COHEN--URRUTIA Florian **|** BACONNAIS Olivier **|** BADANIN Roman **|** BLANC Indie

**TP1 : Appontage automatique d’un avion de chasse**

Introduction

Un appontage est l’action d’atterir sur une plateforme. Pour notre sujet, nous nous intéressons aux avions de chasse qui attérissent sur un porte avion car c’est la catégorie d’avion qui enregistre le plus d’échac pour cette manœuvre. C’est une action délicate en raison des conditions dans lesquelles elle est exécutée : visibilité pour le pilote, mouvement de la masse d’air autour du porte avion, mouvement du porte avion fortement dépendant de l’amplitude et de la houle, …

On considère que l’orientation d’un avion en vol est déterminée par 3 axes fixes :

1. L’axe longitudinale, dont la rotation correspond au roulis. On le définit par
2. L’axe vertical, pour mesurer le lacet. On le définit par
3. L’axe latéral, pour mesure le tangage. On le définit par

La manœuvre d’appontage se déroule en 3 phases :

1. Le pilote place l’appareil en position d’approche. Il manœuvre pour atteindre 3 objectifs :
2. Aligner l’axe longitudinale de l’avion à l’axe d’appontage
3. Ramener les paramètres de navigation à une configuration initiale
4. Enclencher le pilote d’appontage automatique
5. Lancer l’algorithme à bord du porte avion qui va générer en temps réel la trajectoire d’appontage.

Il transmet aussi les paramètres suivants :

* + 1. V qui est la vitesse de l’avion suivant l’axe longitudinale de l’avion
    2. qui est l’angle d’inclinaison de l’axe longitudinale de l’avion vers le porte avion

1. Fin de manœuvre d’appontage. Les paramètres de l’avion sont ramenés à la configuration suivante :

* = 0
* = 0
* = 0
* V = 0

Cela signifit que l’avion apponte et finit par s’immobiliser sur la piste le long de l’axe d’appontage.

Le but de ce projet est de développer une application qui permet d’automatiser la manœuvre d’appontage d’un avion de chasse sur un porte avion.

Question I

Nous avons recours à un développement basé sur une méthode formelle telle que la méthode B car les méthodes formelles permettent d’obtenir une très forte assurance de l’absence de bug dans les logiciels, c’est pourquoi elles sont utilisées pour les logiciels critiques. L’application pour automatiser la manœuvre d’appontage d’un avion de chasse sur un porte avion est une application critique qui requière 0 bug car une simple erreur pourrait avoir des conséquences catastrophiques. Elles permettent aussi de donner une spécification détaillée du système à développer, qui nous permettera de vérifier si la réalisation final est conforme à ce qui était demandé au départ.

Question II

1. **Analyse du système d’appontage**

Le système est organisé autour des systèmes suivants :

* Système Pilote
  + Représente le système de contrôle automatique de l’appontage
  + Etat :
    - Contexte
    - Appontage
  + Fonctions :
* Sous-système Contexte
  + Définit le contexte de fonctionnement global du système
  + Etat :
    - Axe longitudinal : roulis (float)
    - Axe vertical : lacet (float)
    - Axe latéral : tangage (float)
    - Vitesse de l’avion : vitesse (float)
    - Enclenchement du pilote automatique : pilote (booléen)
  + Fonctions :
    - **Initialisation(roulis, lacet, tangage, pilote)** : initialisation du système :

pilote = TRUE, roulis = 0 , lacet = 0, tangage = 0, vitesse = 0

* Sous-système Appontage
  + Système opérationnel chargé de coordonner le maintien de l’avion dans le plan d’approche et la trajectoire de descente de l’avion dans ce plan
  + Etat :
    - PlanApproche
    - Trajectoire
  + Fonctions :
    - **Appontage(PlanApproche, Trajectoire)** : permet de coordonner la trajectoire avec le plan d’approche pour que la trajectoire se rapproche du plan d’approche (trajectoire sera égale au plan d’approche quand l’avion sera posé.
* Sous-système PlanApproche
  + Garantit le maintien de l’avion dans le plan d’approche
  + Etat :
    - RegulateurPlan
    - CapteurPlan
  + Fonctions :
    - **Maintien(RegulateurPlan, CapteurPlan)** :
* Sous-sytème CapteurPlan
  + Récupère les données réelles fournies par les capteurs du roulis et du lacet de l’avion
  + Etat :
    - CapteurLacet
    - CapteurRoulis
  + Fonctions :
* Sous-système CapteurRoulis
  + Mesure le roulis de l’avion
  + Etat :
    - Axe longitudinal : roulis (float)
  + Fonctions :
* Sous-sytème CapteurLacet
  + Mesure le lacet de l’avion
  + Etat :
    - Axe vertical : lacet (float)
  + Fonctions :
* Sous-sytème RegulateurPlan
  + Corrige en temps réel les paramètres (lacet-roulis) de sorte à maintenir l’avion dans le plan d’approche
  + Etat :
    - RegulateurLacet
    - RegulateurRoulis
  + Fonctions :
* Sous-système RegulateurRoulis
  + Corrige le paramètre roulis de sorte que l’avion soit maintenu dans le plan d’approche
  + Etat :
    - Axe longitudinal : roulis (float)
  + Fonctions :
* Sous-sytème RegulateurLacet
  + Corrige le paramètre lacet de sorte que l’avion soit maintenu dans le plan d’approche
  + Etat :
    - Axe vertical : lacet (float)
  + Fonctions :
* Sous-système Trajectoire
  + Contrôle la trajectoire de descente de l’avion maintenu dans le plan d’approche
  + Etat :
    - Aiguilleur
    - CapteurTrajectoire
    - RegulateurTrajectoire
  + Fonctions :
* Sous-système Aiguilleur
  + Reçoit les données réelles sur les paramètres vitesse et tangage de l’avion et retourne les données pour corriger en temps réel la trajectoire d’appontage
  + Etat :
    - Vitesse de l’avion : vitesse (float)
    - Axe latéral : tangage (float)
  + Fonctions :
* Sous-système BASIC\_IO
  + Gère les entrées/sorties du simulateur
  + Etat :
    - Entrées du simulateur : simulateurE
    - Sorties du simulateur : simulateurS
  + Fonctions :
* Sous-système CapteurTrajectoire
  + Récupère les données réelles sur la vitesse et le tangage de l’avion
  + Etat :
    - CapteurVitesse
    - CapteurTangage
  + Fonctions :
* Sous-système CapteurVitesse
  + Mesure la vitesse réelle de l’avion
  + Etat :
    - Vitesse de l’avion : vitesse (float)
  + Fonctions :
* Sous-système CapteurTangage
  + Mesure l’angle de tangage réel de l’avion
  + Etat :
    - Axe latéral : tangage (float)
  + Fonctions :
* Sous-système RegulateurTrajectoire
  + Commande les régulateurs de vitese et de tangage afin de corriger la trajectoire de l’avion
  + Etat :
    - RegulateurVitesse
    - RegulateurTangage
  + Fonctions :
* Sous-système RegulateurVitesse
  + Corrige la vitesse de l’avion en appliquant la consigne de l’aiguilleur automatique
  + Etat :
    - Vitesse de l’avion : vitesse (float)
  + Fonctions :
* Sous-sytème RegulateurTangage
  + Corrige le tangage de l’avion en appliquant la consigne de l’aiguilleur automatique
  + Etat :
    - Axe latéral : tangage (float)
  + Fonctions :

1. **Modélisation en B du système**
2. **Validation du modèle précédent**
3. **Génération du code source C++ du logiciel qui contrôle l’appontage**