

VAFC REACT

<i>de ACHBAH Fatima-Zahra</i>	FatimaZahra.Achbah@etu.univ-valenciennes.fr
<i>FOURNET Camille</i>	Camille.Fournet@etu.univ-valenciennes.fr
<i>HERVE Aideen</i>	Aideen.Herve@etu.univ-valenciennes.fr
<i>JAYASEELAN Arunjah</i>	Arunjah.Jayaseelan@etu.univ-valenciennes.fr
<i>KAMDEM Barbara</i>	Barbara.Kamdem@etu.univ-valenciennes.fr

Abstract In this report we present the adaptation and improvement of an existing project. The project is a reflex game called VAFC React, and we happen to create a graphic interface and improve the existing code. Besides, we create mechanical support for the Raspberry Pi graphical interface, without forgetting the USB connection between the interface and the central Arduino.

Keywords—React, électronique, code, Arduino, Raspberry, Interface, Mécanique, Technique, connections, Smart, CAO, Python,...

I. INTRODUCTION

Les jeux de réflexes testent notre réaction aux différents types d'excitation.

Jusqu'à présent, il n'y a pas énormément de produits ni de systèmes qui testent les réflexes des joueurs et qui affichent le temps de réaction de ceux-ci. C'est dans ce cadre, que des élèves de la promotion 2014 ont pris l'initiative de réaliser une machine capable d'afficher le temps de réaction moyen du joueur. La machine fût développée en partenariat avec le Football Club de Valenciennes, VAFC. Cette machine permet d'entrainer les gardiens mais également les joueurs de champ. En effet, composée de 15 leds disposées face au joueur, celui-ci a alors pour but d'éteindre le plus rapidement possible les leds s'allumant. Il existe divers programmes suivant l'objectif d'entraînement voulu. En reprenant ce projet, différentes missions nous ont été confiées, que ce soit dans le domaine électronique, informatique ou mécanique. Nous devions améliorer la machine existante.

II. PRESENTATION DU PROJET

A. Contexte du Projet :

Chaque sportif doit mettre à sa disposition une préparation physique constituée d'entraînements pour être au meilleur niveau. La préparation physique est l'ensemble organisé et hiérarchisé des procédures d'entraînement sportif. Ces entraînements visent au développement et à l'utilisation des qualités physiques du sportif et dans notre cas le footballeur.

La préparation physique commence par l'analyse des exigences de la discipline sportive (évaluation et expertise). Cette analyse se fait à l'aide de système de tracking (GPS, Accéléromètre) de hautes technologies couplées avec la vidéo et des analyses statistiques. Il sera alors possible de recueillir des données qui correspondront à l'effort à fournir en compétition (exemple : durée de l'effort, distance parcourue, vitesse, accélération, nombre d'actions intenses,

fréquence cardiaque, dépense énergétique...). À partir de ces données on pourra établir un ordre de priorité des qualités physiques (Endurance, Force, Vitesse, Coordination et Souplesse) à développer en relation avec la discipline. Ce développement passera par un test des qualités physiques de l'athlète ou du joueur, puis de la programmation et la planification de la préparation physique de ces qualités en tenant compte de l'individualisation des charges de travail de chaque athlète ou joueur. Ces tests peuvent être effectués à l'aide de jeu testant le temps de réaction de chaque joueur.

Les méthodes contemporaines d'entraînement physique sont parfois décrites selon six principes physiologiques fondamentaux :

- Individualisation : Le potentiel génétique et les capacités d'adaptation à la charge d'exercices physiques sont différents selon les individus. Le programme d'entraînement doit donc être personnalisé, afin d'éviter les blessures et maximiser la progression de l'amélioration physique. Le programme d'entraînement d'un sportif débutant sera différent de celui d'un amateur déjà entraîné, celui d'un sportif de grande corpulence différent d'un sportif de petite corpulence.
- Spécificité : Le type et l'intensité des exercices d'entraînement sont spécifiques à la discipline sportive, puisque les capacités physiques requises sont différentes selon les sports. Le programme de musculation d'un joueur de tennis sera différent de celui d'un athlète.
- Régularité : La pratique régulière des exercices physiques permet le maintien d'un niveau de performance sportive. L'arrêt de l'entraînement a pour conséquence l'effondrement des aptitudes physiques gagnées par l'entraînement.

- Progressivité : Les exercices physiques reposent sur un principe de contraintes physiques ou « surcharge » (musculaire, cardiovasculaire, articulo-tendineuse) dans le but de forcer une amélioration physique (augmentation du volume musculaire, des capacités cardiaques...). Néanmoins cette augmentation de charge doit être progressive, pour permettre au corps de s'adapter sans blessure (dans le cas contraire on parle de surentraînement).
- Alternance : La qualité de l'entraînement ne réside pas seulement dans l'intensité et le volume des exercices (charge), mais aussi dans leur alternance avec des jours de moindre charge, permettant au corps de récupérer (repos, reconstitution des fibres musculaires...). Un programme d'entraînement intègre ainsi des jours d'entraînement modéré et des jours de récupération.
- Périodicité : Selon ce principe popularisé depuis une vingtaine d'années, le programme d'entraînement d'une saison sportive est planifié sous forme de cycles progressifs pour la spécificité, l'intensité et le volume des exercices (en jouant notamment sur l'effet de surcompensation). Le but est d'atteindre un pic de forme au moment de la compétition.[1]

Le VAFC React étant une machine d'entraînement, il se doit de remplir les 6 conditions d'entraînement efficace décrites ci-dessus.

Batak Pro, Fitlight trainer, Blaze pod, Treax pads, sont des systèmes d'entraînement physiques qui permettent de développer la coordination des mouvements, la réactivité, la position périphérique ainsi que le positionnement dans l'espace. Chaque utilisateur dispose d'un temps donné pour éteindre un maximum de cibles.

B. Principe:

Suivant le mode d'entraînement sélectionné, des leds vertes ou rouges vont s'allumer dans une zone limitée dépendant du programme. Le joueur doit alors, le plus rapidement possible éteindre les leds vertes. Si celui-ci éteint une led rouge, il commet une erreur. En fin de partie le temps moyen de réaction est annoncé, ainsi que le nombre d'erreurs. L'entraîneur peut alors suivre la progression de ses joueurs. Actuellement, la machine fonctionne via un ordinateur. Nous avons pour objectifs de créer une interface homme/machine et de l'intégrer à celle-ci.

