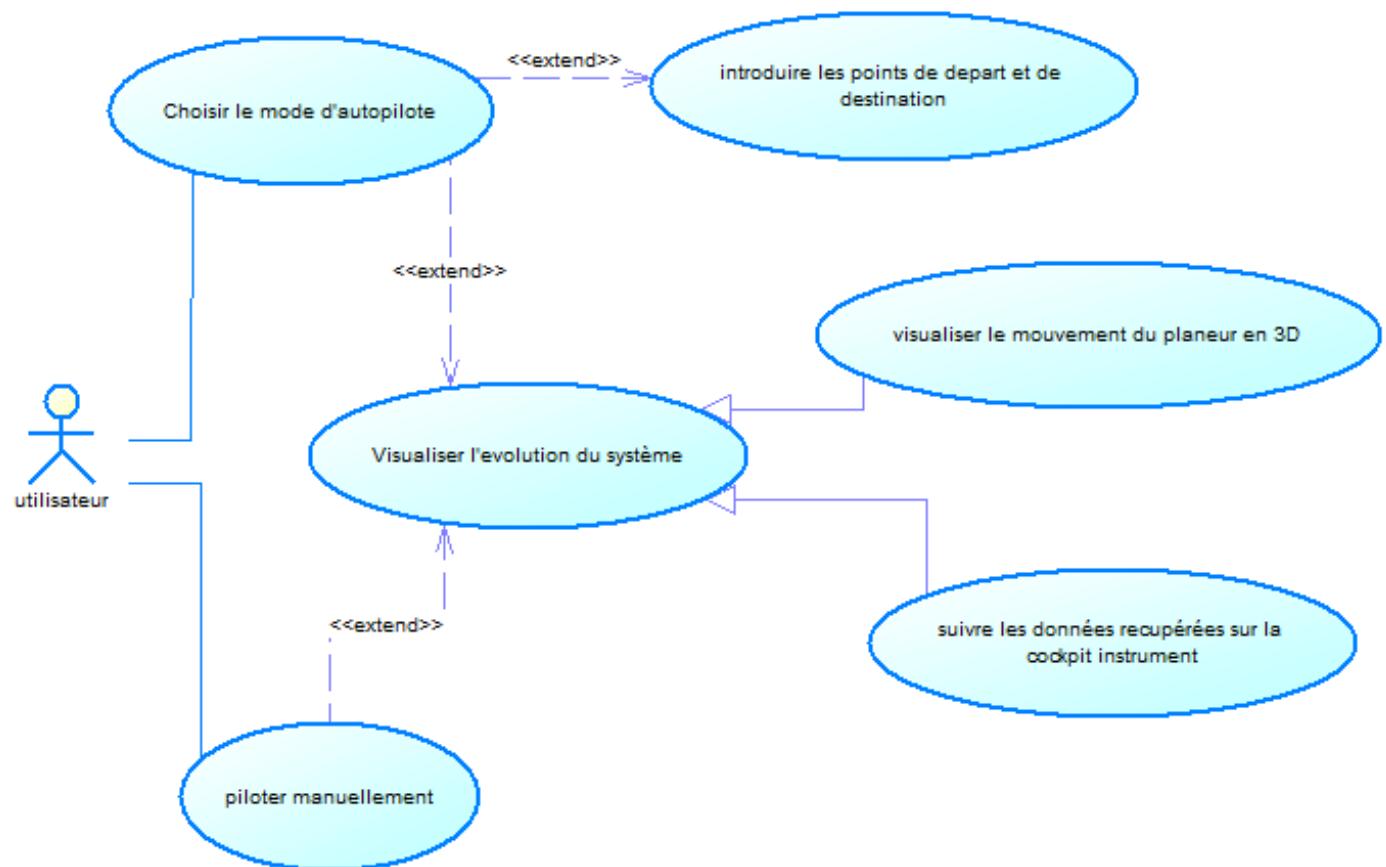


FIGURE 2.1 – Diag des cas d'utilisations

2.2.2 Diagramme de séquence

Ces diagrammes modélisent les scénarios nominaux de système pour le pilotage manuel et le mode autopilote, c'est une description dans l'ordre chronologique de l'ensemble des services et des messages échangés entre l'utilisateur et le planeur.

Pilotage manuel :

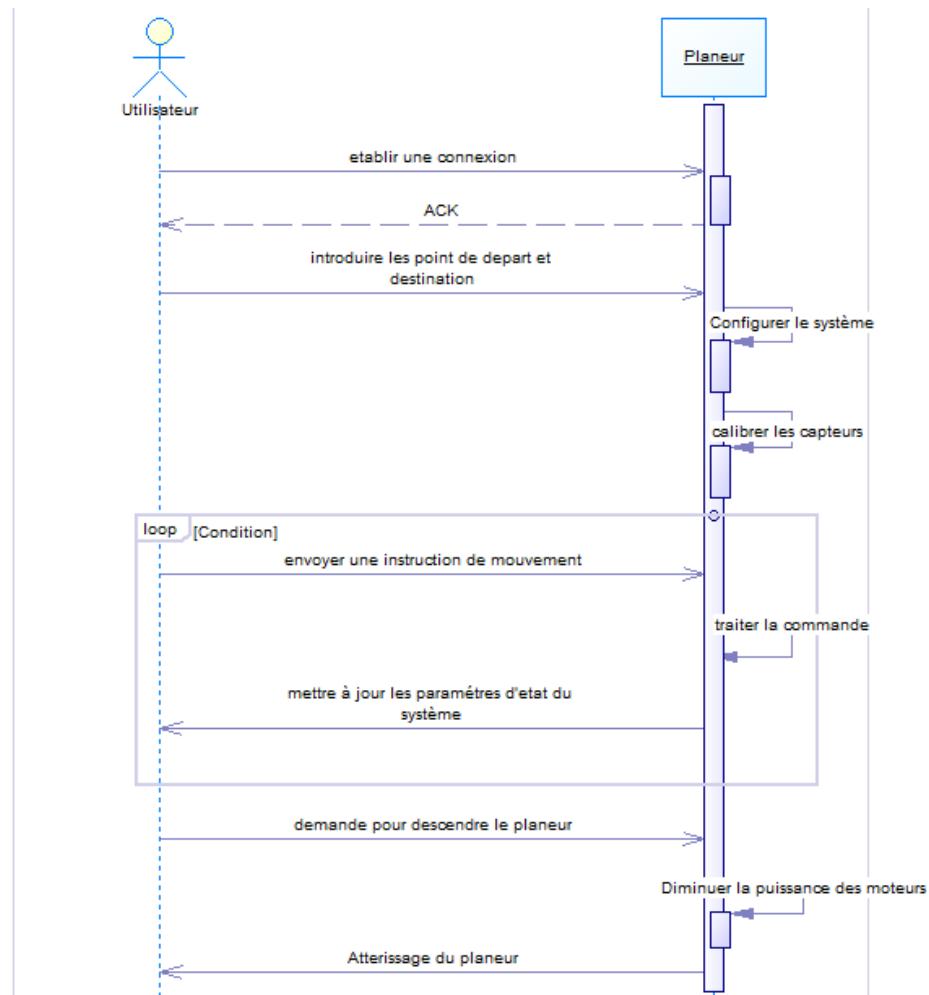


FIGURE 2.2 – Diagramme de séquence : pilotage manuel

pilotage automatique :

