



Rapport Projet d'Integration

16 SEPTEMBRE 2024 - 27 MARS 2025

THIBAUD PICCINALI

Tuteurs parcours/filière

Alexandre KRUSZEWSKI
Riad BRAHAM

Tuteur projet

Thomas
BOURDEAUD'HUY

Table des matières

Remerciements	2
Introduction	3
1 Contexte du projet	4
1.1 De l'arcade bartop	4
1.2 ... aux fléchettes!	4
2 Le projet	4
2.1 La conception logicielle	6
2.1.1 L'unité centrale	6
2.1.2 Le serveur web	8
2.2 La réalisation de la structure	9
3 Détection de la position de la fléchette	11
3.1 Principe de triangulation	11
3.2 Solution implémentée	12
3.2.1 Calibration des caméras	12
3.2.2 La détermination des pixels d'intérêts	13
4 Budget	16
5 Gestion de projet	17
5.1 Réunions	17
5.2 Outils utilisés	17
6 Ressentis	17
6.1 Ressentis vis à vis du travail proposé	17
6.2 Ressentis vis à vis du projet dans son ensemble	17
7 Conclusion	18

Remerciements

Je tenais tout d'abord à remercier Monsieur THOMAS BOURDEAUD'HUY pour avoir organisé ce projet d'intégration. Je souhaitais notamment le remercier pour son accompagnement et ses excellents conseils qu'il a pu me partager.

Je voulais également remercier Monsieur PATRICK GALLAIS et Monsieur FABIEN VERBRUGGHE pour leurs aides et expertises techniques qu'ils ont pu m'apporter notamment vis à vis de la réalisation de la structure au Fablab.

J'adresse également des remerciements à Messieurs ALEXANDRE KRUSZEWSKI et RIADH BRAHAM pour leurs implications en tant que tuteur parcours et filière.

Enfin je souhaitais remercier l'Ecole Centrale de Lille, pour avoir permis de rendre ce projet possible.

Introduction

Dans le cadre de ma dernière année à l'Ecole Centrale de Lille, je devais réaliser un projet d'intégration. Ce projet vise à intégrer les connaissances et compétences acquises au cours de la formation et permet aussi de renforcer la personnalisation et la cohérence d'ensemble. Il permet notamment de consolider des acquis méthodologiques, particulièrement en gestion de projet.

Pour le choix du sujet, j'ai souhaité approfondir un domaine que j'avais pu explorer en stage : celui de la *computer vision*. La *computer vision*, est une branche de l'informatique qui a pour but de permettre à une machine de traiter et d'analyser des images. C'est une technologie qui est particulièrement utilisée par des systèmes embarqués, ce qui fait écho à mon choix de parcours (*Systèmes embarqués et Cyberphysique*). C'est donc avec cet objectif que je me suis mis en recherche. Je me suis ainsi rapproché de Monsieur BOURDEAUD'HUY qui proposait un sujet dans ce domaine. Après un premier entretien, Monsieur BOURDEAUD'HUY a accepté de m'accompagner sur ce projet.

Au travers de ce document, je souhaitais faire un rapport de l'ensemble des travaux que j'ai pu effectuer au cours de ce projet. Pour mener à bien cette idée j'adopterais le plan suivant. Dans un premier temps, je tâcherais d'expliquer le principe du sujet. Fort de cette introduction, j'expliquerais les tâches sur lesquelles j'ai travaillé. Je détaillerais notamment la partie technique qui représente la particularité du sujet. Suite à cela, je présenterais la partie gestion et organisation du projet. Enfin je clôturerais ce rapport en donnant mon ressenti personnel ainsi que les limites et perspectives d'évolutions.

1 Contexte du projet

1.1 De l'arcade bartop ...

L'origine de ce projet remonte à une initiative précédemment menée au sein de l'école. En effet, il y a quelques années, l'association du personnel avait organisé un atelier de fabrication de bornes d'arcade bartop. Ces bornes (voir figure 1) permettent de rejouer à plusieurs grands classiques du jeu vidéo. Grâce à cette activité, les membres du personnel ont eu l'opportunité de concevoir et d'assembler leur propre borne d'arcade, qu'ils pouvaient ensuite rapporter chez eux. Cette activité a suscité l'intérêt de Monsieur BOURDEAUD'HUY qui souhaitait pourvoir proposer ce concept mais autour d'un jeu différent.