C. Vue d'ensemble du système :

La machine est représentée dans la figure suivante:

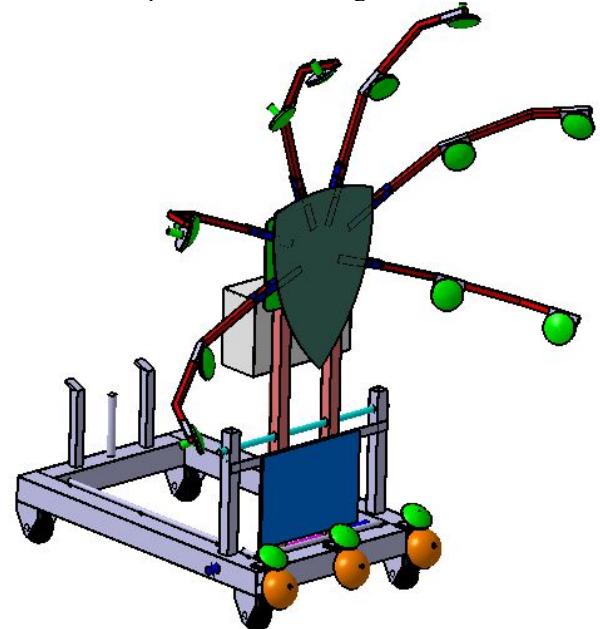


Figure 1: VAFC React

Le principe de fonctionnement de base est résumé à la Fig.2. Nous avons également besoin d'un petit support mécanique qui sera fixé sur le système montré dans la Fig.1

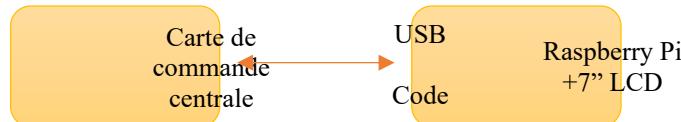


Figure 2: Communication entre la tablette et la carte de commande

La carte Arduino est connectée à Raspberry Pi avec un câble USB. De plus, avec un code implémenté sur Raspberry pi, nous établissons la connexion entre les deux cartes. Cela nous permet d'envoyer des informations à Arduino en utilisant Raspberry Pi au lieu d'un simple ordinateur par la suite l'Arduino contrôle les LED et les données d'entrée du système, puis renvoie les données recueillies à Raspberry avec le câble USB. Le joueur n'a alors accès qu'à ses données grâce à l'interface graphique qui sera programmée sur Raspberry Pi.

III. MANAGEMENT DES TACHES

A. Définition des objectifs :

Le projet 2A permet d'effectuer un travail d'équipe sur plusieurs mois, et de se confronter à une première expérience dans la peau d'un ingénieur. Ce programme a pour objectif de nous confronter à un client et donc à un besoin et d'y répondre de la meilleure façon possible. Dans notre cas, en choisissant le projet VAFC React, nous avons du travailler sur de l'électronique, de la programmation

mais également de la mécanique. L'objectif de la machine est de calculer un temps de réaction moyen d'un joueur de football. Disposant de plusieurs programmes d'entraînement, la machine est complète et demande une amélioration physique et fonctionnelle. Les principaux objectifs de ce projet sont les suivants :

- Réalisation d'une interface Homme/Machine. Actuellement, pour fonctionner un ordinateur doit être relié au système. La création d'une interface sur écran de Raspberry nous a été demandée.
- Réalisation d'un bras mécanique. L'écran de l'interface se situera derrière la machine, or le châssis de la machine ne facilite pas son accès. Un bras mobile doit donc simplifier l'accès à l'écran.
- Réalisation de la liaison Raspberry/Arduino. Les programmes actuels de la machine sont réalisés sur arduino. Il faut donc trouver le moyen d'effectuer la liaison entre les données d'entrées et de sorties du programme arduino avec la Raspberry.
- Optimisation du programme d'entraînement. Cette optimisation a pour but d'analyser les données des joueurs et ainsi d'adapter son programme d'entraînement.

Le principal objectif pour réaliser à bien cette mission est l'organisation. Comment avoir une organisation optimale ? Comment séparer les différentes tâches ?

B. Organisation et acteur du projet :

En définissant une organisation précise, il est possible de mener à bien ce projet. C'est pourquoi lors des premières séances de projet, des réunions *visible planning* étaient mise en place. Ces réunions avaient pour but d'échanger sur le travail en cours, les recherches effectuées, et de partager nos idées. En plus de ces réunions, des comptes rendu sont réalisés pour justifier chaque concept, chaque choix. De ce fait si jamais quelqu'un doit effectuer une modification sur le système, les documents lui sont disponibles. Il peut alors regarder si l'idée ou le concept a été proposé précédemment et pourquoi il a été écarté ou non.

Pour prévenir de tout risque, deux matrices de risque du projet ont été réalisées. Une matrice purement technique, concernant les câblages, les problèmes de matériel; et une seconde regroupant nos tâches et l'aspect management.

Probabilité →	Très faible	Faible	Modéré	Elevé
Impact ↓				
Elevé	Réalisation de l'interface	Retard dans réalisation du bras + support	Connexion Arduino – Raspberry Pi	
Modéré	Absence / Manque de travail		Câblage maquette	Modification du code Arduino
Faible		Création / Conception du bras mécanique + support écran		
Très faible				

Figure 3 : Matrice de défaillance du projet

En ce qui concerne notre projet, celui-ci est mené à bien, malgré les différents problèmes que nous avons eu.

De plus, il a fallu prendre en compte les différents acteurs du projet. En effet, nous avons du dépendre de personne dans l'objectif de réussir le projet. Nous avons eu l'aide de Mr. Lesbros sur la partie mécanique, fabrication et assemblage du bras maintenant l'interface. Il a fallu donc que nous nous rendions disponible lorsque celui-ci l'était. De plus, ayant rencontré des problèmes avec notre maquette soudée, nous avons demandé des renseignements à Mme. Sadaune dans l'espoir de les résoudre. La gestion de projet s'est également fait autour de Mr. Dupont, tuteur du projet. Des réunions étaient organisées pour suivre l'avancement du projet mais également pour valider la conception mécanique, et l'interface homme/machine.

Répartition des responsabilités :

Fatima-Zahra ACHBAH : maquette 15 boutons poussoirs, modification code Arduino

Camille FOURNET : plan de convergence, management, conception mécanique du bras

Aideen HERVE : mise en place de la liaison entre la Raspberry Pi et l'Arduino

Arunjah JAYASEELAN : modification code Arduino

Barbara KAMDEM : réalisation de l'interface homme/machine

C. Plan de convergence et livrables :

Pour atteindre l'objectif final du projet, nous avons établi le plan de convergence ci-dessous pour définir les différentes étapes qui nous mèneront à cet objectif, et à nos livrables.

Le plan de convergence se trouve en annexe 1.

On peut observer que le livrable attendu n'est pas validé. Nous avons livré un produit fonctionnel sur maquette, le test sur la machine n'a pas été effectué c'est pour cela que nous ne pouvions pas valider ce livrable. De plus, certaines tâches restent non validées. Les résultats sont partiellement obtenus. Sous forme de dégradé nous avons pu mener le projet à bien. En effet, certaine tâche ont dues être modifiées pour avancer dans le projet et ainsi produire un bon résultat.

IV. DÉROULEMENTS ET RÉSULTATS

Dans cette section, nous présenterons le travail effectué au cours des quatre derniers mois: Le codage, le CAD, l'électronique.

A. L'adaptation du code Arduino

a. Analyse existant

Avant de débuter le projet, nous avons cherché à étudier tous les fichiers existants qui étaient dans les archives du projet VAFC REACT réalisé par les anciens étudiants de l'ENSIAME. Parmi eux, nous avons trouvé les schémas de câblages et les programmes utilisés par les étudiants afin de réaliser le système.