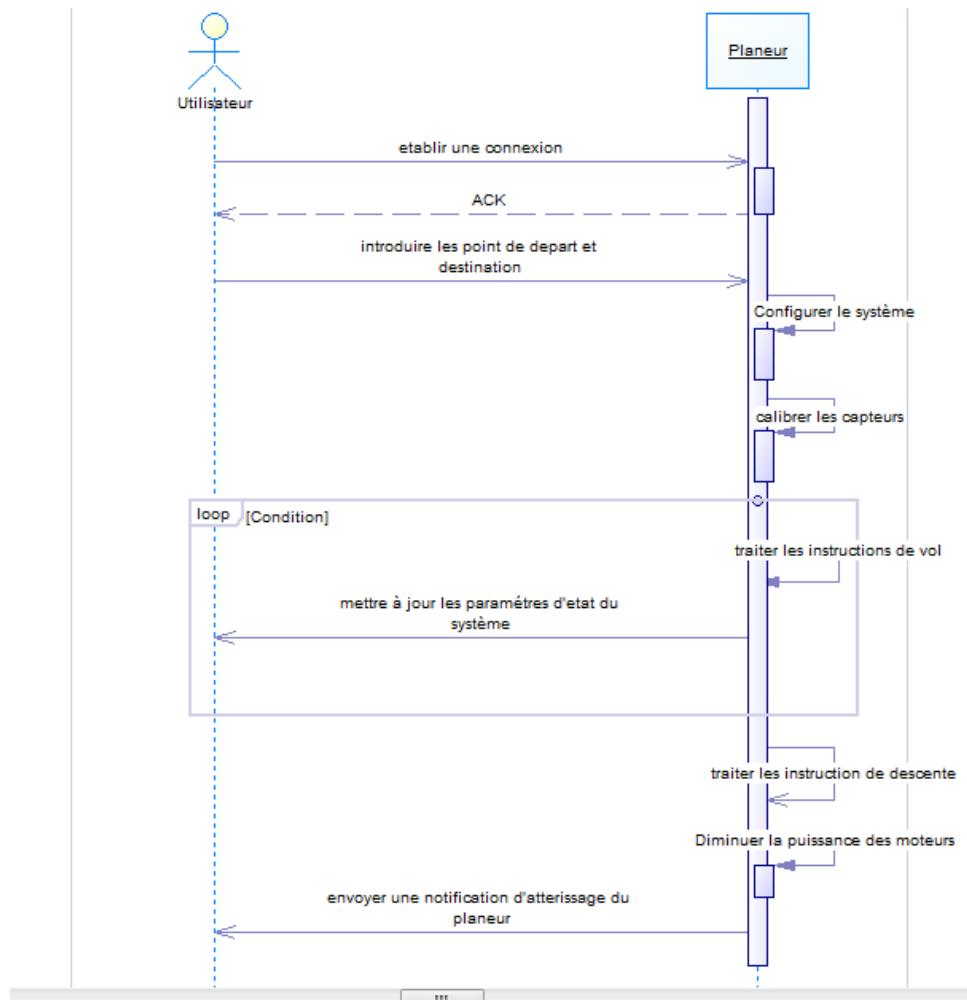


FIGURE 2.3 – Diagramme de séquence : pilotage automatique

3 | CONCEPTION

Afin de rendre le développement, notre système plus fidèle aux besoins fonctionnels et non fonctionnels présentés dans le chapitre précédent, une bonne conception s'avère primordiale avant la réalisation. En premier lieu, notre intérêt va porter sur la conception globale. En deuxième lieu, nous présenterons la conception détaillée.

3.1 Conception globale

Dans cette section, nous allons présenter notre système de façon générique en définissant l'architecture de notre projet. Dans le but de détailler notre système, nous présentons la figure 3.1 qui illustre les principaux compartiments de notre système. En fait, notre système se divise en trois parties ; une partie embarquée, une partie logicielle, un et une partie de commande qui devient accessible en cas de coix de mode pilotage manuelle . La partie embarquée dans notre planeur est la partie opérative qui englobe toute configuration au niveau matériel de notre système. La partie logicielle qui se présente sur est une application qui offre à l'utilisateur des interface interactives pour contrôler le planeur et avoir plus de détails sur son état en mode de vol. On présente aussi une station de commande pour notre système, cette dernière offre à l'utilisateur une méthode de controler le planeur via une manette afin de faciliter la tache de pilotage.

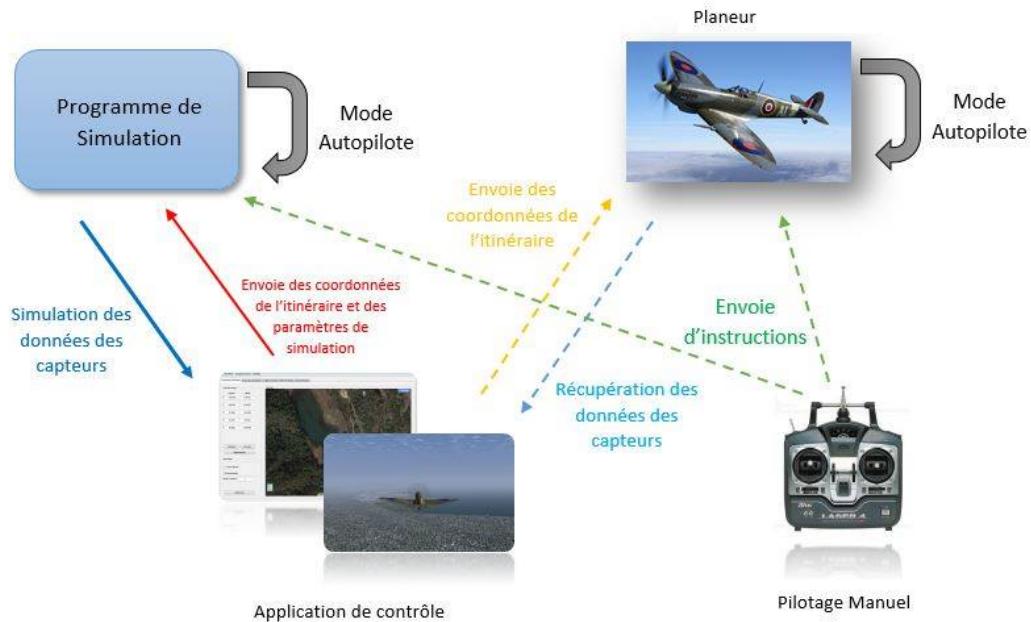


FIGURE 3.1 – schéma descriptif de l'architecture du système

3.2 Conception détaillée

Dans cette étape, nous allons développer les différents modules de notre système. En premier lieu, nous présentons la conception du planeur dans son environnement logiciel et matériel. En deuxième lieu, nous abordons la conception de l'application en décrivant le diagramme des classes qui la composent et puis nous finirons par la partie applicative qui représente la station de commande de notre système.

3.2.1 Conception du planeur

Architecture matérielle

Dans cette partie, nous présentons l'architecture matérielle du planeur. La figure 3.2 introduit notre système en spécifiant ces différents composants ainsi que la façon avec laquelle ils interagissent avec le microcontrôleur.

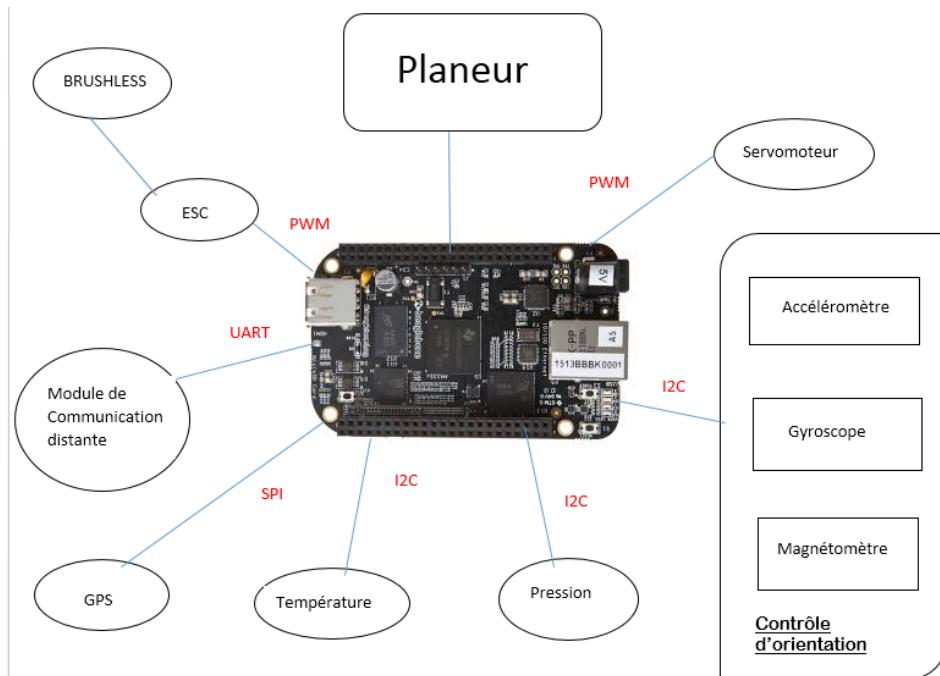


FIGURE 3.2 – Architecture matérielle

Architecture logicielle

Cette section concerne la programmation du microcontrôleur. Plusieurs systèmes embarqués nécessitent une réponse en temps réel. De ce fait, nous avons pensé à utiliser un système d'exploitation temps réel pour notre planeur. Il existe une panoplie de système d'exploitation temps réel ayant prouvé leurs efficacités et leurs robustesses pour la plupart des applications embarquées. L'atout majeur de ces systèmes est la programmation multitâche. De ce fait, nous pouvons assimiler notre logiciel embarqué à un ensemble de tâches comme le montre la figure 3.3. Notre logiciel se décompose donc de trois tâches principaux dont chacune d'entre elles offre un service à la tâche qui le suit.