FIGURE 1 – Exemple d'une borne arcade bartop (*source*)

1.2 ... aux fléchettes !

C'est dans cette optique que Monsieur BOURDEAUD'HUY s'est intéressé au jeu de fléchettes, envisageant celui-ci comme une suite à l'activité de construction d'arcade. Dans sa version du jeu de fléchette, il souhaitait intégrer un système capable de détecter automatiquement les lancers et de calculer le score de chaque joueur et ce, avec une cible et des fléchettes traditionnelles. C'est ainsi qu'est née l'idée de ce projet d'intégration : concevoir un système de comptage de points pour un jeu de fléchette traditionnel et facilement reproductible.

2 Le projet

Le principe global du projet est résumé sur le diagramme de cas d'utilisation ci-après (figure 2). Le joueur, à l'aide d'un appareil connecté (ordinateur, tablette...), doit pouvoir se connecter au système, consulter la table des scores et gérer la partie (paramétrier le nom des joueurs, recommencer une partie...). A chaque interaction avec la cible (arrivée d'une nouvelle fléchette par exemple), le système doit pouvoir tenir automatiquement la table des scores à jour.

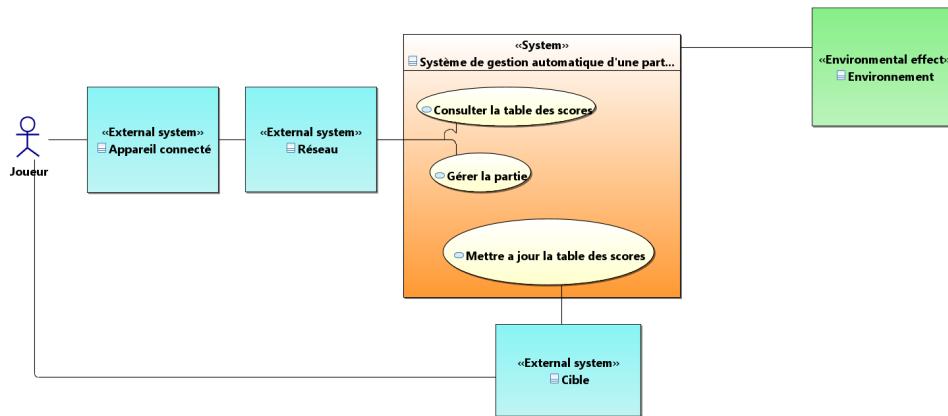


FIGURE 2 – Diagramme de cas d'utilisation du projet

L'ensemble de ces cas d'utilisation induisent des exigences sur le système. Ces dernières sont détaillées sur le diagramme d'exigence ci-après (figure 3). Beaucoup d'exigences sont présentées mais je souhaitais m'attarder sur celles que je juge comme étant les plus importantes. L'exigence principale du système est de pouvoir gérer de manière automatique le déroulement d'une partie de fléchette. Cela induit des exigences indiquant que le système doit permettre de détecter l'arrivée d'une nouvelle fléchette sur la cible et calculer le score associé. Le système doit aussi permettre aux joueurs de consulter/modifier l'état de la partie. Enfin, on note deux dernières exigences importantes : le système doit être facilement reproduis (avec des outils et du matériel de l'école) et ce à bas coût (moins de 150€).

