PLAN DE VALIDATION POUR UNE INTERFACE DE CALIBRATION POUR IMU DE DRONE

Authors :

- M. Abrassart Alinoé
- M. Gervais Florent
- M. Saas Guillaume
- M. Naulais Christophe

Date :

Release :

1. Définition du système considéré

L'unité drone de l'ENAC développe un système de pilote automatique pour micro-drone baptisé *Paparazzi*. Créé en 2003, ce système gratuit et libre permet de faire naviguer de façon autonome, à l'aide d'un GPS, un ou plusieurs micro-drones à voilures fixes ou rotatives contrôlés par une station au sol. Il comprend l'ensemble des logiciels permettant aux drones de naviguer ou d'être contrôlés et les méthodes de fabrication des composants matériels nécessaires à l'assemblage d'un drone complet. La clé de ce système repose sur la combinaison d'une centrale inertielle, du GPS et d'autres éléments qui nous permet d'obtenir l'orientation et la position du micro-drone. Aujourd'hui, *Paparazzi* est régulièrement utilisé par de nombreux utilisateurs expérimentés. Il tend à s'élargir de plus en plus vers le grand public et nécessite donc de nouveaux outils pour permettre aux novices de réaliser facilement les tâches nécessaires à la création d'un drone. C'est dans ce contexte que notre projet s'installe.
Une étape importante pour la préparation d'un nouvel engin est alors la calibration de sa centrale inertielle. Notre projet intervient dans la réalisation de cette étape. La méthode actuelle consiste à enregistrer des valeurs des capteurs positionnés particulièrement puis d'appeler un script de calibration sur ces données. Cette méthode repose sur l'utilisation du bus Ivy qui est un bus logiciel permettant simplement d'envoyer des messages textes entre applications distantes ou non. Pour la calibration du micro-drone, la récolte des données se fait en ouvrant une navigation et en enregistrant les données via un fichier généré par serveur (client du bus Ivy). Elle manque clairement d'intuitivité ce qui la rend complexe pour des utilisateurs novices. De plus même les utilisateurs expérimentés ne sont pas satisfait de cette méthode à cause du manque de retour sur la qualité des données acquises par les capteurs. Notre projet consiste à guider l'utilisateur pendant la méthode de calibration de manière claire et précise pour faciliter la tâche à l'utilisateur et aussi donner un retour des données acquises par les capteurs pour améliorer la précision de la calibration.

2. Définition du matériel d'entrée

Trois matériels d'entrées pour valider ce système peuvent être identifiés :

1. Le dossier des besoins
2. Le dossier de conception
3. Le plan de projet

Le dossier des besoins regroupe tous les besoins formés par les parties prenantes du projet. Il va jouer un rôle essentiel dans la validation puisque le seul moyen de valider le système est de confirmer qu'il apporte une réponse à ces besoins.

Le dossier de conception détaille le fonctionnement du système et les différents composants du logiciel. Ceci est nécessaire pour identifier les composants à tester, les entrées et les sorties des tests ainsi que les observables.

Le plan de projet est essentiel pour la gestion des ressources pour la V&V du projet. Le plan de test sera conçu selon les couts en temps des tests et le temps disponible pour la réalisation du projet.

Documents décrivant le système en cours de validation :

- JavaDoc du logiciel.
- Manuel d'utilisation pour l'utilisateur sous forme de page wiki.
- Code Java
- Wiki de l'application

3. Revue critique du matériel d'entrée

Le dossier des besoins fournit des exigences que le logiciel doit satisfaire. Le dossier de conception nous permet d'identifier les classes dont est constitué le logiciel de calibration. Les deux nous permettent de distinguer quelle classe sera pertinente à tester pour chaque exigences. La première étape consiste à détailler les exigences formulées par le dossier des besoins.
Le dossier des besoins ne fournit pas nécessairement de quantification par rapport à chaque besoin, cette revue critique vise:

B1.1 Le système permet la calibration des Accéléromètres
- les paramètres de calibration obtenus divergent de moins de 5% avec le système existant.

B1.2 Le système permet la calibration des Magnétomètres
- les paramètres de calibration obtenue divergent de moins de 5% avec le système existant.

B1.3 Les gyromètre ne sont pas faits
B1.4 L'interface doit proposer la calibration de plusieurs types de senseur, Accéléromètres et Magnétomètres

B1.5 Le logiciel doit afficher les paramètres de calibration obtenus, en format XML FORMAT exemple

B1.6 Les données obtenues pendant la prise de mesure doivent être sauvegardées dans un fichier log. data

B2.1 L'interface doit offrir des indications pour aider l'utilisateur à orienter le drone pour une couverture optimal des axes

B2.2 L'interface graphique doit offrir un indice de précision de la calibration à tout moment pendant la prise de mesure.