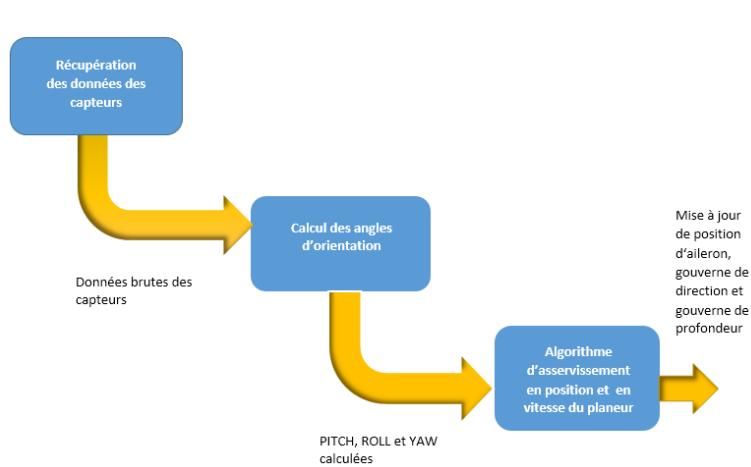


FIGURE 3.3 – les tâches de la partie logicielle embarqué

La première tâche de la récupération des données des capteurs est en d'autres termes la configuration de l'accéléromètre, gyroscope et magnétomètre et la lecture des registres qui contiennent des données (figure 3.4). Ces grandeurs physiques vont être traduites en trois angles qui seront par la suite les contrôleurs de mouvements du planeur. Ces derniers décrivent l'orientation de l'engin dans un repère absolu.

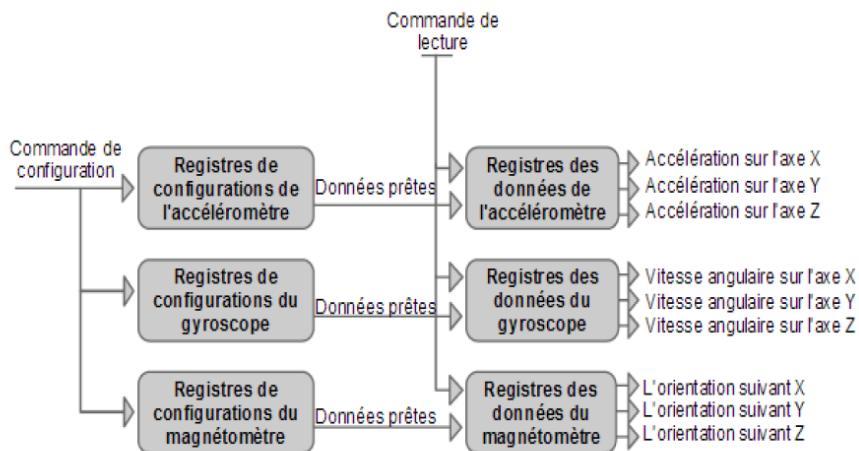


FIGURE 3.4 – Schéma descriptif de la tache : Récuperation des données des capteurs

En fait, seul l'accéléromètre peut nous donner ces différents angles, sauf que ce type de capteur mesure les accélérations uniquement sur les axes, d'autres accélérations hors l'accélération gravitationnelle peuvent entraîner un jeu qui va perturber notre calcul. Un autre capteur optionnel qui peut être ajouté à notre système nommé le magnétomètre. Avec les données récupérées de ce type de capteur, il est possible de repérer notre système dans un repère absolu.

La figure 3.5 décrit le processus de calcul de ces angles. Étant donné que ce calcul est basé sur les lectures provenant de l'accéléromètre, un autre type de capteur s'avère utile pour remédier à ce problème appelé le gyroscope. En effet, le gyroscope est un capteur qui mesure la vitesse angulaire, ce qui nous permet de calculer les trois angles relatifs du plan du planeur.

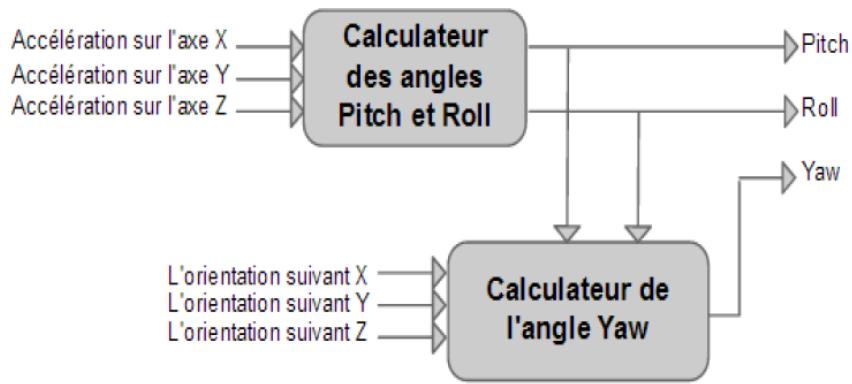


FIGURE 3.5 – Schéma descriptif de la tâche : Calcul des angles

Tout au long de cette partie, nous avons présenté le processus de l'extraction des données qui vont nous servir à maintenir un vol stable de notre quadricoptère. Il reste à ajouter un algorithme de régulation de position dans le but d'avoir plus de précision, nous avons eu recours à un régulateur PID ;

3.2.2 Conception de l'application de contrôle

La partie logicielle se compose d'une application qui offre à l'utilisateur des interfaces interactives pour définir le trajet du planeur et avoir tous les détails sur son état en mode de vol ou en mode simulation. L'application permet aussi à l'utilisateur de se connecter à un simulateur, choisir différents paramètres d'autopilotes et tester son comportement dans un monde virtuel. On présente aussi une station de commande pour notre système, cette dernière offre à l'utilisateur une méthode de contrôler le planeur via une manette sans fil afin de faciliter la tâche de pilotage et prendre le relais en cas de problème avec le système d'autopilote.

Diagramme de classes

La figure illustre le diagramme ds classes de notre application avec ses différentes interfaces. Tout d'abord on a la classe Data Acquisition qui récupère les valeurs de ses attributs soit à partir de la classe Serial Acquisition soit à partir de la classe Simulator Acquisition selon la valeur de son attribut Acquisition mode. En suite les classes Coordinates Visualisation, Gyroscopic Visualisation Cockpit Instruments, Real Time Analysis et FlightGear interface vont récupérer à chaque fois les valeurs des attributs de l'objet Data Acquisition et mettre à jours leurs attributs pour modifiée les affichages associés et permettre à l'utilisateur de percevoir les changements de l'état du planeur ou de la simulation. La classe Coordinates Visualisation exploite les données introduites par l'utilisateur et fait passer les attributs depart,destination et itinéraire au microcontrôleur et au simulateur. Finalement la classe Virtual simulation exploite les données introduites par l'utilisateur et les fait passer à la classe FlightGear interface et au simulateur.