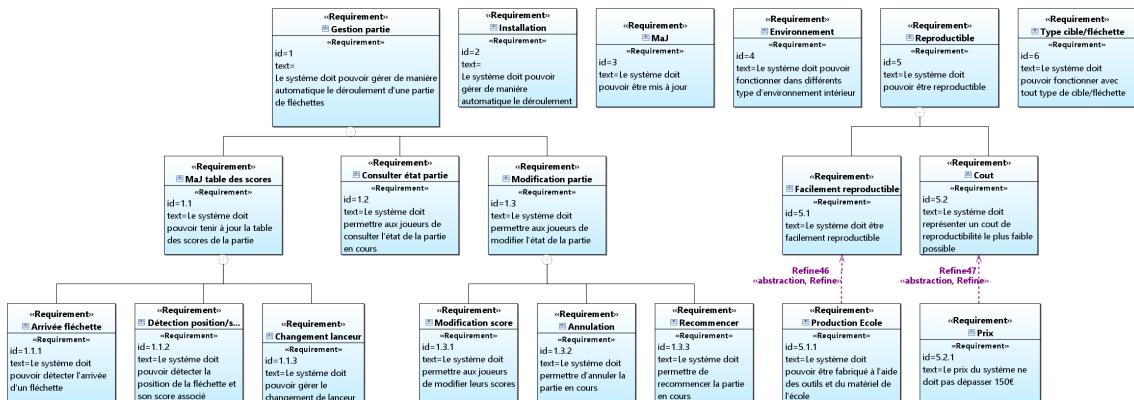


FIGURE 3 – Diagramme d'exigences du projet

Pour répondre à l'ensemble de ces exigences, j'ai décidé de modéliser le système de la manière suivante (voir figure 4). La partie principale est une **Unité Centrale** qui doit se charger d'effectuer l'estimation des positions des fléchettes et la mise à jour des scores. Cette unité dialogue avec des caméras (externes au système) qui observent la cible (elle aussi externe). L'utilisation de caméras induit le choix fait pour l'estimation des positions qui sera présenté dans la partie suivante. Une **Structure** est également présente et permet de maintenir l'ensemble de ces éléments. Enfin, un **Serveur Web** servira d'intermédiaire entre le joueur et l'unité centrale et permettra à ce dernier d'agir sur cette unité via une interface web.

Avec ce choix de modélisation, j'espère pouvoir répondre à l'ensemble des exigences du cahier des charges, notamment celles concernant la facilité et le faible coût de reproduction.

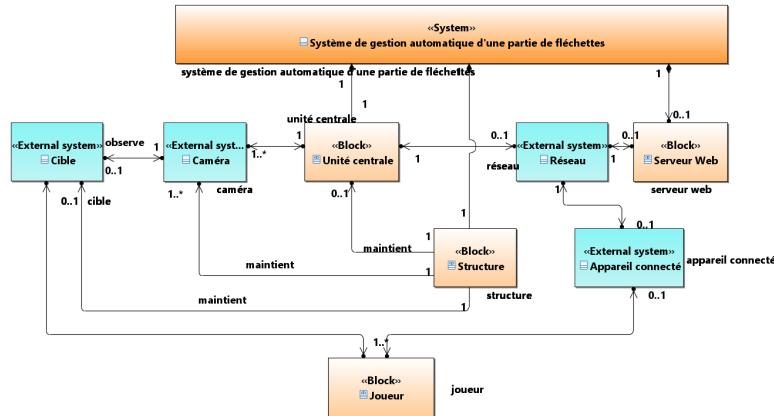


FIGURE 4 – Diagramme de bloc du système

Ce système est donc composé de deux parties principales : la structure permettant de maintenir l'ensemble de ses composantes et les logiciels (unité centrale et serveur web) permettant de gérer la partie de manière automatisée. Je souhaitai détailler dans cette section ces deux points.

2.1 La conception logicielle

2.1.1 L'unité centrale

La conception de cette unité centrale représente le cœur du projet. Au cours de ce dernier, différentes méthodes et architectures ont été testé. Particulièrement, en ce qui concerne le langage de programmation. Si dans les premières versions, l'intégralité du code était écrit en Python (ce qui permettait de faire des itérations de tests rapides) dans sa dernière version, la totalité du code est écrit en C++.

Cette unité est organisée selon le diagramme de blocs interne suivant.

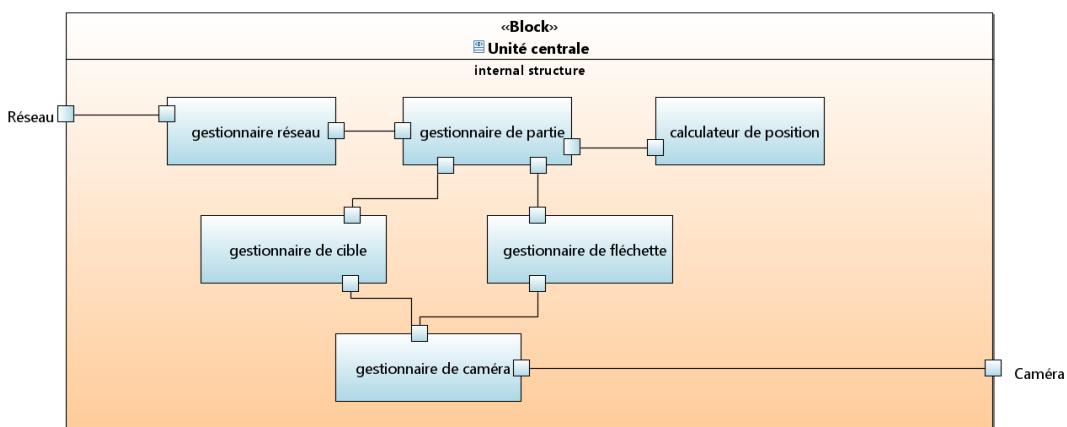


FIGURE 5 – Diagramme de bloc interne de l'unité centrale

Six processus s'exécutent en même temps, en parallèle en s'échangeant des informations à l'aide d'une mémoire partagée.