B2.3 L'utilisateur doit avoir la possibilité de stopper la calibration et de la reprendre en temps voulu.

B2.4 L'utilisateur peut générer des paramètres de calibration à tout moment pendant la prise de mesures.

B2.5 L'utilisateur peut choisir entre une calibration des accéléromètres ou des magnétomètres.

B2.6

- L'interface doit afficher le drone connecté
- Le système peut lire les données raw_data du drone sur le bus Ivy
- Le système doit pouvoir filtrer les données entrantes

B2.7 Les résultats de la calibration doivent pouvoir être copier/collable depuis notre logiciel vers l'airframe

B3.2 L'interface doit afficher les données de l'IMU en <5 secondes, pendant la prise de mesures
(Pour permettre à l'utilisateur de cibler des zones bien précises pour la calibration)

B2.8
- L'interface graphique doit permettre de guider correctement l'utilisateur en limitant la phase de recherche de fonction à 10 secondes en moyenne
- Lorsque le mode RAW_DATA utile à la calibration est actif un rendu visuel doit prévenir l'utilisateur que le logiciel enregistre des données.
- L'interface doit afficher des recommandations de positionnement du drone à l'utilisateur pour faciliter la calibration accélérométrique.
- Une barre de progression doit permettre à l'utilisateur de savoir quand est-ce que celui-ci doit changer de position l'IMU lors de la calibration des accéléromètres.
- L'utilisateur peut changer le mode de transmission du drone à partir de l'interface

B3.1
- Le temps d'apprentissage pour un utilisateur novice est <5 secondes

B4.1 Le système fonctionne de manière nominale sous un environnement et une charge de travail normale et une configuration de travail classique.
- L'interface n'a pas de latence supérieure à une seconde

B4.3 La calibration après la prise de mesures doit s'effectuer en moins de 10 secondes.
B4.4 L'interface doit afficher à tout moment l'état du système.

4. Stratégie

4.1 Analyse et classification

4.1.1 Classification des caractéristiques

Une définition des différents niveaux de criticité pour chacune des exigences est nécessaire au bon déroulement des opérations de validation et de vérification.

Niveau de criticité :

Niveau 1: Le système doit impérativement respecter ces exigences, si celle-ci ne sont pas respectées le système ne sera pas utilisable.
Niveau 2: Le système doit suivre ces exigences pour une meilleure utilisation du système.
Niveau 3: Le système peut suivre ces exigences pour une meilleure utilisation du système.

Il est impératif que l'exigence avec un niveau de criticité 1 soit accomplie pour le succès du projet. Les exigences de niveau 2 sont importantes pour les utilisateurs du logiciel. Si ces exigences ne sont pas atteintes, les gens utiliseront le logiciel de calibration existant.
Les exigences de niveau 3 ne sont pas critique mais simplifient l'usage du logiciel.

4.1.2 Classification des components du système

Le logiciel peut être décomposé en plusieurs parties, qui chacunes ont un rôle particulier au sein du système.
Les quatre parties sont :

A. **Calibration**

La classe Calibration est la partie du logiciel qui gère la calibration. Cette classe obtient une liste de vecteurs filtrés en entré. Une calibration est effectuée et les six paramètres de calibration, ainsi que la qualité de la calibration sont calculés. On doit pouvoir vérifier que le système garantit que la qualité des résultats sera la même qu'avec le système précédent.

B. **Filtre/Data/IMU/Vecteur**

Les classes Filtre, Data, IMU et Vecteur sont traitées ensemble à cause de l'interdépendance forte de ces classes. Elles sont responsables de la récupération des données sur le bus ivy, la transformation des données en vecteurs, le filtrage et le stockage de ces vecteurs. Ce composant a les données du bus Ivy en entré, un fichier log et une liste de vecteur en sortie. La liste des vecteurs servira d'entrée au composant Ellipsoide et Acelerometer view et le log au composant Calibration
La première stratégie de tests visera à faire une revue de chaque classe indépendante. Ceci est nécessaire parce que ces classes ne peuvent pas être testées indépendamment. La deuxième stratégie de tests consiste à tester les entrées et sortie du groupe des classes.

C. IHM

Deux cas peuvent être différenciés dans l'IHM. Il y a les tests des fonctionnalités réelles de l'IHM, les différentes options disponibles pour l'utilisateur et il y a les tests utilisateurs qui vont valider les performances globales du logiciel.