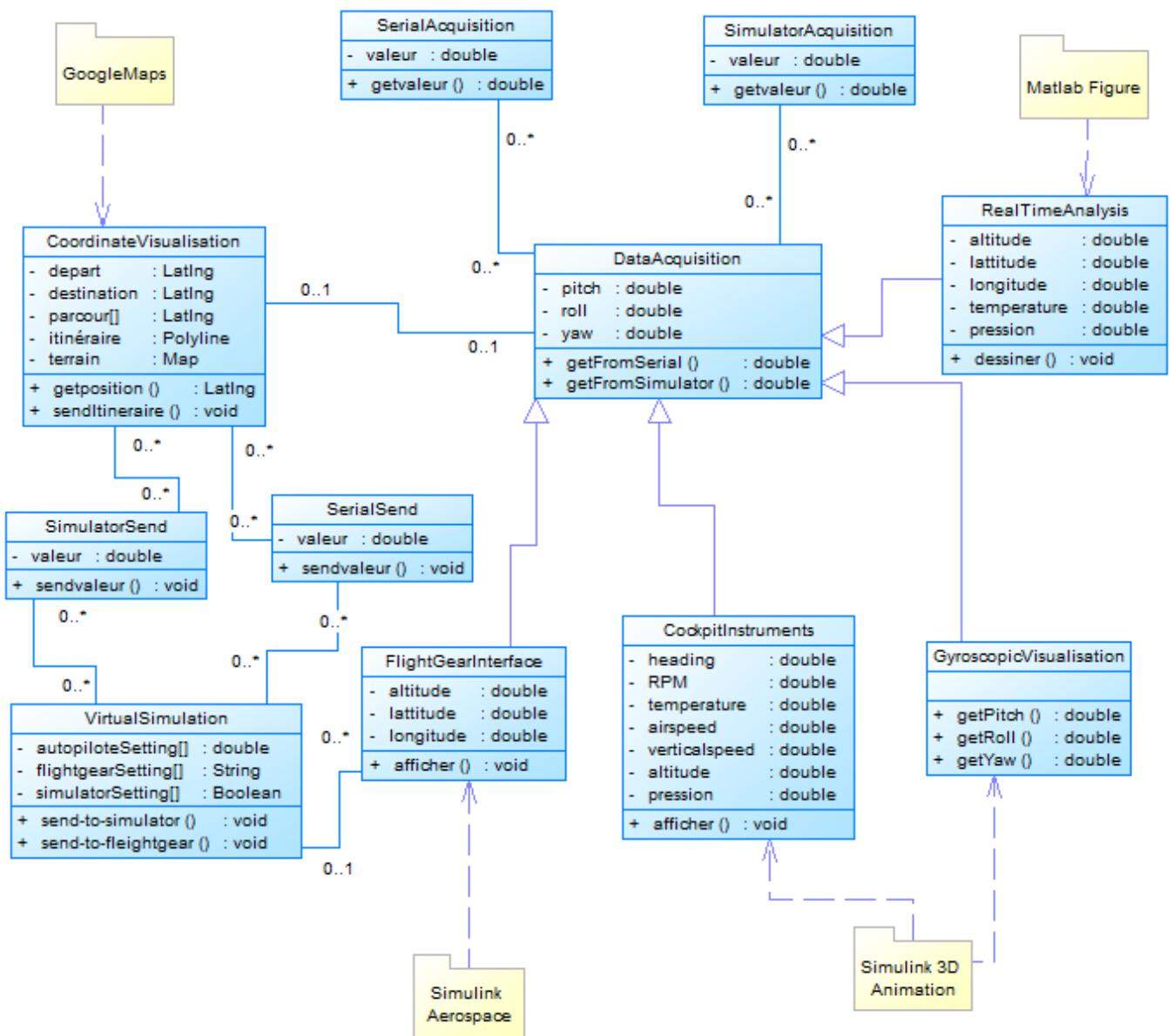


FIGURE 3.6 – Diagramme de classes de l’application

3.2.3 Conception de simulateur

Description

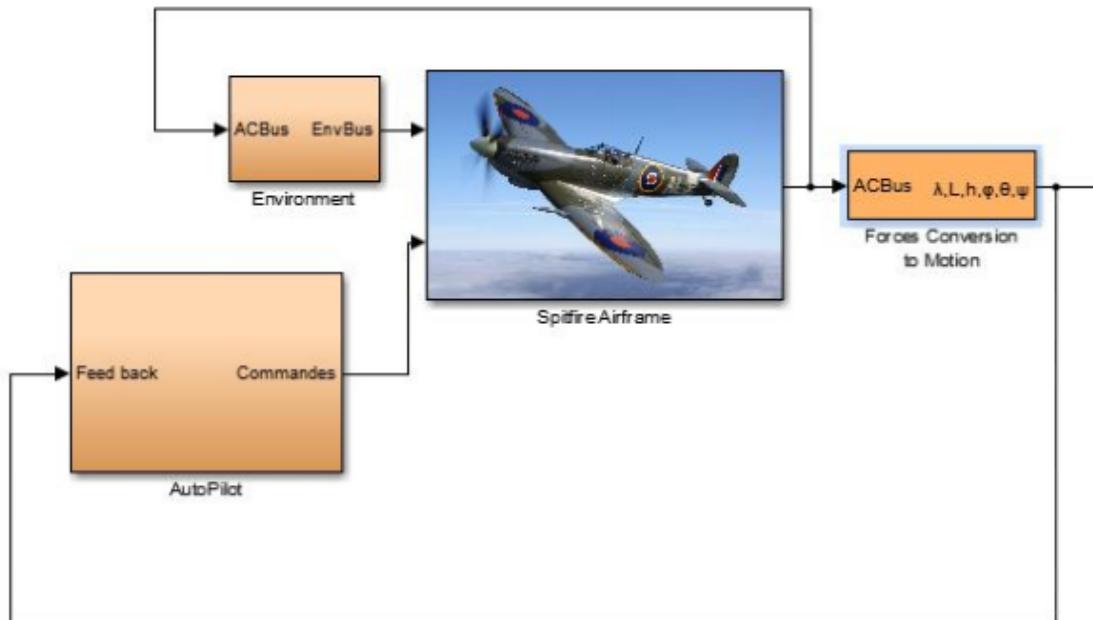


FIGURE 3.7 – Bloque-Diagramme de simulateur

cet une description du notre simulateur sous forme d'un bloc-diagramme qui est constitué de 4 compartiments :

- le bloc Autopilot : ce block fournit les commandes au modèle du planeur (positions des ailerons, rudder, elevators ainsi que la vitesse du moteur) en se basant sur un algorithme d'asservissement choisis par l'utilisateur.
- le bloc Environnement : Réalise la simulation de l'effet de la gravité, l'atmosphère et le vent en se basant sur des équations mathématiques fournis par l'AeroSpace Toolbox de simulink.
- Spitfire Airframe : Ce block réalise le calcul des sommes des moments et des forces exercés sur le planeur en prenant en compte les différents coefficients aérodynamiques et le poids du modèle géométrique de l'avion, ainsi que les caractéristiques du moteur et de l'hélice.
- Forces Conversion to Motion : Ce block interprète les sorties du block Spitfire Airframe sous forme de trois angles d'Euler (pitch, roll, yaw) ainsi que la nouvelle position du planeur dans le repère terrestre (altitude, latitude, longitude).

