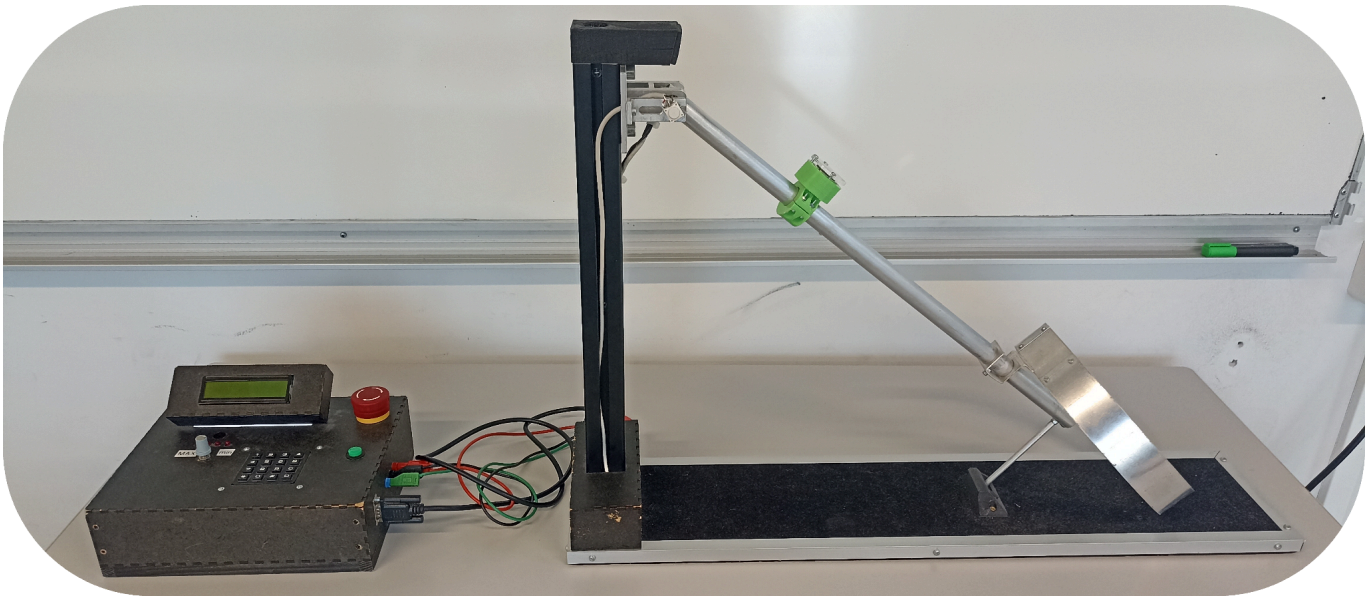
Rapport de mi-projet

Maximilien Kulbicki & Ronan Le Corronc

Introduction

Le projet FlyingArm est une maquette composée d'une hélice montée sur un axe pivotant et d'un boîtier de contrôle. Ce projet a été initié par Pr. Ducard en 2013 et repris par des étudiants au fil des années. Nous avons décidé de le continuer afin d'appliquer les connaissances acquises en cours sur la modélisation de système dynamique et le contrôle digital. Le principal axe de travail concerne la programmation du boîtier de commande afin de le rendre utilisable pour les travaux pratiques de la formation ROBO4.



2 - Objectifs

Dans le cadre de nos cours de contrôle digital et modélisation de systèmes dynamiques, nous souhaitons implémenter un contrôleur pour asservir la position du bras. Le boîtier fait office d'IHM. Il est doté d'un pavé numérique, d'un potentiomètre et d'un boutoir en entrée, de 4 leds, un écran et un buzzer en sortie. Le bras est actionné par un BLDC. Il est doté de capteur renvoyant son mouvement angulaire (potentiomètre, gyroscope et accéléromètre) et sa vitesse moteur.

Un des points majeurs de cette reprise consiste en la modernisation du code. En effet, celui-ci a recours à des librairies et packages KEIL obsolètes, pouvant induire des erreurs de compilation sur les machines développant sous une version récente de KEIL. Nous avons donc pris la décision de refactoriser l'intégralité du programme en utilisant les librairies actuelles fournies par STM et KEIL.

3 - Avancement

3.1 - Première approche

Dans un premier temps, nous avons estimé possible la création d'une librairie de fonctions permettant de contrôler / communiquer avec chaque composant du système. Nous avons reconfiguré le microcontrôleur STMF407VGT6 sous STMCubeMx.

Ainsi, on s'est focalisé sur le fonctionnement et la réalisation de PWM pour contrôler le moteur. Nous nous sommes aidés du configurateur STM32CubeMX mais une approche théorique à tout de même été nécessaire pour comprendre le fonctionnement des timer et comment régler ses paramètres et les différentes horloges du STM32. Un timer a par la suite été utilisé pour générer un signal PWM de longueur d'impulsion facilement modulable en modifiant les valeurs d'un registre du timer.

Concernant le clavier, un dictionnaire était nécessaire pour traduire les données sur 4 bits reçues en lettres correspondantes aux touches. Les pins utilisés ont été configurés en interruption pour qu'à chaque nouvel appuie sur le clavier, la lettre correspondante puisse être enregistrée et exploitée à notre guise.

Un autre moyen de communiquer avec l'utilisateur est d'utiliser l'écran LCD de la maquette. L'écran est câblé au STM32 à l'aide de 4 fils de données. Nous avons donc récupéré et adapté des librairies trouvées sur internet pour pouvoir interagir avec. Ainsi, on est en mesure d'effacer l'affichage, changer la place du curseur et afficher des chaînes de caractères ou des entiers.

Afin d'afficher les différents modes de fonctionnement disponibles, un menu déroulant consistant en une structure de données a été mis en place. Chaque champ du menu assimile un gestionnaire d'interruptions différent au bouton poussoir. Ainsi, selon le champ sélectionné, on est en mesure de sélectionner les actions souhaitées. La mise en forme du menu a présenté la contrainte principale en raison du nombre de pixel limité.

Des difficultés ont été rencontrées avec le gyroscope. La documentation de ce composant indique un fonctionnement en communication I2C. Après de nombreux essais et échecs à communiquer avec lui, nous nous sommes intéressés à l'hardware du projet et avons constaté un fonctionnement en UART.

3.2 - Approche adoptée

Les résultats ont été prometteurs mais refaire l'intégralité des fonctions se présentait comme une tâche chronophage. Nous avons choisi de consacrer une grande partie de notre temps à étudier le programme conçu par Pr. Ducard et ses élèves. A partir de ces fichiers nous avons pu modifier certaines de leur fonction pour les adapter aux librairies actuelles. Il a parfois fallu juste changer le nom des commandes utilisées et parfois restructurer complètement l'algorithme, ça a été notamment le cas pour les interruptions.

4 - Travaux à réaliser

Nous avons assimilé les principales fonctionnalités que propose le programme du Pr. Ducard et comptons à présent les réimplémenter afin de modéliser un système complet.

Afin de contrôler le bras au travers de l'interface QT, il nous reste à comprendre la nature des données échangées et programmer les fonctionnalités des modules de cours. Cela consiste en l'asservissement en position du bras en agissant sur le moteur et l'estimation de certains paramètres physiques (inertie ou masse) du bras.

Par ailleurs, le capteur de vitesse du moteur n'est pas pris en charge et constitue une piste de travail supplémentaire.

Finalement, nous étudierons la nécessité d'implémenter le gyroscope et accéléromètre en DMA.