Le contrôle IHM est l'afficheur qui va permettre un feedback à l'utilisateur. Sans ces classes l'utilisateur n'a aucun contrôle sur l'application. Cette classe permet aussi de modifier les paramètres du drone connecté au logiciel.

D. **Ellipsoide et Accelerometer view**

L'ellipsoïde et l'accéléromètre view qui font le lien entre les données filtrées et l'afficheur. Pour tester l'ellipsoïde il est notamment indispensable d'avoir une vue de celle-ci avec une représentation des points issus de la mesure. Doit répondre aux spécifications attendues de l'affichage exprimé dans le dossier des besoins une fois celles-ci rendues quantifiables.

Type de calibration : classe fondamentale (le système ne peut pas être lancé sans) mais celle-ci ne consistant qu'en un lancement des différentes classes du système, une simple revue et un test de lancement de chaque classe permet de s'assurer de leur création.

4.1.3 Matrice de stratégie

La matrice stratégique relie les composants avec les caractéristiques des exigences. Cela permet d'avoir un aperçu du composant a tester pour valider des exigences selon leur ordre de criticité.

Groupe	Exigence	Ordre de criticité
A, B, C,	Le système permet la calibration des accéléromètres	1
A, B, C,	Le système permet la calibration du magnétomètre	1
A	Les paramètres de calibration obtenus divergent de moins de 5% avec le système existant.	1
C	L'interface doit proposer la calibration de plusieurs types de senseur, accéléromètre et magnétomètre	2
C	Le logiciel doit afficher les paramètres de calibration obtenu, en format XML	2
B	Les données obtenues pendant la prise de mesure doivent être sauvegardé dans un fichier log.	3
D	L'interface doit offrir des indications pour aider l'utilisateur à orienter le drone pour une couverture optimal des axes	1
C	L'interface graphique doit offrir un indice de précision de la calibration à tout moment pendant la prise de mesure.	2
C	L'utilisateur doit avoir la possibilité de stopper la calibration et de la reprendre en temps voulu.	3
A, C	L'utilisateur peut générer des paramètres de calibration à tout moment pendant la prise de mesure.	3
C	L'utilisateur peut choisir entre une calibration des accéléromètres ou des magnétomètres.	
B, C	L'interface doit afficher le drone connecté	3
B	Le système peut lire les données raw_data du drone sur le bus Ivy	1
B	Le système doit pouvoir filtrer les données entrantes	2
D	L'interface doit afficher les données de l'IMU en <5 secondes, pendant la prise de mesures	1
C	L'interface graphique doit permettre de guider correctement l'utilisateur en limitant les phases de recherche de fonction à 10 secondes en moyenne	2
C	Lorsque le mode RAW_DATA utile à la calibration est actif un rendu visuel doit prévenir l'utilisateur que le logiciel enregistre des données.	3
D	L'interface doit afficher des recommandations de positionnement du drone à l'utilisateur pour faciliter la calibration accélérométrique.	2
C	Une barre de progression doit permettre à l'utilisateur de savoir quand est-ce que celui-ci doit changer de position l'IMU lors de la calibration des accéléromètres.	1
C	L'utilisateur peut changer le mode de transmission du drone à partir de l'interface	3
C	Le temps d'apprentissage pour un utilisateur novice est <5 secondes	3
C	Le système fonctionne de manière nominale sous un environnement et une charge de travail normale et une configuration de travail classique.	3

D	L'interface n'a pas de latence supérieure à une seconde	2
A	La calibration après la prise de mesures doit s'effectuer en moins de 10 secondes.	2
C	L'interface doit afficher à tout moment l'état du système.	3

4.2 Identification des moyens V&V et objectifs

L'approche globale du processus de V&V s'effectuera de façon bottom up, Avec pour commencer la creation et verification des classes IMU/Data/Vecteur/Filtre. Ces classes serviront de brique de base pour le reste. Apres la classe IHM, ellipsoïde et Accelerometer View seront créées et rattachées aux briques de base. La verification de ces classes se fera avec les données generées par IMU/Data/Vecteur/Filtre. La classe calibration sera d'abord verifiée séparement, en utilisant des logs, ensuite en la racordant au classes Data/Vecteur/Filtre/IMU.
En fin de phase de verification la classe de Calibration sera aussi rattachée a IHM pour verifier leur interoperabilité.