Diagramme de classes

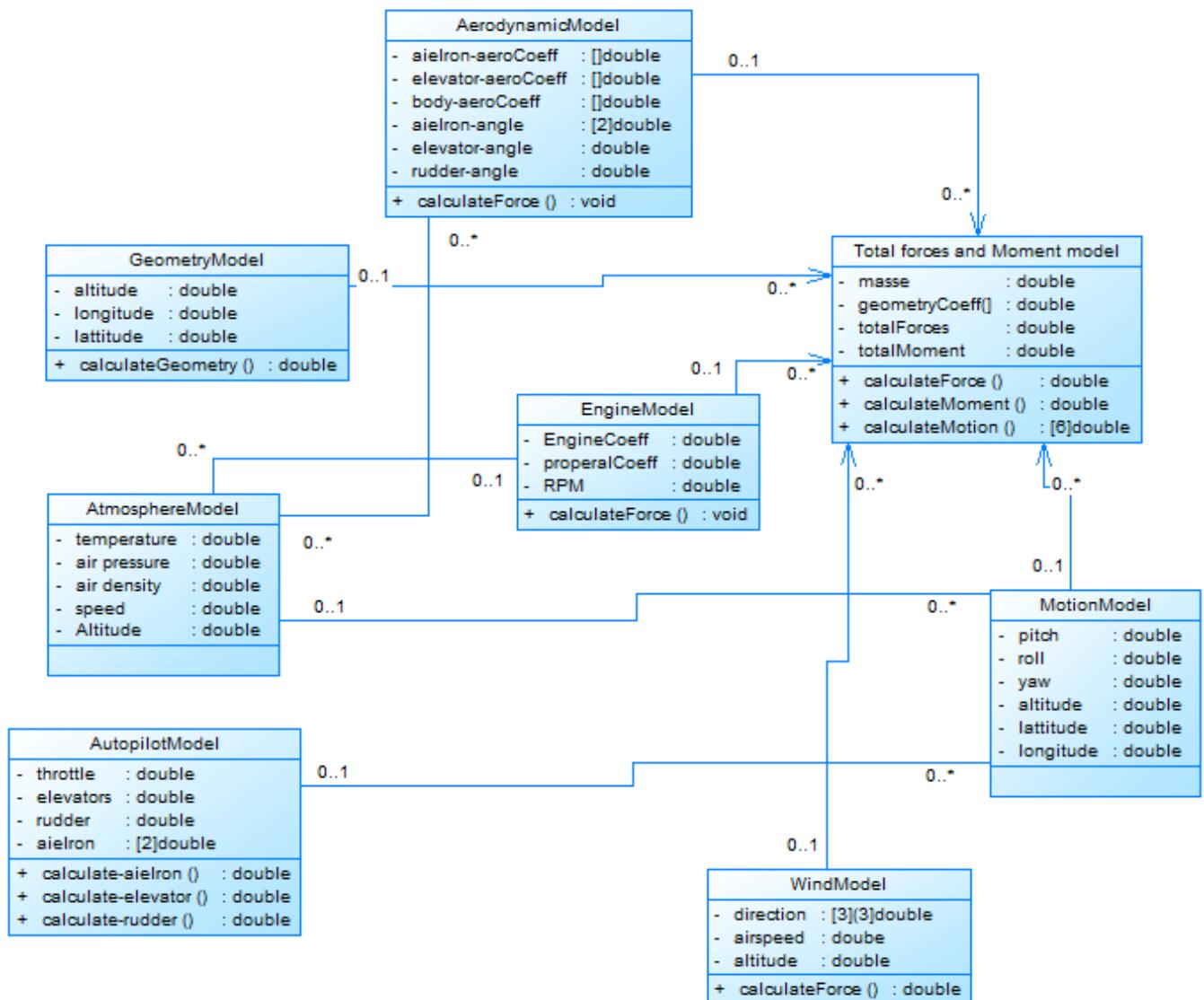


FIGURE 3.8 – Diagramme de classes de simulateur

4 | RÉALISATION

4.1 Réaliation de la partie matérielle

4.1.1 Choix de microcontrôleur

Comme il est connu dans le domaine embarqué le microcontrôleur constitue le cerveau du système ; il est considéré comme le noyau principal qui doit tout gérer (récupération de données, exécution des algorithmes de calcul, génération de signal de commande...) d'où l'importance de son choix selon le type de l'application. Dans notre projet nous avons décidé d'utiliser le microcontrôleur BeagleBone Black présenté par la figure , Ce choix est justifié par sa performance en faite ;BeagleBone Black est un mini-ordinateur Linux Open-Source Android de la taille d'une carte bancaire. Il est équipé d'un puissant processeur TI SitaraTM AM335x ARM CortexTM de 1 GHz, d'une interface HDMI, d'une connexion Ethernet 10/100 et d'une mémoire vive de 512 Mo. Grâce aux nombreuses entrées/-sorties et une puissance de calcul pour des analyses en temps réel, le BeagleBone Black convient parfaitement pour la technique de commande et d'automatisation.

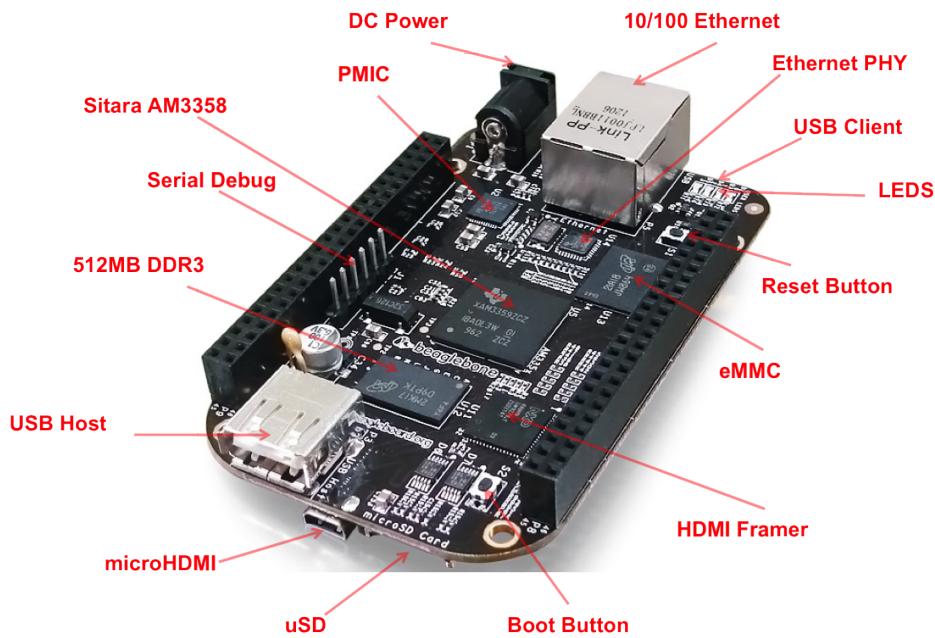


FIGURE 4.1 – Beaglebone Black

4.1.2 choix de capteurs

Capteurs d'orientation : Acceloromètre/ gyroscope/ Magnétomètre

Le contrôle du planeur est basé principalement sur des données récupérées à partir des capteurs d'orientation c'est ce qui justifie leur choix très décisif. Ce capteur dépend énormément de type de l'application dont il et la marge d'erreur tolérée. Dans notre cas, notre projet exige une précision élevée d'où l'impossibilité d'utiliser un seul capteur (gyroscope ou accéléromètre) comme source de données. Nous devons donc penser à une centrale inertie (IMU) qui combine les résultats de gyroscope et d'accéléromètre pour déduire les angles d'orientation. En avançant un peu plus dans notre recherche, nous avons rendu compte que même avec le gyroscope et l'accéléromètre il existe certains mouvements tellement rapide que la centrale inertie retourne des résultats erronés. Nous avons donc penser à intégrer un magnétomètre pour surmonter ce problème. Afin de répondre à ces exigences notre choix a été fixé sur le 9 Degrees of Freedom-Sensor Stick illustré par la figure : un circuit imprimé qui contient un accéléromètre 3 axes lsm303DLHC un gyroscope 3 axes l3G4200D et un magétomètre 3 axes HMC5883L. Cet IMU est caractérisé par sa haute précision comme le montre la description de chaque MEMS ci-dessous. *L'accéléromètre lsm303DLHC : possède une haute résolution avec un codage de données sur 13 bit, une marge de mesure entre -16g et 16g. Il



FIGURE 4.2 – Accelormètre lsm303DLHC



FIGURE 4.3 – Gyroscope l3G4200D

est capable de mesurer une inclinaison de 1 degré. * Le gyroscope l3G4200D : les données sont codées sur 16bit, il est capable de mesurer une vitesse angulaire de 2000 degrée/s. * Le magnétomètre HMC5883L : le codage des mesures se fait sur 12 bit capable de mesurer une déviation de 1 degré à 2 degré. La récupération de données de capteur se fait via un interface de communication I2C.

L'accéléromètre permet de savoir dans quelle direction le planeur se déplace ; il ne détecte pas une position, mais une accélération sur l'un des trois axes X, Y, Z. Un accéléromètre mesure les accélérations, mesure les changements de vitesse et les changements de position (mouvements de translation). Ils sont généralement utilisés pour mesurer de petits mouvements.