Etude du câblage :

Le câblage présent dans le VAFC REACT est simple. 2 LEDs (une verte et une rouge) avec 2 résistances (220 Ohm) de pull-up pour 1 bouton poussoir avec une résistance de 150 Ohm de pull-down. Les LEDs sont définies comme des sorties numériques avec les pins de 22 à 51. Nous avons cherché à lier les boutons à des entrées numériques avec les pins de 0 à 14 mais nous sommes rendues comptes que les pins 0 et 1 sont associés au TX0 et RX0. Par conséquent, les résultats obtenus étaient erronés. Ainsi, les boutons sont liés à des entrées analogiques avec les pins de 1 à 15.

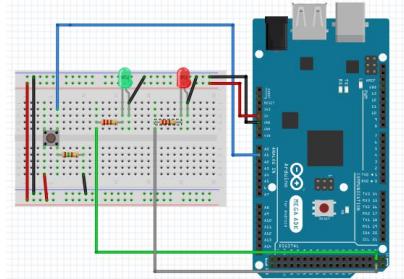


Figure 4 Schéma électrique pour 1 bouton

C'est ce circuit de que nous allons répéter pour les 15 boutons.

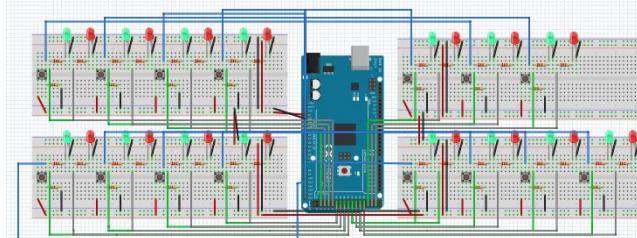


Figure 5 Schéma électrique pour 15 boutons

Etude du programme:

- initialisation

Nous avons dans un premier temps cherché à analyser les lignes du programme Arduino l'essayer avec nos plaques d'essai liée à l'Arduino Mega 2560.

Les premières lignes de codes correspondent à l'initialisation, notamment celles des variables du jeu et les pins sous forme de tableau des LEDs rouges, des LEDs vertes et des boutons poussoirs.

- Setup

Ensuite, on a la partie démarrage de l'Arduino avec le Setup. Le débit d'envoi de données est fixé à 9600 bauds. On y définit également une mise à zéro de toutes les LEDs définies comme des sorties et les boutons sont définis comme des entrées.

- Loop

Puis, nous avons tout le code qui va gérer tout le jeu entre les LEDs et les boutons mais également de temps de réaction dans la partie Loop de l'Arduino.

On commence tout d'abord par remettre à zéro tous les tableaux du programme, c'est-à-dire celui des temps de réaction, des fautes et des bonnes réponses.

Ensuite, le programme attend une réponse de l'utilisateur : un caractère du code ASCII qui va correspondre à un mode de jeu (entraînement, professionnel, évaluation, la durée du jeu, le nombre de boutons...). On va appeler alors la fonction « start » qui va déterminer le mode choisi. Ainsi, le programme sait quelle est la durée de la partie, le nombre de boutons, la place du premier et du dernier bouton...

Une fois tous les paramètres définis pour la partie, on annonce la partie en allumant toutes les LEDs vertes.

On définit ensuite le début de la partie en lançant le temps qui va permettre de déterminer les temps de réactions mais également la durée de la partie.

On arrive maintenant à la boucle « jouer_coup » qui va réaliser le jeu de LEDs et le calcul des bonnes et mauvaises réponses. La boucle « while » va comparer le temps à chaque fois avec la durée totale de la partie afin de respecter le choix du joueur. La fonction « jouer_coup » va récupérer tous les paramètres définis par la fonction « start ». On y fait intervenir deux variables aléatoires : une pour choisir la position de la LED allumée qui est définie grâce aux paramètres de la fonction « start » et une autre variable qui choisit si c'est une LED verte ou rouge qui sera allumée. Les LEDs rouges ne sont allumées que si le mode évaluation est à zéro. On a ensuite le code qui va additionner les erreurs. Si une LED verte est allumée et qu'on appuie sur un mauvais bouton, une erreur est comptée et si on appuie sur le bouton est qu'une LED rouge est allumée, le nombre d'erreurs s'incrémentera aussi.

Lorsque la boucle « while » se termine, on allume les LEDs rouges afin d'annoncer la fin de partie.

Ensuite, nous avons une fonction « afficher » qui va calculer la somme des erreurs et le temps de réaction moyen obtenu pour chacun des boutons. Ensuite, on calcule le temps de réaction moyen qui est la somme des temps de réaction moyen par boutons divisé par le nombre de réponses juste du joueur. On affiche alors sur le moniteur série de l'Arduino le temps de réaction moyen et le nombre de fautes.

b. Modifications à apporter

Pour notre projet, il nous a été demandé de réaliser une interface homme-machine propre à la machine en utilisant le Raspberry Pi.

Par conséquent, il a été nécessaire de réfléchir à la liaison entre les deux microcontrôleurs. La fonction

« start » ne reçoit plus alors un caractère ASCII depuis le moniteur série de l'Arduino mais depuis la liaison série avec le Raspberry PI.

On veut également déterminer les différentes erreurs et différents en fonction des diverses positions des boutons poussoirs. Il est donc impératif de déterminer non plus un temps moyen total mais un temps moyen par bouton. On affichera alors un temps de réaction par bouton.

D'autre part, on souhaiterait comptabiliser les erreurs qui correspondent à l'évènement « LED verte allumée mais aucun bouton appuyé ». En effet, lorsque l'on ne réagit pas à la LED allumée, l'évènement n'est pas considéré.

De plus, on souhaiterait également modifier le code afin de faire travailler des zones plus ciblées, notamment les zones « Haut », « Gauche », « Droit » de la machine.

c. Difficultés rencontrées

Lors de ces études, nous avons rencontrés plusieurs problèmes que ce soit pour le câblage ou l'exécution du programme.

En effet, tout d'abord, il était convenu dans le cahier des charges que nous réalisons une maquette avec les 15 boutons et les 30 LEDs soudées à une plaque de cuivre. Cependant, malgré les divers essais réalisés, nous n'avons pas réussi à réaliser cette maquette. Soit le câblage était erroné soit les soudures étaient mal réalisées. Sans cette maquette il nous a été très difficile de tester le code Arduino. Par faute de temps, nous avons décidé de négliger la maquette soudée et d'utiliser des platines d'essai afin d'avoir une maquette fonctionnelle sans erreur de court-circuit avec le nombre adéquat de LEDs et boutons. Nous avons donc utilisé 3 platines d'essai avec 5 boutons chacun.

Nous avons eu besoin également de debugger le programme Arduino. Nous avons cherché à comprendre ligne par ligne le code écrit par les anciens élèves de l'ENSIAME. Nous avons alors compris que le code « 5 boutons » était un code « 15 boutons » adapté grâce à des tableaux de 5 valeurs au lieu de 15. En effet, la fonction « start » définie plutôt n'était pas adaptée aux 5 boutons. L'exécution correcte des appels de modes ne sont réalisables qu'avec un circuit de 15 boutons. Par exemple, le caractère ASCII « B » fait appel aux 6 LEDs inférieures de la machine. Avec le programme 5 boutons, nous ne recevons pas de temps de réaction moyen par bouton pour les 2 premiers boutons puisque ceux-ci correspondent aux boutons du haut. Ainsi, récupérer un code déjà établi n'a pas été une chose facile. Peu de lignes du code étaient commentées et détaillées.