- Le premier est un **gestionnaire réseau** (basé sur TCP) en mode serveur, pouvant recevoir des commandes du client distant et les traiter. Ces commandes prennent en compte : l'initialisation et l'arrêt d'une partie, la modification de scores ou la récupération des informations de la partie en cours.
- Le deuxième agit comme un **gestionnaire de la partie** selon les ordres reçus du gestionnaire réseau. C'est ce processus qui va coordonner les autres, demander le calcul de position d'une fléchette, maintenir la table des scores à jour, compter combien de fléchettes ont été lancées...
- Le troisième, agit comme un **gestionnaire de fléchette**. Son rôle est de déterminer l'arrivée d'une fléchette sur la cible.
- Le quatrième est le **gestionnaire de caméra**. Ce processus lit les images renvoyées par ces dernières. S'il détecte un changement entre deux images, il informe les autres processus et sauvegarde ces images.
- Le cinquième est un processus chargé de **calculer la position de la fléchette**. Il est commandé par le gestionnaire de partie et utilise les images enregistrées par le gestionnaire de caméra pour réaliser ses estimations.
- Le dernier est le **gestionnaire de cible**. Il agit de manière similaire au gestionnaire de fléchette : son rôle est de déterminer lorsque la cible a été vidée de ses fléchettes par un joueur.

L'ensemble de ces processus interagissent selon le diagramme de séquence suivant :