Capteur température :

Les sondes de température (ou capteurs de température) sont des dispositifs permettant de transformer l'effet du réchauffement ou du refroidissement sur leurs composants en signal électrique.

capteur de pression barométrique :

Un capteur de pression (ou sonde de pression) est un dispositif destiné à convertir les variations de pression en variations de tension électrique. Lorsque la sonde est reliée à un système numérique, les variations analogiques sont d'abord converties en signaux numériques binaires par un convertisseur analogique-numérique avant d'être transmises à la station de contrôle et de gestion. Ce capteur nous permet de déterminer l'altitude du quadricoptère avec une précision d'environ 25cm. L'acquisition des données de ces capteurs se fait via l'interface I2C.



FIGURE 4.4 – Capteur de pression BMP085

4.1.3 Module de communication distante

La commande à distance de planeur exige l'ajout d'un module de communication wifi. Généralement les constructeurs utilisent des kits radio pour le pilotage ,ce kit se compose d'un module récepteur qui s'ajoute au plate-forme volant et une manette de commande qui possède un ensemble de canal pour l'envoi de signaux. Cette solution ne convient pas à notre application vu que nous devons récupérer des données envoyées par le drone et que ses données peuvent être reçus par l'ordinateur pour le besoin de simulation. Nous avons ainsi pensé au XBee [N8] figure 4.5 , un module simple et fiable à mettre en oeuvre, il possède une portée importante : jusqu'à 30 m dans un milieu urbain et jusqu'à 100 m dans un milieu champs libre. Un autre avantage des modules XBee est la facilité de communiquer avec le PC grâce aux cartes explorateur USB XBee figure 4.6 . Le module XBee illustre le protocole de communication Zigbee qui est un protocole de haut niveau permettant la communication de petites radios, à consommation réduite, basée sur la norme IEEE 802.15.4 pour les réseaux à dimension personnelle.



FIGURE 4.5 – Module Xbee Serie 1



FIGURE 4.6 – Explorateur USB Xbee

4.1.4 Choix des actionneurs

Les servo-moteurs

Les servomoteurs permettent de déplacer un bras, sur lequel est fixé un objet, jusqu'à une certaine position (ou angle de rotation), puis à maintenir solidement cette position. Le terme même de servomoteur signifie qu'il s'agit d'un moteur asservi, obéissant à une commande externe. Contrairement à un moteur CC simple, qui peut être piloté par des variations de tension ou par allumage/extinction, le servomoteur réagit en fonction d'une impulsion de durée variable. C'est la durée de ce signal qui détermine la rotation de l'axe donc la position de l'objet fixé dessus.



FIGURE 4.7 – Servo-moteur SG90

Moteurs Brushless

Comme dans toutes les plates-formes aériennes, les moteurs sont les générateurs de force de la portance qui assure le vol. Généralement, le choix est orienté vers les moteurs brushless qui ont l'avantage d'être petits, légers et puissants. Le seul inconvénient de ce type de moteur, outre le prix, est sa mise en oeuvre. En effet, ils doivent obligatoirement être associés à des régulateurs particuliers, appelés contrôleurs, qui obligent une temporisation bien précise.



FIGURE 4.8 – Moteur Brushless

4.2 Réalisation de la partie logicielle

4.2.1 Environnement et outils de développement

L'application de contrôle de planeur avec ses 4 compartiments(Map visualisation , 3D model , Cockpit instrument et VR simulation) est totalement réalisée dans un seul environnement qui est Matlab avec l'extension simulink, qui nous

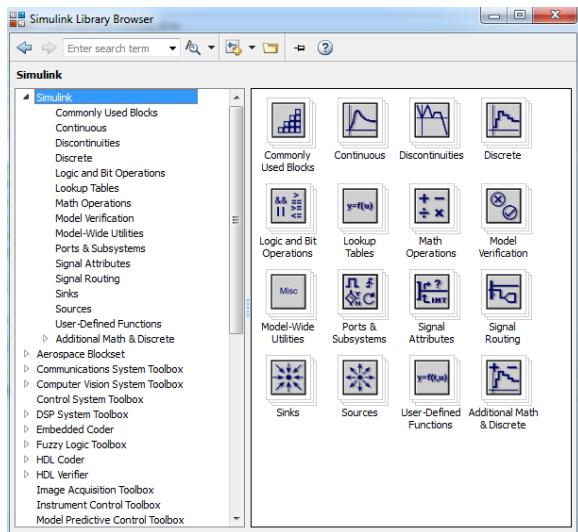
offrent tout les deux les bibliotheque ; aerospace toolbox et 3D Animation , de plus l'environnement Matlab permet d'integrer une fichier HTML afin de manipuler le API Google Maps.

MATLAB

Matlab est un logiciel de calcul matriciel à syntaxe simple, il peut être considéré comme un langage de programmation adapté pour les problèmes scientifiques, grâce à ses fonctions spécialisées. il est considéré aussi comme un interpréteur, car ses instructions sont interprétées et exécutées ligne par ligne.

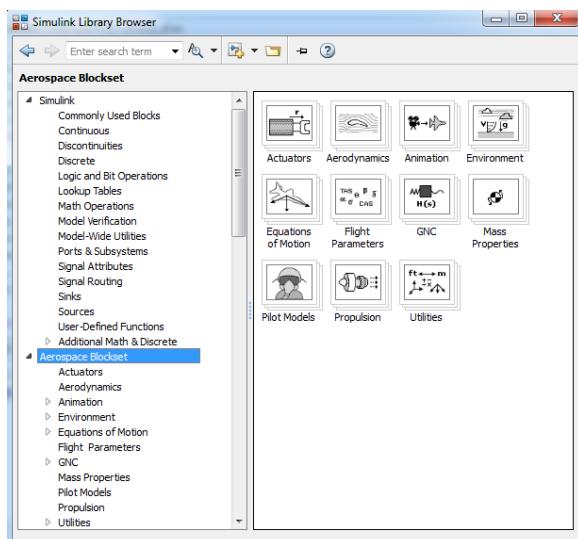
SIMULINK

Simulink est un environnement de diagramme fonctionnel destiné à la simulation multi domaine et à l'approche de conception par modélisation Model-Based Design. Il prend en charge la conception et la simulation au niveau système, la génération automatique de code, ainsi que le test et la vérification en continu des systèmes embarqués.



La bibliothéque Aerospace Toolbox :

Aerospace Toolbox est un produit qui propose des normes de référence, des modèles environnementaux, la possibilité d'importer des coefficients aérodynamiques, ainsi que des fonctionnalités de visualisation en trois dimensions dans Matlab®. Grâce à son interface intégrée avec le simulateur de vol FlightGear, Aerospace Toolbox permet aux ingénieurs de visualiser les données de vol dans un environnement tridimensionnel et de reproduire les anomalies comportementales dans les résultats des vols d'essai. En rationalisant l'analyse des données aérospatiales dans Matlab, cette boîte à outils accélère la conception et le développement de systèmes pour l'aérospatial et la défense.



La bibliothèque 3D Animation

Simulink® Animation 3D™ offre des applications pour relier les modèles Simulink et algorithmes MATLAB® à des objets graphiques 3D. Il vous permet de visualiser et de vérifier le comportement du système dynamique dans un environnement de réalité virtuelle. Les objets sont représentés dans le Virtual Reality Modeling Language (VRML), un langage de modélisation 3D standard. Vous pouvez animer un monde 3D en changeant la position, la rotation, l'échelle et d'autres propriétés de l'objet lors de la simulation de bureau ou en temps réel.

Vous pouvez aussi injecter des signaux de capteurs virtuels et des données d'animation accès 3D dans Simulink ou MATLAB pour le post-traitement.