Lors de la recherche du code qui permettrait de comptabiliser les erreurs lorsqu'aucune action n'est faite par le joueur face à une LED allumée, les tentatives de code n'ont pas abouti.

Pour la recherche du code concernant le travail des zones ciblées, nous avons réussi à atteindre notre première étape. Nous avons écrit un code, qui grâce à un caractère ASCII, permet de jouer soit avec les 4 LEDs d'en haut, soit les 4 LEDs de gauche soit les 4 LEDs de droite. Nous n'avons pas encore réussi à associer le travail des zones

ciblées avec le jeu de LEDs entier. En effet, cet exercice ne permet que de focaliser son entraînement sur une zone précise. Par faute de temps, nous n'avons pas réussi à obtenir le code qui permettrait d'allumer aléatoirement toutes les LEDs du jeu mais aussi de façon plus fréquente celle de la zone étudiée. Par conséquent, ces lignes de codes ne font pas partie du code final.

d. Résumé du programme final

Nous avons repris entièrement le code que nous possédions mais avons fait les modifications nécessaires afin de récupérer le caractère ASCII venant du Raspberry PI et l'envoi des temps de réaction moyen par bouton, toujours au Raspberry.

La première modification a lieu dans la fonction « start », nous avons ajouté quelques lignes de code qui vont permettre de convertir la donnée reçue par le port série de l'Arduino.

Dans la fonction « afficher », nous n'affichons plus un temps de réaction moyen total mais un temps de réaction moyen par bouton et le nombre de fautes. Ainsi, il est possible de calculer les moyennes de temps de réaction par zones.

Ensuite, grâce à une fonction « envoi », on récupère le temps de réaction moyen par bouton qu'on envoie de façon fractionnée au Raspberry Pi pour la suite du traitement de données.

B. Programmation de l'interface graphique

Le choix du mode de jeu se faisait avec l'envoi d'un code ASCII par l'utilisateur, sur le port série de l'arduino depuis un ordinateur, ce qui demandait de connaître le caractère correspondant au mode choisi. L'objectif de l'interface homme/machine est de rendre l'utilisation de la machine plus intuitive et de mettre en place un suivi des performances des différents joueurs. Cette partie du projet s'est donc divisée en trois parties: la gestion de l'interface avec python et le module tkinter, la gestion des différentes données et guider l'utilisateur dans le choix du mode.

a. Choix des modes

Il existe 32 modes chacun définis par 4 options: l'allumage des leds au niveau des pieds ou non, l'allumage des leds extérieures ou non, l'allumage de leds rouges et le temps de la partie. Afin de permettre à un utilisateur ne connaissant pas les différentes options, leurs valeurs ou le code ASCII représentant l'ensemble de ses choix, chaque possibilité lui sera proposée et le caractère correspondant sera envoyé par le raspberry sur le port série de l'arduino.

Nous avons donc cherché un lien entre le code ASCII d'un mode et ses différentes options. Pour cela, en nous appuyant sur la notice des codes ASCII présentée en annexe 2, nous avons affecté une valeur à chaque choix possible, voir le tableau 1, puis nous avons établie une relation mathématique entre les différents possibilités de chaque options et le caractère du mode:

Code=vPieds+vDiff+vEv+vT avec Code la valeur du code ASCII.

vPieds est la valeur minimale du code du mode lorsque seule l'option "pieds?" est choisie, vPieds+vDiff à la valeur minimale du code lorsque seule les options relatives sont sélectionnée. La même méthode est utilisée pour déterminée vEv et vT.

Ainsi le choix de l'utilisateur pour chaque option sera traduit par l'affectation d'une valeur aux différentes variables qui permettent le calcul du code à envoyer à l'arduino.

option	choix1	valeurs hexadécimale du choix 1	choix2	valeurs hexadécimale du choix 2
pieds?	Avec	vPieds=2	Sans	vPieds=60
difficulté	jeune	vDiff=0	Pro	vDiff=8
evaluation	oui	vEv=0	non	vEv=4
Temps de partie	30s	vT=0	1m	vT=1
	1m30s	vT=2	2m	vT=3

Tableau 1: Tableau récapitulatif

b. Gestion des données

Nous avons décidé de stocker les données dans des fichiers Libreoffice Calc car c'est un logiciel libre déjà installé sur le raspberry, accessible par python grâce au module ezodf facile d'utilisation.

Pour chaque utilisateur, les données sont réparties dans deux fichiers: une base de donnée globale contenant la liste des joueurs et les informations relatives à leur dernier entraînement (mode, score et date), et une base de donnée personnelle listant l'ensemble des entraînements effectués. A chaque ajout de joueur, une ligne est ajoutée dans le premier fichier et un fichier personnel est créé en copiant un document source qui sert de modèle avec le module subprocess qui permet de lancer des commandes depuis python plutôt que depuis le terminal. Cette méthode à l'avantage sur celle de la création d'un tout nouveau fichier d'avoir directement les cellules et le tableau sous le format désiré et de recopier les macros ainsi que les événements qui les lancent. Les macros servent à calculer les différentes moyennes (totale, des leds à gauche, à droite, ...) et mettent en rouge le pire temps pour chaque led et en vert le meilleur pour chaque entraînement et la moyenne. Elles sont déclenchées par l'activation du document. Afin d'enregistrer les modifications, une macro est lancée pour enregistrer lorsque le document est désactivé.

Afin d'avoir accès à ces informations, le tableau des données personnelles et des courbes peuvent-être exportées vers un périphérique USB. Pour cela les fichiers calc seront d'abord exporter en pdf dans le même dossier puis copiés sur la clé USB grâce au module subprocess. Les courbes

sont tracées par python avec le module matplotlib.pyplot avant d'être enregistrée en format pdf et copiées sur la clé USB avec subprocess.

Les courbes représentent le temps du joueur en fonction du numéro de l'entraînement pour les leds ou l'ensemble des leds sélectionnées. L'entraîneur doit cependant avoir accès aux données de tous les joueurs. Nous avons donc décidé de distinguer joueurs et entraîneurs en donnant plus de possibilités à ces derniers qui pourront en plus avoir accès à la base de donnée totale.

Chaque utilisateur sera donc distingué par son nom, son prénom, son pseudo et son statut (entraîneur ou joueur). Afin de permettre une éventuelle confidentialité des données nous avons également décidé qu'un mot de passe serait attribué à chaque profil et à un administrateur qui pourra définir le statut de chaque membre. Une feuille du fichier calc des données globales ("Mdp administrateur") contient les différents mot de passes administrateur définis et une autre ("Paramètres") les différents paramètres tels que la distinction entre joueur et entraîneur, c'est-à-dire la nécessité d'un mot de passe pour avoir accès aux données, et la durée d'une saison. A chaque nouvelle saison une fichier calc de données globales est créé et la date de début de saison est celle de la création du fichier. L'ancien fichier est copié vers le dossier "archive", renommé en ajoutant la date du jour et les cellules contenant les informations sur les entraînements sont effacées. Ainsi la liste des membres est sauvegardée.

c. L'interface graphique

L'interface graphique est développée pour être affichée sur un écran raspberry 7" alimenté par le raspberry par GPIO.