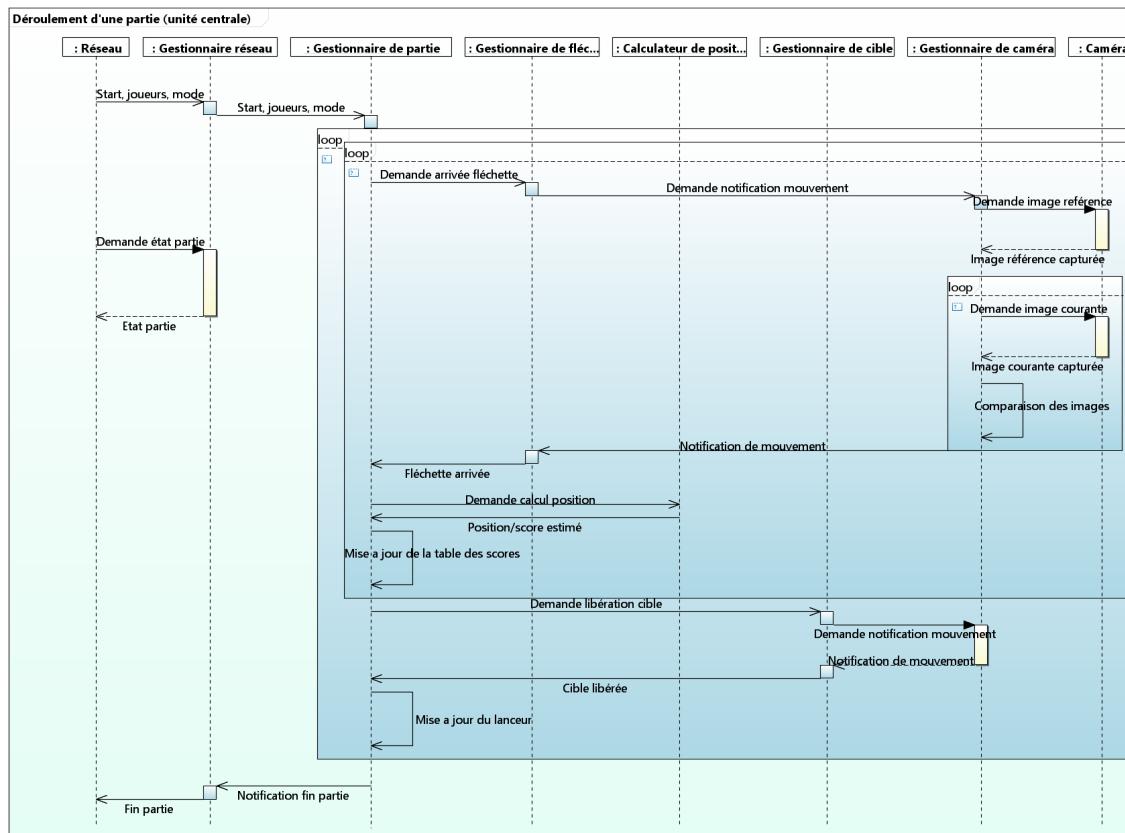


FIGURE 6 – Diagramme de séquence de l'unité centrale

La partie est lancée via l'intermédiaire du réseau et de son gestionnaire qui transmet au gestionnaire de partie les noms des joueurs et l'autorisation de commencer. Ensuite, tant que la partie n'est pas terminée (i.e. tant qu'aucun des joueurs n'atteint le score de 0), deux mécanismes vont s'exécuter. Le premier est relatif au lancer des fléchettes par un joueur. D'abord, le gestionnaire de partie demande au gestionnaire de fléchette de le notifier à l'arrivée d'une nouvelle fléchette sur la cible. Pour se faire, ce dernier va dialoguer avec le gestionnaire de caméra qui, en commandant la prise d'images et en les comparant entre elles, pourra déterminer lorsqu'une fléchette est arrivée. Une fois l'information remontée jusqu'au gestionnaire de partie, ce dernier pourra demander au calculateur de position d'estimer le score associé. Lorsque que le joueur aura lancé toutes ses fléchettes (dans une partie classique, chaque joueur lance trois fléchettes à son tour) ce gestionnaire demandera au gestionnaire de cible de le notifier lorsque cette dernière sera libérée (pour se faire il utilisera un mécanisme similaire à celui de la détection de l'arrivée d'une fléchette). Le gestionnaire de partie pourra ensuite mettre à jour la table des scores. A noter qu'à n'importe quel moment le gestionnaire réseau traite les requêtes externes permettant notamment d'exporter l'état de la partie.

Le schéma de Petri suivant permet également d'observer l'interaction entre ces processus.