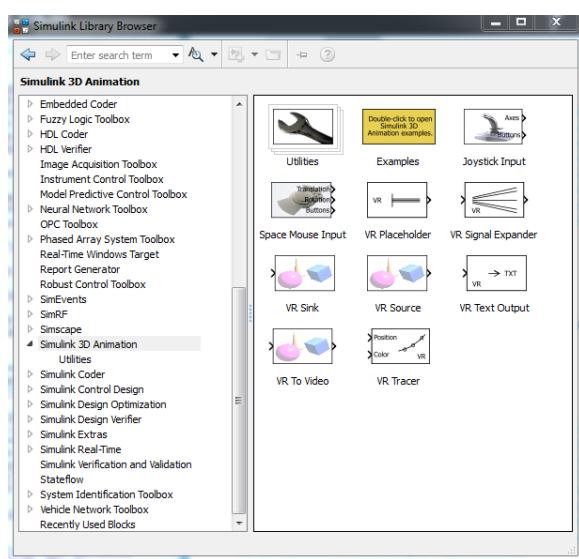


FIGURE 4.9 – Simulink 3D Animation Tools

API Google Maps

Api de Google permettant de géolocaliser des adresses sur une carte à l'aide de sa latitude et de sa longitude. Cet api permet de localiser tout type de données sur une carte (routière, satellite, mixte) à partir de son adresse postale. Cet api s'avère très utile pour proposer aux internautes une vision globale et géographique de données .Les résultats sur la carte apparaissent sous la forme d'un picto (petite icône) cliquable. En cliquant sur cet icône, une fenêtre s'ouvre dans laquelle se trouve toutes les informations concernant la donnée géolocalisée, Il est également possible de calculer un itinéraire depuis un point de départ jusqu'à un point d'arrivée.

4.2.2 Description de l'application

L'application permet principalement à l'utilisateur de saisir l'itinéraire à parcourir par le planeur, envoyer les coordonnées de l'itinéraire au microcontrôleur et visualiser en temps réel toutes les données de vol du planeur. De plus l'opérateur peut utiliser les différentes interfaces pour tester tous les capteurs avant de lancer sa mission. Une autre fonctionnalité est possible via l'application qui offre à l'utilisateur la possibilité tester différents types d'asservissement d'autopilote sur le simulateur avant de choisir celui qui convient le mieux à ses besoins (étape à réaliser avant l'implémentation de l'algorithme sur le microcontrôleur).

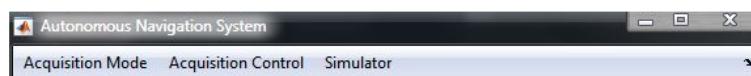


FIGURE 4.10 – Menu de l'application

L'utilisateur a le choix entre deux modes d'acquisition de données :

- L'acquisition à partir du microcontrôleur via une communication série. Dans ce cas, toutes les interfaces de l'application seront mises à jour à partir des valeurs des différents capteurs.
- L'acquisition à partir du simulateur, ainsi toutes les interfaces seront mis à jour à partir des sorties émises par le simulateur. Ensuite l'opérateur peut choisir de commencer l'acquisition des données, la mettre en pause ou la stopper.
- Et enfin l'utilisateur peut choisir de se connecter ou se déconnecter du simulateur.

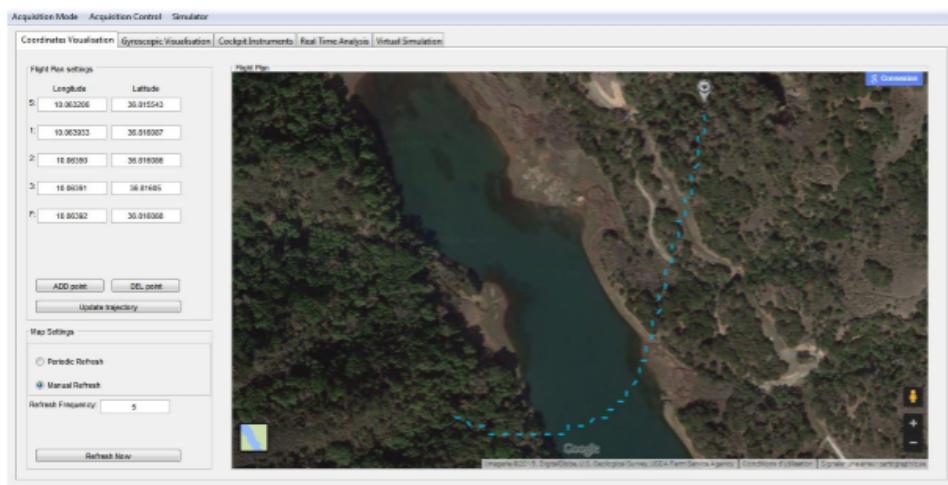


FIGURE 4.11 – Interface : Coordinates visualisation

Cette interface permet de saisir les coordonnées du trajet et suivre en temps réel le trajet parcourus par le planeur.

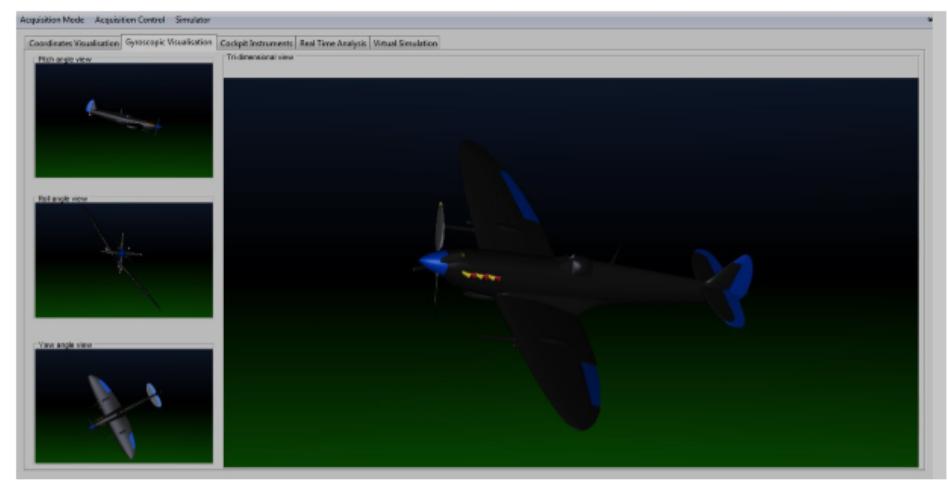


FIGURE 4.12 – Interface : Gyroscopique Visualisation

Cette interface permet de suivre en temps réel les différentes rotations effectuées par le planeur (pitch, roll, yaw).



FIGURE 4.13 – Interface : Cockpit Instruments

Cette interface contient les instruments d'aviation basiques disponibles dans tout avion qui sont notamment (Altimetre, RPM indicator, Artificial Horizon, Heading indicator, Airspeed indicator, Vertical Speed indicator et Turn Coordinator) ainsi que la température et la pression.

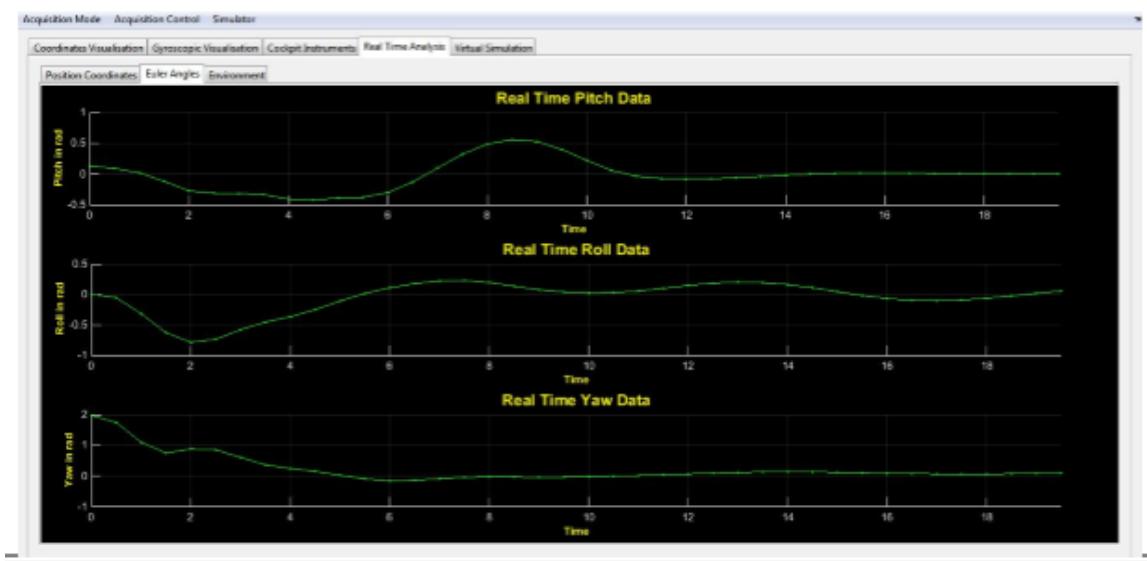


FIGURE 4.14 – Interface : RealTime Analysis

Cette interface illustre l'évolution en temps réel dès le début de la mission des valeurs suivantes (Altitude, Longitude, Latitude, Pitch, Roll, Yaw, Température et Pression) qui serait très utile pour l'évaluation du système d'autopilote.