La taille de l'écran imposant un affichage épuré et le fait qu'il soit tactile demandant que les zones pouvant être sélectionnées soient d'une taille suffisante pour permettre le confort de l'utilisateur, et l'insertion d'un clavier virtuel, le nombre d'objet graphique affiché à l'écran est limité. Nous avons donc décidé de n'afficher qu'une seule étape à la fois. La première fenêtre, la page d'accueil, permet de faire le choix de s'identifier et de s'enregistrer. Une fois le choix validé, il y a donc trois fenêtre qui peuvent apparaître:

- La fenêtre d'identification avec une ou deux entrées dépendant du mode d'identification choisi: nom et prénom, pseudo et mot de passe ou pseudo.
- La fenêtre d'inscription avec cinq entrées: nom, prénom, pseudo, mot de passe et la confirmation du mot de passe. Il y a également un checkbox qui permet de s'inscrire en tant qu'entraîneur lorsque ce statut existe.
- La fenêtre principale où le mode est choisi grâce la sélection des différentes options. Le bouton de lancement de partie ne s'affiche qu'une fois qu'elles sont toutes sélectionnées.

Lorsque le bouton valider de la fenêtre d'identification est appuyé, les informations rentrées sont vérifiées en les comparant à celle du fichier calc contenant les informations

sur tous les joueurs. Si elles sont incorrectes, un message d'erreur va s'afficher en indiquant que le membre n'est pas trouvé ou que le mot de passe est incorrecte en fonction de la valeur qui ne correspond pas à celle du fichier. Si la méthode d'identification est le nom et le prénom et que plusieurs membres ont les mêmes, le pseudo est alors demandé. Une fois identifié, l'utilisateur est dirigé vers la page principale. Les informations liées à son profil (nom, pseudo, date d'inscription) et à son dernier entraînement sont affichées et des boutons disposées comme les leds de la machine indique le dernier temps ainsi que le temps moyens lié à cette led lorsque le bouton est enfoncé. Si aucun temps n'est disponible, les leds sont désactivées et les informations sur le dernier entraînement sont remplacées par "Aucun entraînement".

Pour la fenêtre d'inscription, lorsque le nouveau membre a le statut de joueur, seule la longueur du mot de passe, qui doit être supérieure à cinq, la confirmation du mot de passe et le fait que le pseudo ne soit pas déjà utilisé sont vérifiés avant qu'une fenêtre propose de rester connecter avec ce compte. Si le bouton oui est appuyé, la page principale s'affiche sinon, l'utilisateur est redirigé vers la page d'accueil. Un fichier personnel est alors créé au nom du nouveau membre et une ligne est rajoutée dans le tableau principal. Si le nouveau membre a le statut de coach, le mot de passe administrateur peut être demandé en plus une fois que les informations du profil sont validées.

Lorsque le joueur est identifié, la fenêtre principale, permet, en plus de choisir son mode d'entraînement, d'exporter les courbes et tableaux grâce au "menu fichier" de la barre de menu et de modifier les paramètres de son profil (pseudo et mot de passe et le statut avec le mot de passe de l'administrateur) depuis le menu "option". Les entraîneurs peuvent également modifier et obtenir les documents d'autres profils et modifier les paramètres généraux: mode d'identification, distinction entre joueurs et entraîneurs.

Pour l'affichage du clavier virtuel, matchbox-keyboard, une commande lance son exécution en début de programme. Le clavier se place en haut de l'écran et la fenêtre de notre application le laisse apparaître en diminuant sa hauteur lorsque des entrées sont présentent sur la page.

d. Le programme

L'affichage de tout objet tkinter demande la création d'une racine (d'un root) qui servira de "support" à tous les autres objets. Notre root a le rôle de bureau dans notre application. C'est sur cet objet que les éléments qui sont présents à chaque étape, comme la barre de menu ou le bordereau du haut, et les fenêtres seront construitent.

Comme nous avons fait le choix de ne pas afficher nos fenêtres sur la même page, le programme python est construit avec plusieurs classes. ce qui donne plusieurs possibilités pour une fenêtre et permet donc le le retour vers les pages précédentes avec le bouton "retour", de demander le mot de passe administrateur à l'inscription comme lors

d'export de donnée ou de modification de profil ou paramètre.

Afin de rendre notre programme le plus modulable possible, nous avons protocolisé la création du plus d'objets possible et utilisé les méthodes tkinter les moins complexes. Nous avons par exemple, préféré utiliser la méthode grid pour l'affichage plutôt que la méthode pack car cette dernière propose de positionner les objets dans l'espace par rapport à l'objet parent (contenant l'objet) alors que grid divise le parents en différentes cellules et permets de positionner les enfants (objet contenu) en indiquant la ligne ou la colonne.

Les pages d'accueil, d'inscription, d'identification, de demande de mot de passe administrateur et celle du maintien de la connection, sont construites sur la même fenêtre (frame), ce qui permet de garder les même dimensions. Au début de chaque classe d'objet, les widgets (bouton, label, textes, entrée, etc) sont supprimés avant que ceux définis par la nouvelle classe soient affichés. Les autres fenêtres, liées aux fichiers et paramètres s'affichent au dessus de ces pages. C'est également le cas des messages d'erreur qui apparaissent dans des fenêtre pop-up. Afin de réduire le nombre de ligne du programme et de le rendre plus accessible, la création de certains objets s'est faite grâce à des fonctions "secondaires" tels que "makeentry" qui crée un label et un nom à partir de ses arguments d'entré, ou avec des class "secondaires" comme "makeledbutton" qui crée un bouton qui affiche le temps du joueur et qui le positionne selon l'angle et le rayon donné en argument. Le lien entre les différentes classes et fonctions est faite grâce aux variables globales.

Une variable globale est affectée à chaque paramètre et la valeur de celle-ci permet d'adapter l'affichage. Celles correspondant au mode d'identification, à la distinction entre joueur et entraîneur et à la protection des données par le mot de passe administrateur, et donc éventuellement au mot de passe administrateur, sont définies en début de programme en fonction des valeurs des cellules de la feuille "parametres" du fichier calc global. Par exemple, la demande du mot de passe administrateur n'est faite que lorsque la distinction entre joueur et entraîneur est faite, l'affichage de cette fenêtre dépend donc de la valeur de la variable correspondante. Les informations liées à l'utilisateur (nom, pseudo, profil, etc) sont stockées dans des variables globales dont la valeur est définie dans l'une des classes. Le numéro de ligne de la base de donnée globale correspondant au joueur et la dernière ligne de son fichier personnel correspondant au dernier entraînement sont également stocké dans dans des variables globales afin que des informations puissent être affichées et que les documents puissent être exportés lors de plusieurs étape mais également afin que les temps du joueur puissent être indiqués dans les tableaux lorsque la communication série avec l'arduino est finie.