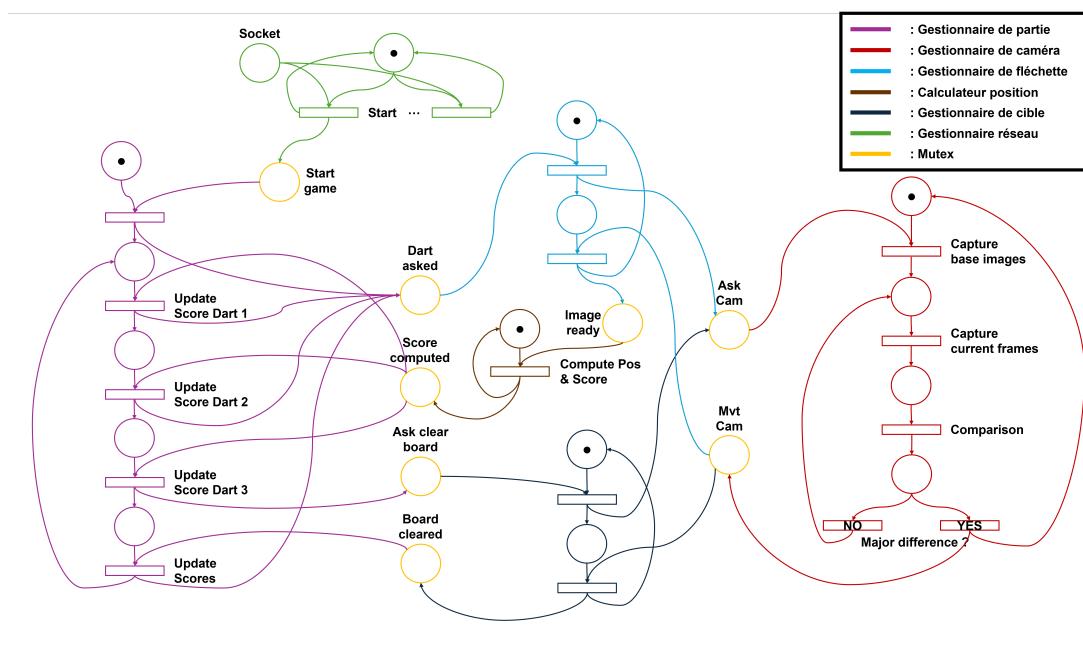


FIGURE 7 – Schéma de Petri de l'unité centrale

2.1.2 Le serveur web

L'objectif du serveur web est de permettre une gestion plus simple des parties via un site internet, offrant ainsi une expérience de jeu plus agréable qu'une interaction uniquement en ligne de commande. Ce serveur, développé en Python, utilise le micro-framework *Flask* pour gérer la logique backend et les requêtes HTTP. C'est ce script qui initialise l'application et définit les routes pour traiter les requêtes des utilisateurs.

Côté frontend, HTML permet de structurer les pages, CSS d'y appliquer un style, et JavaScript pour rendre l'application interactive, notamment en gérant les événements et en envoyant des requêtes asynchrones (AJAX). Ce serveur communique avec le jeu via l'interface réseau de l'unité centrale présentée précédemment.

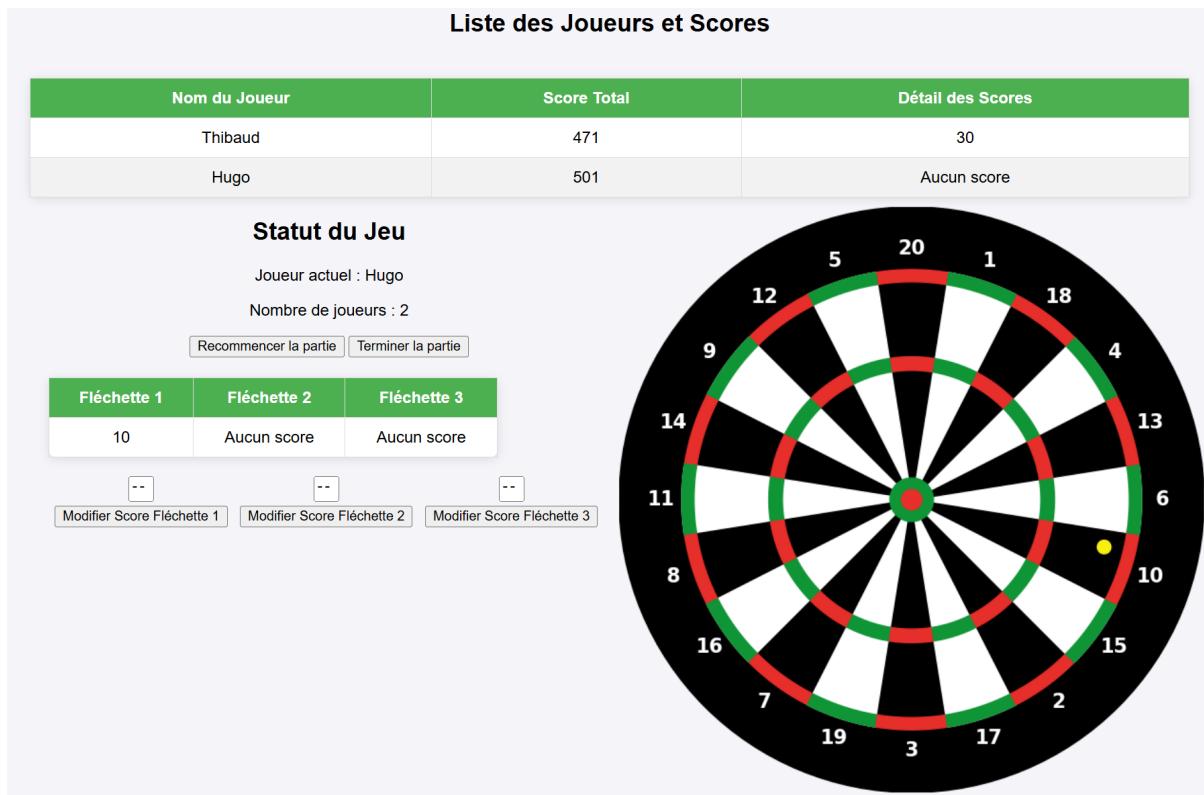


FIGURE 8 – Capture d'écran de l'interface web permettant de piloter la partie

2.2 La réalisation de la structure

La réalisation d'une structure rigide est cruciale pour le bon fonctionnement du système. En effet, comme expliqué précédemment, des caméras vont être utilisées pour obtenir des images de la cible et ainsi détecter les positions des fléchettes. Assurer que ces caméras soient correctement positionnées et qu'elles soient dans l'impossibilité de bouger est donc primordial pour permettre de capturer des images de bonne qualités. De plus, pour améliorer la robustesse des algorithmes de détection, cette structure doit permettre d'accueillir des luminaires qui se chargeront d'assurer une luminosité constante pour les caméras (et donc d'améliorer la qualité de traitement des images).