Fleight Gear Simulator

En conséquence de la demande en ressources matérielles de cette interface, on a opté pour la séparer de l'application et de l'exécuter sur une machine distante via une communication WIFI.



FIGURE 4.15 – Interface : FlighGear Simulator 1



FIGURE 4.16 – Interface : FlighGear Simulator 2

Cette interface Permet de visualiser les sorties du simulateur dans un monde virtuel. Mais qui permet aussi de visualiser le comportement en temps réel du planeur ce qui offre à l'utilisateur une vision optimale pour le contrôle en mode manuel.

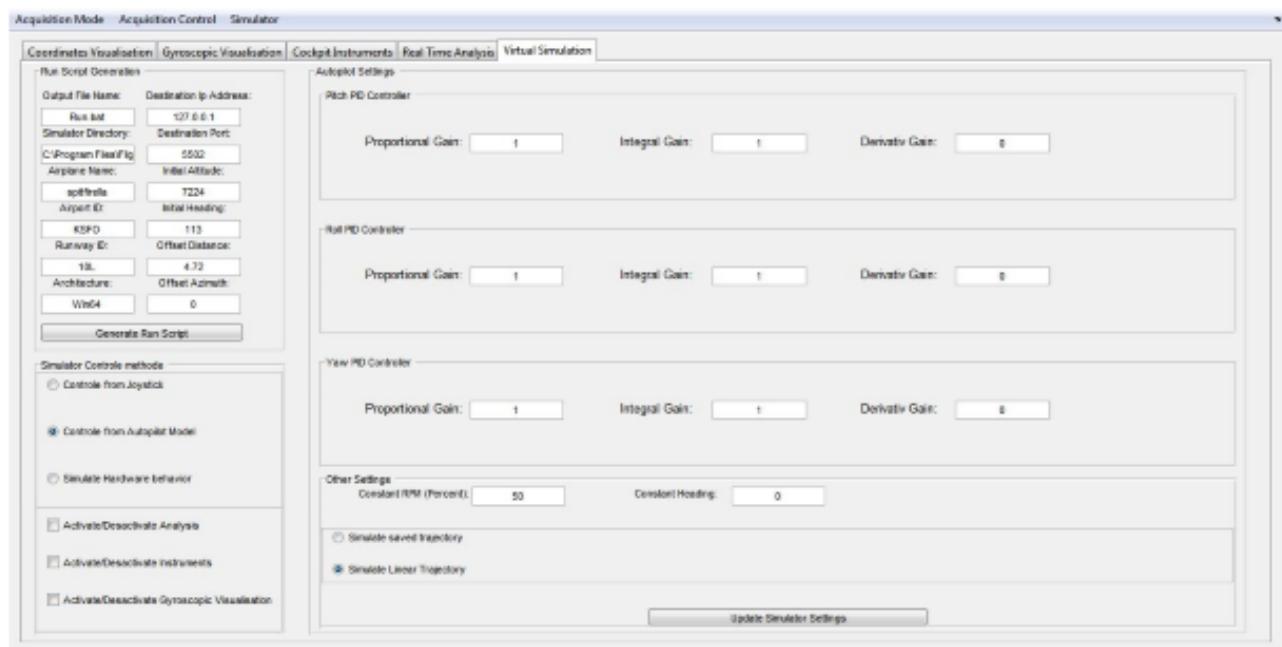


FIGURE 4.17 – Interface : Virtual simulation

Cette interface permet de modifier tout paramètre relatif au simulateur et à l’asservissement de l’autopilote ainsi que l’interfaçage avec le moteur graphique de FlighGear Simulator

4.2.3 Description du simulateur

Le programme de simulation, s’exécute en arrière-plan lorsque l’utilisateur choisit de se connecter au simulateur via l’application de contrôle. Le programme se décompose en un modèle de calcul de nouvelle position et un modèle d’autopilote. Le simulateur prend comme entrée la position initiale, les coordonnées du trajet et les paramètres de l’autopilote (tous fournis par l’utilisateur via les interfaces de l’application). Ensuite le modèle de calcul de nouvelle position effectue une simulation des forces et moments exercées sur le modèle géométrique de l’avion (la gravité, la force de traction du moteur, la résultante aérodynamique des ailerons du rudder et des élévateurs, la résultante aérodynamique du corps de l’avion, les forces exercées par l’environnement) Après il calcule leurs somme et les applique sur le modèle géométrique de l’avion. Au finale il renvoie la nouvelle position de l’avion (altitude, longitude, latitude, pitch ,roll, yaw) ces données seront passés aux interfaces de l’application de contrôle et exploiter pour le calcul de la nouvelle position ainsi que par le modèle de l’autopilote.

CONCLUSION GÉNÈRALE

Dans ce rapport, nous avons présenté les différentes étapes menant à la réalisation de ce système. Notre travail a débuté par une phase théorique, à travers laquelle nous avons présenté le cadre général de notre travail. Dans le deuxième chapitre nous avons spécifié les besoins qui nous ont permis de préciser les différentes fonctionnalités du système. Ensuite, le troisième chapitre nous avons consacré pour exposer la conception de notre projet à travers une explication détaillée des différentes parties du système. En dernier lieu, nous avons conclu par la phase de réalisation qui décrit l'environnement matériel et logiciel utilisé.

Dans ce projet nous nous sommes intéressés à la conception et à réalisation d'un système embarqué qui est composé principalement de quatre parties : le planeur, le simulateur , l'application de contrôle de vol et le module de pilotage manuelle.

Le planeur représente en fait la partie la plus importante de notre système. Il s'agit de l'aéronef destiné aux missions d'exploration des endroits inaccessibles. Les trois autres parties tournent effectivement autour de lui dans le but ultime d'effectuer ce type de missions.

La réalisation de ce projet fût assez délicate à cause des contraintes temporelles et la disponibilité du matériel. En effet, pour remédier au problème de la stabilité du planeur, plusieurs fonctionnalités ont subi quelques modifications, parfois majeures, au fur et à mesure de l'avancement du projet. Vu que les planeurs sont des systèmes critiques qui nécessitent des corrections assez rapides pour effectuer des vols stables, une forte synchronisation entre les différentes tâches avec des périodes bien définies s'impose.

Le système réalisé au cours de ce projet présente tout de même certaines insuffisances en raison de la durée limitée du projet et auxquelles nous pouvons remédier. Ces insuffisances constituent les futures perspectives de notre projet. Nous citons par exemple l'intégration d'un module de détection des obstacles par le planeur. Nous pouvons ajouter d'autres modifications à notre système pour effectuer la livraison des petits charges.

NETOGRAPHIE

- 1 https ://www.mathworks.com
- 2 http ://www.mathworks.com/products/simulink/
- 3 http ://openclassrooms.com/courses/google-maps
- 4 https ://ladyada.net/make/xbee
- 5 http ://www.beagleboard.org/
- 6 http ://www.planeur-colmar.net/
- 7 http ://www.futura-sciences.com/magazines
- 8 http ://www.ferdinandpiette.com/
- 9 http ://www.splitfire.fr
- 10 http ://w3.gel.ulaval.ca/intromatlab
- 11 http ://www.mathworks.com/help/sl3d/vrml.
- 12 http ://www.digi.com/xbee
- 13 https ://www.turbines-rc.com/fr/servos/
- 14 http ://www.sparkfun.com/products/12857
- 15 https ://www.lextronic.fr/P28675-platine-beaglebone-black-rev-c

BIBLIOGRAPHIE

- 1 M.Attene,B.Falcidieno, M.Spagnolo - Hierarchical mesh segmentation based on fitting primitives, The visual computer : international Journal Of Computer Graphics, 2006
- 2 Li APPLIED NONLINEAR CONTROL de Jean-Jacques Slotine