Strategie de Verification
L'approche adoptée pour cette partie consiste tout d'abord a des revues du code et de la javadoc associée. Ensuite, des drivers et des classes tests vont être crées pour éxécuter des tests white box sur les differents composants. La verification se fera comme expliqué ci-dessus de facon bottom up, brique par brique au depart pour assembler des classes déjà fonctionelles sur elle-même.
Malgré l'absence de plan de V&V abouti au début du projet plusieurs tests ont d'ores et déjà été validés pour vérifier l'implémentation de certaines classes.
Les outils mis en places pour réaliser ces tests sont les suivants :

- sender qui permet d'émettre les messages contenus dans un fichier
- testIMU qui agit comme un driver de StartUp en déclenchant l'instanciation de plusieurs classes
- EmulData qui agit comme driver de DATA
- AleatGen qui émule l'IMU
- GUIHelper, driver de Shell

Des classes spécifique de test ont aussi été créées telles que :

- Test qui est la version parallelisée de EmulData
- TestSender qui permet de tester le driver Sender
- TestShell qui permet de lancer Shell qui est la fenêtre de l'application

Strategie de Validation
La validation du système se fera en fonction des besoins exprimés dans le dossier des besoins qui sont detaillés dans la revue critique du materiel d'entrée. Les tests de Validation seront des tests Black Box effectués par l'utilisateur sur le prototype final.
La validation s'effectuera principalement avec des tests d'utilisateurs pour une couverture des conditions. L'objectif étant de valider les besoins des différentes parties prenantes. La validation est decoupée en trois parties :

Validation des parametres de calibration

Validation des affichage en temps reel

Validation des fonction IHM

4.3 Phasage

Quatre phases de V&V peuvent être distinguées :

1. Phase de vérification des briques de base ainsi que de la chaine IMU-Data-Filtre
2. Phase d'intégration de la brique de base dans un contexte d'affichage
3. Tests avec input utilisateurs pour tester l'IHM et l'intégration du niveau précédent dans cette IHM. Test de connexion IMU-drone. Tests de Calibration.
4. Test du prototype final

Phase 1 et 2 sans input utilisateurs avec rejeux. Phase 3 avec input utilisateurs.

4.4 Délégation

Absence de délégation, sauf pour la phase 4 les tâches de V&V sont réalisées en interne. En phase 4, un utilisateur externe réalisera les tests en conditions réelles au sein du laboratoire de drones de l'enac.

5. Outillage

5.1 Plate-forme

Les plateformes compatibles avec ce logiciel sont les plateformes Linux dérivées de Debian avec paparazzi installées et une machine virtuelle Java SE 6, et Python.

5.2 Outils

Eclipse
Eclipse est in IDE, Integrated Development Environment. L'outil principal pour les tests de phase 1 et 2.

Paparazzi
Pour la phase 3 et 4 le Framework Paparazzi notamment le client Rejeu et la GCS sont importants. Ces outils permettent la création d'un moyen de rejouer les simulations d'une IMU connectée.

Drone
Un autre outil pour les phases 3 et 4 est une véritable IMU connectée directement au logiciel qui permettra de tester le bon fonctionnement de l'IHM.

5.3 Données

Fichier log
Les fichiers logs sont des fichiers au format .data qui contiennent des données utilisées pour la calibration. Ces données sont formatées comme suit:

Indice de temps	Id du drones	Types de données	coordonees x,y,z
12.345	2	IMU_ACC_RAW	34 56 78

Ce fichier log est lu par le script python calibrate.py qui permet de fournir les données nécessaires à la calibration. Nottament la sensibilité, le biais ainsi que la précision des mesures.

Mesures en temps réel
Ce sont des données generées en temps reel, en établisant une connexion avec un drone sur paparazzi. Cela entraine un flux de données sur le bus Ivy tous dans le meme format que le fichier log.

6. Organisation

6.1 Taches de validation

Revue des classes et de leur javadoc
Tests black box des composants du logiciel
Tests black box de groupes de composants
Tests utilisateurs sur le prototype

6.2 Planning

Phase 1 – pendant la conception des classes IMU, Data, Filtre, Vecteur
Phase 2 – en fin de conception des classes IMU, Data, Filtre, Vecteur. Pendant la conception de la classe IHM.
Phase 3 – pedant la conception de la classe IHM, Calibration.
Phase 4 en fin de conception de la classe IHM et calibration.

6.4 Risques

Numero	Identification du risque	Classe de risques
1	Retard dans la livraison des fichiers de log pour les tests de phases 3 et 4.	
2	Indisponibilité des drones pour les tests de phases 3 et 4	
4	Retard dans la livraison du prototype final retardant la dernière campagne de test	
5	Perte des fichiers de log	
6	Modification des fichiers de log faussant la campagne de test	

Annexes

Les fichiers de tests sont fournis à côté IHM 5 et 6 ainsi que IMU 1, 2, 3 et 4.