C. Connexion entre Arduino et Raspberry Pi

b. Préambule :

Chaque mode de jeu (combinaison entre le nombre de led, l'utilisation ou non des leds rouges, le temps) est représenté par un caractère et son code ASCII.

Nous avons donc choisi d'envoyer directement à partir du code python situé sur la Raspberry, le caractère du mode choisi par l'utilisateur. Pour se faire, nous remplissons une liste avec des 0 et des 1 en fonction des critères (avec/sans pieds, Jeune/Pro, Entrainement/Evaluation, Temps) choisis par le joueur. Par exemple, si le joueur choisit le critère « sans les pieds », la liste se remplit de la combinaison suivante : [1 0]. Chaque liste propre à chaque mode de jeu est ensuite associée à son caractère. C'est ce caractère qu'on envoie par la suite au Raspberry.

c. Liaison Python-Arduino :

La première étape de réflexion fût de mettre en évidence quel type de données était envoyé par la Raspberry et donc en Python afin de commander l'arduino. Comme expliqué précédemment, nous avons choisi d'envoyer un caractère. Ceci peut être réalisé en encodant en ASCII le caractère avant de l'envoyer sur le port série. Une fois réceptionné côté Arduino, ce code ASCII est lu comme un caractère. Or, le code Arduino nécessite la valeur ASCII du mode envoyé afin de réaliser la fonction de démarrage. C'est pour cela que nous avons dû convertir le char lu sur le port série en un entier.

La deuxième étape fût de traiter les données renvoyées par l'Arduino. Ces données sont de type entier. Le problème ici était d'envoyer un entier supérieur à 500 (temps de réaction) sur le port série. Ne pouvant l'envoyer dans son intégralité (car seul un code ASCII ou un byte peut être transmis sur la liaison série), nous avons dû réfléchir à un moyen de l'envoyer de façon segmentée. Or un entier supérieur à 500 peut être codé sur deux bytes qui peuvent être envoyés sur le port série puis assemblé côté Python. Ainsi, en sachant qu'un byte peut coder un entier jusqu'à 255, nous avons récupéré le quotient et le reste de la division de notre entier à envoyer par 256. Ces deux valeurs obtenues ont ensuite été converties en byte envoyé sur le port série puis récupérés par le code Python qui les retraduit en entier (grâce à la fonction `ord`) et les rassemble pour former l'entier.

Afin de vérifier le bon fonctionnement de ces programmes, nous avons fait un test qui consistait à envoyer un caractère depuis le code Python. Ce code une fois lu par l'Arduino permettait de lancer une boucle commandant l'allumage d'une led. De plus, l'appui sur un bouton entraîne l'envoi d'un entier converti sur deux bytes récupéré par le code Python et rassemblé. Ces codes de test se trouvent en annexe 3.

D. Conception mécanique :

Cette partie traitera sur la conception mécanique de la machine

a. Analyse du besoin

Le VAFC React est une machine destinée aux joueurs professionnels de football pour définir et améliorer leur temps de réaction et donc leurs performances. Pour ce faire l'entraîneur ou le joueur sélectionne son programme d'entraînement sur l'interface homme machine réalisée lors de ce projet. Pour permettre de l'intégrer au système, un écran LCD nous a été fourni. Nous devions donc concevoir un système permettant de le fixer sur la machine. Ce système avait pour contrainte de pouvoir être mobile, il ne devait pas être fixé et orienté vers l'arrière de la machine car peu accessible. L'entraîneur doit pouvoir avoir accès facilement à cette tablette pour pouvoir contrôler facilement le jeu. C'est pourquoi la conception devait se tourner vers un mécanisme de bras pliable et dépliable.

b. Conception sur l'existant

Ayant la machine à notre disposition, il était facile de se familiariser avec la structure. Après quelques manipulations et quelques mesures, nous trouvons plusieurs concepts possibles pour la réalisation de ce bras mécanique. Nous devions résoudre plusieurs problématiques:

- Localiser la zone de fixation du bras. En effet, en position fermée, la machine s'appuie sur différentes zones centrales, c'est pourquoi la zone de fixation devait être définie en premier lieu. De plus le bras ne doit pas venir en contact avec la partie avant de la machine, d'où la nécessité de trouver une zone adéquate.
- Pouvoir maintenir une position fermée. Le bras ne doit pas pouvoir être mobile lors du passage « ouvert/fermée » de la machine. Cette contrainte permet la sécurité de l'utilisateur, des différents composants ainsi que du mécanisme.
- Réaliser simplement les différentes liaisons permettant au bras de fonctionner. Cette problématique, une fois la première résolue, nous a permis de nous orienté vers des composants standard du marché pour rester sur une solution simple.

Une fois les différents besoins et les différentes problématiques identifiées, nous nous sommes réunis lors d'une réunion de concepts pour proposer différentes idées de réalisation. Pour ce faire nous avons séparé la conception mécanique en trois parties distinctes, chaque partie a été validée grâce à plusieurs matrices de décisions. Nous devions nous mettre d'accord sur le système de bras utilisé : ouverture-fermeture en portefeuille ou un bras sur un rail que l'on viendrait tirer. Nous devions également nous mettre

d'accord sur le système maintenant l'écran LCD (système perche à selfie ou simple ajout de matière sur la pièce support écran). Enfin nous devions valider le concept permettant la liaison pivot entre le bras et la pièce de support écran. Cette étape nous a permis d'effectuer des recherches dans des composants du commerce et donc de comparer différents composants satisfaisant au maximum notre besoin.

Beaucoup de concepts ont été proposés, il nous est alors paru judicieux de les comparer pour choisir la solution respectant le besoin et étant la plus simple à réalisée.

Après quelques échanges avec Monsieur Dupont ainsi que Monsieur Lesbros, la solution de recréer la pièce support écran dans une chute d'aluminium a été choisie. De plus, le bras et la pièce permettant le pivot du bras seront usinés dans les chutes des pièces existantes du VAFC React. Seule la charnière et le système de maintien en position fermée seront à commander. Une pièce permettant la fermeture du système sera usinée, celle du composant standard ne nous permettant pas de la fixer sur notre pièce support écran (épaisseur de notre pièce trop fine).

c. Commande des composants

Une fois le concept choisi, et avant de commencer la conception, nous devions trouver le plus de composants standard pour limiter la conception et la fabrication de pièce. Nous avons donc cherché des charnières du commerce. Après sélection de plusieurs composants, une matrice de décision, créée à partir de critères précis, a été établie pour sélectionner celle qui satisfera le mieux notre solution. Les critères établis sont les suivants:

- Coût : plusieurs composants ont été sélectionnés, le coût doit être le plus faible possible.
- La fonction est respectée : le système remplit-il sans complexité l'ouverture et la fermeture du bras mécanique.
- Les plans complets sont disponibles : pour permettre la modélisation des différentes pièces support écran et pièce de renfort, il nous faut posséder les dimensions précises du composant pour permettre l'usinage des pièces sans devoir attendre de les avoir en main propre.
- Le maintien en position fermé : est-il possible ou non pour la pièce de garantir le maintien du système en position fermée, notamment au niveau des pivots (axe à serrer).
- Solidité du système : le composant n'est-il pas trop fin pour garantir l'intégrité de notre système.

Comme il était difficile de garantir le maintien en position fermée du système, nous devions également trouver un système de fermeture. Après quelques recherches dans le

commerce, nous trouvons le composant qui répondrait aux besoins. Néanmoins, l'épaisseur de notre pièce support écran ne nous permet pas de fixer ce composant. Nous devons donc recréer une partie de ce composant à l'aide des plans disponibles.



Figure 6: Charnière de la liaison pivot entre le bras et le support écran



Figure 7 : Système de fermeture choisi

d. Modélisation CAO : usinage des pièces/fabrication

La réalisation de chaque pièce s'est fait sous le logiciel AUTODESK. Nous avons décidé de conserver la forme support de l'écran en lui rajoutant sur le côté un ergot pour permettre la fixation d'une charnière coudée. Nous avons donc modélisé cette pièce avec les dimensions nécessaires et effectué les modifications nécessaires. Cette pièce est entièrement usinée dans une plaque d'aluminium fournie par Mr Dupont.

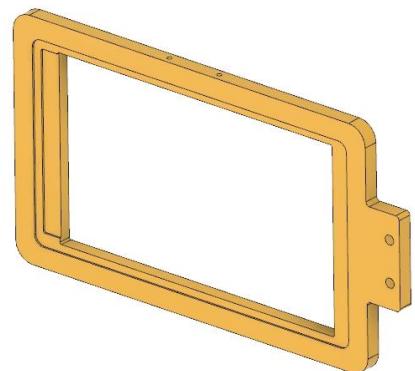


Figure 8: Conception assistée par ordinateur de la pièce support écran

Après avoir fait le choix du système de fermeture, nous avons donc recréé une pièce pour permettre de la fixer sur la pièce en aluminium.

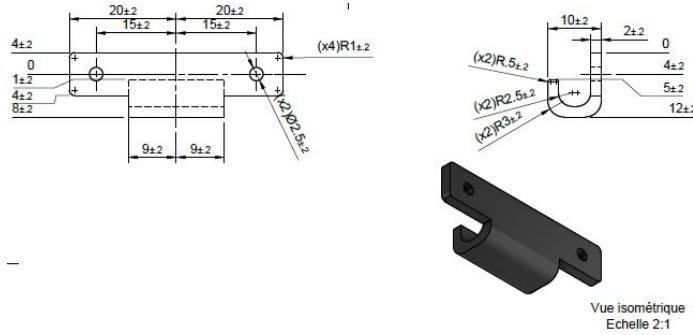


Figure 9: Plan de la pièce fermeture du système

La fabrication des pièces se fait par Monsieur Lesbros. Celui-ci a également soudé les pièces nécessaires sur la machine existante.

e. Tests de validation

Des protocoles de tests ont été rédigés, et un compte rendu de ces différents tests a été écrit pour valider ou non le système. Tous ces tests se trouvent dans le document *compte rendu rapport de tests partie mécanique*. Ces tests ont pour but de valider ou non les différentes pièces mécaniques, valider l'usinage, la conformité des pièces, de l'assemblage, mais également de vérifier si la fonction désirée est remplie.

Cette étape est essentielle dans la conception d'un système. De cette manière, il a été observé que la pièce support écran avait quelque défaut. L'écran de la Raspberry ne rentrait pas dans la pièce usinée en impression 3D, l'emplacement de celui-ci n'était pas assez grand. Pour remédier à ce problème, il a été décidé de limer la pièce pour pouvoir faire tenir l'écran. Les fichiers CAO ont été modifiés par la suite pour garantir la fixation de l'écran. De plus, la machine permettant d'usiner la pièce en aluminium ne fonctionnait pas, la pièce support écran devra rester en ABS. Pour garantir une certaine solidité du système, la pièce a été modifiée pour permettre de rigidifier le système. De plus un test avec la charnière a été effectué pour contrôler l'alignement des trous et leur bon positionnement. La liaison pivot entre la machine et le bras a révélé un jeu. Ce jeu rendait le système moins stable. Une rondelle a été ajoutée pour permettre de stabiliser l'ensemble. Celle-ci ne gêne pas la rotation du bras. Les tests ont montré que le bras rendait l'écran accessible au joueur dans la zone de jeu. Celui-ci pourra donc choisir son mode d'entraînement sans faire des allers-retours derrière la zone de jeu. En ce qui concerne la charnière reliant le bras et l'écran, un jeu est présent, obligeant l'utilisateur à maintenir la tablette lorsqu'il veut effectuer des saisies sur l'interface homme/machine. Un test en position fermé a montré que le concept sélectionné fonctionne facilement. La machine garantit l'intégrité des composants dans n'importe quelle

position.



Figure 10: Assemblage du bras sur le VAFC React



Figure 11: Test en position fermée



Figure 12: Bras en position ouverte

De plus, lors de l'avancement du projet, une question est apparue : comment alimenter le Raspberry Pi ? Devant être alimenté sur secteur, il faut donc un câble assez long pour pouvoir relier le secteur à la Raspberry. Deux solutions semblent les plus réalisables. L'intégration d'une batterie sur le système mécanique est une solution. Néanmoins celle-ci entraîne un modification du bras ou de la pièce contenant la Raspberry. De plus, la machine est reliée au secteur. Il suffirait alors d'intégrer une multi-prise sur le système pour donner l'accès facilement à la Raspberry Pi.

V. CONCLUSION ET PERSPECTIVES

Ce projet a pour but d'améliorer la machine du VAFC React, notamment avec la création d'une interface Homme-Machine appropriée.

Premièrement, nous avons cherché à appréhender dans son intégralité le projet déjà réalisé avec les anciens élèves de l'ENSIAME afin de comprendre au mieux leur démarche de fabrication pour la machine mais aussi pour ce qui concerne la programmation du jeu de LEDs et des boutons. Ensuite, nous avons procédé aux répartitions des différentes missions pour le bon déroulement du projet.

Chacune de nos missions rendent le projet très complet et nous ont confrontées aux problématiques de l'ingénieur mécatronicien. En effet, nous avons dû adapter et modifier un code Arduino, créer un code python pour le Raspberry Pi pour la programmation de l'interface, créer une liaison série entre les deux microcontrôleurs et réaliser un bras mécanique mobile afin de supporter l'écran tactile. Tout ceci grâce à une cohésion de groupe.

Nous avons ainsi réussi à respecter une grande partie du cahier des charges. Nous avons maintenant une interface homme-machine programmée sur Raspberry Pi communiquant avec l'Arduino qui permet le calcul du temps de réaction. Cependant, nous avons négligé certaines missions. La maquette soudée n'a pas pu être réalisée. Les modifications du code Arduino afin de pouvoir travailler des zones ciblées n'a pas pu aboutir entièrement. Nous n'avons pas pris le temps d'apporter les réparations nécessaires au niveau des LEDs des pieds.

Lors de ce projet, nous avons aussi fait face aux avantages et aux inconvénients du travail de groupe. Effectivement, l'organisation des tâches nous a permis de travailler efficacement et rigoureusement chacune de notre côté. Chaque membre de l'équipe avait son rythme de travail. Par ailleurs, nous avions une bonne entente entre nous, ce qui a permis de partager chacune de nos expériences et découvertes. Grâce aux réunions de mises en commun, nous étions au courant de toutes nos avancées respectives. Cependant, lorsqu'une de nous rencontre des difficultés, nous n'avons pas su réagir à temps car nous étions concentrées chacune dans nos missions respectives. Ceci a causé quelques retards et des missions non accomplies. Nous avons donc appris qu'analyser les problèmes de chacun des membres de l'équipe et réagir à temps est important afin que cela ne nuise pas au bon déroulement du projet.

Le projet pourrait être encore amélioré sur plusieurs aspects que nous explicitons succinctement dans la partie suivante.

Tout d'abord, notre interface, certes fonctionnelle, pourrait être encore développée pour atteindre une ergonomie et un esthétisme permettant sa commercialisation.

Ensuite, nous pourrions continuer de travailler sur le code Arduino afin de travailler sur des zones plus ciblées tout en intégrant le jeu entier de LEDs pour renforcer la rapidité de réaction d'un joueur dans ses zones de faiblesse. D'autre part, nous pourrions ajouter un bruit afin de perturber le joueur et améliorer ainsi son temps de réaction. La possibilité de motoriser les différents bras de l'Arduino rendrait l'expérience du jeu plus dynamique et plus dure. On approcherait ainsi l'exercice idéal pour travailler la réaction pour un joueur de football.

En conclusion, ce projet nous a permis de travailler entièrement en tant qu'ingénieur mécatronicien. D'une part, nous avons énormément appris sur les principaux enjeux de ce domaine qui sont l'utilisation de deux microcontrôleurs, la communication entre eux, la programmation ou encore la conception mécanique. D'autre part, celui-ci a été réalisé sur toute la période du semestre 2 et est donc un projet de plus grande envergure que ce dont on a eu l'habitude. Il nous a appris à coopérer afin de contribuer à sa réussite. Par conséquent, l'expérience apportée par ce projet a été autant enrichissante sur le plan humain que professionnel pour chacune d'entre nous.

ANNEXE

Annexe1: PDCA (Plan de convergence)

	R	Résultat non obtenu
	Partie mécanique	Partie électronique
Une maquette de 15 jets est réalisée.	Fatima Achbah 31-May O	
Tester le code existant sur la maquette.	Arunjah Jayaseelan 03-Jun V	
La liste détaillée de tous les composants nécessaires est établie.	Arunjah Jayaseelan 18-May V	
L'emploi du temps des intervenants est pris en compte.	Camille Fournet: 19-Apr V	
La répartition des tâches selon le diagramme de GANTT est établi.	Camille Fournet: 20-Feb V	
La documentation de suivi de projet est établi.	Equipe 06-Jun V	
Des axes d'améliorations sont étudiés.	Equipe 27-Apr O	
La matrice de décision concernant le choix de programmation est identifiée.	Arunjah Jayaseelan 22-Feb V	
Le concept du bas de l'interface homme-machine est identifié.	Camille Fournet: 23-Mai V	
Le cahier des charges fonctionnel est défini.	Equipe 22-Feb V	
L'analyse de l'existant est réalisée.	Equipe 20-Feb V	
L'organigramme de l'interface homme-machine est défini.	Barbara Kandem 16-Mai V	
Le système mécanique est défini.	Camille Fournet: 23-Mai V	
Le langage de programmation est défini.	Aïdeen Hervé 22-Feb V	
Les pièces commandées sont conformes.	Camille Fournet: 31-May V	
Le diagramme de défaillance est réalisé.	Arunjah Jayaseelan 24-May V	
Les fournisseurs sont identifiés.	Camille Fournet: 24-May V	
L'organisation du stockage des documents.	Barbara Kandem 07-Jun V	
La matrice de décision concernant la chose de fabrication du support bras est défini.	Camille Fournet: 17-May V	
P (Prévoir)	D (Piloter)	C (Vérifier)
		A (Pérenniser)

Annexe2: Tableau code ASCII

Pieds?	Dificulté?	Evaluation?	Temps de partie	Code ASCII du mode
Sans les pieds	Jeune	Entrainement	30s	B
			1m	C
			1m30s	D
			2m	E
		Evaluation	30s	F
			1m	G
	Pro	Entrainement	1m30s	H
			2m	I
		Evaluation	30s	J
			1m	K
			1m30s	L
			2m	M
		Evaluation	30s	N
			1m	O
			1m30s	P
			2m	Q
Avec les pieds	Jeune	Entrainement	30s	Z
			1m	3
			1m30s	4
			2m	5
		Evaluation	30s	6
			1m	7
	Pro	Entrainement	1m30s	8
			2m	9
		Evaluation	30s	:
			1m	;
			1m30s	<
			2m	=
			30s	>
			1m	?
			1m30s	@
			2m	A

Annexe3:

Code Python :

```
import serial
def test_liaison():
    ser = serial.Serial('COM7', 9600) #liaison avec le port série
    ser.write('H'.encode()) #encodage du caractère en ASCII et écriture sur le port série
    while 1:
        if(ser.inWaiting() > 0):
            val2 = ser.read() #lecture du HSB de la valeur envoyée
            val3 = ser.read() #lecture du LSB
            i1 = ord(val2) #conversion byte->entier
            i2 = ord(val3)
            val = i1 * 256 + i2 #reconstruction de la valeur
            print(val) #affichage de la valeur pour vérification
```

Code Arduino:

```
int pinBouton;
int pinLed;
char mode;
void setup() {
pinBouton = 2;
pinLed = 4;
Serial.begin(9600);
pinMode(pinLed, OUTPUT);
pinMode(pinBouton, INPUT);
}
void loop() {
if (Serial.available()) //s'il y a des données qui arrivent
{
mode = Serial.read(); //Lecture d'un entier sur le
```

```
tampon série
if (mode == 'H')
{
int mode0 = (int)mode; //conversion du char à valeur ASCII
Serial.print(mode0); //affichage de l'integer verification
digitalWrite(pinLed, HIGH);
delay(1000);
digitalWrite(pinLed, LOW);
}
}
boolean etatBouton = digitalRead(pinBouton); //de l'état du bouton (appuyé ou non)
if (etatBouton == HIGH)
{
int val = 500; //valeur à envoyer, représentée par bytes
int Hval = val/256;
int Lval = val%256;
byte val2 = (byte)Hval;
byte val3 = (byte)Lval;
Serial.write(val2);
delay(200);
Serial.write(val3);
}
delay(2000);
Serial.flush(); //On vide le tampon
}
```

• BIBLIOGRAPHY

- [1] [Online]. Available: https://fr.wikipedia.org/wiki/Entra%C3%A9nement_sportif
- [2] C. Pociello, La science en mouvement.
- [3] T. M. Rivolier, Le psychologie du sport de haut niveau.
- [4] L. R. & Y. Léziart, L'homme en mouvement.
- [5] R. W. & D. Gould, Psychologie du sport et de l'acquisition physique.
- [6] [Online]. Available: https://fr.wikipedia.org/wiki/Entra%C3%A9nement_sportif
- [7] [Online]. Available: https://fr.wikipedia.org/wiki/Pr%C3%A9paration_en_sport.