Plusieurs versions de cette structure ont ainsi été imaginées. La dernière version est présentée sur la figure 9. On peut noter deux points importants sur cette version. D'abord la présence de trous rectangulaires sur les côtés permettant d'y fixer les caméras. Aussi, la présence de jointures adaptées à la fixation entre les différentes faces.

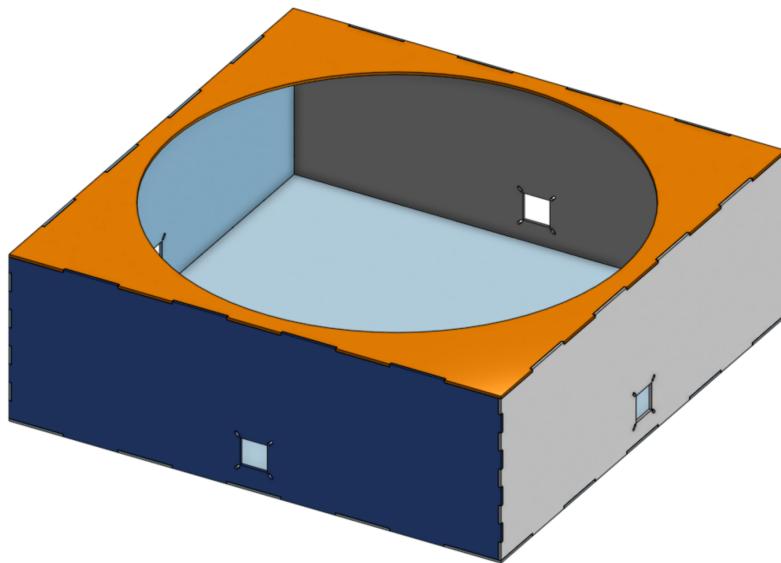


FIGURE 9 – Structure du système

L'ensemble de cette structure a été pensé pour pouvoir être facilement reproduisible avec le matériel du FabLab de l'école. L'idée est que chacune des parties soit une planche de bois découpée à la découpeuse laser.

Au cours de ce projet, j'ai donc produit une version de cette structure. L'ensemble des découpes ont été effectué comme prévu lors de la modélisation, à l'exception de la plaque de fond et de l'anneau supérieur qui ont été réalisés avec deux planches de bois pour chaque. Les liaisons entre les pièces ont été consolidées à l'aide d'une colle à bois.



FIGURE 10 – Réalisation de la structure au Fablab

Des tasseaux triangulaires ont par la suite été ajoutés, permettant ainsi de positionner un ruban LED orienté vers le cœur de la structure et d'éclairer ce dernier.

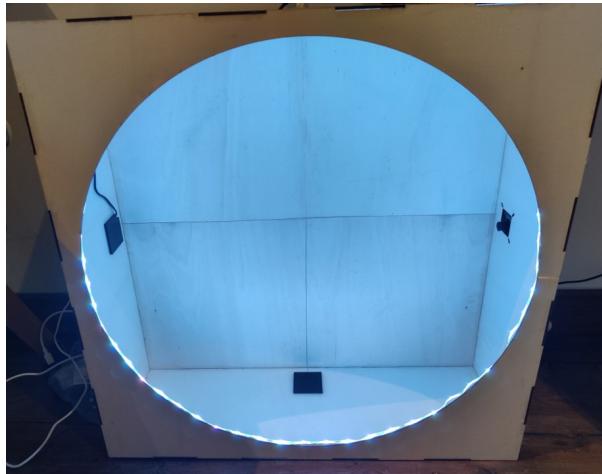


FIGURE 11 – Positionnement du ruban LED sur le support

3 Détection de la position de la fléchette

Le module de détection de la position de la fléchette représente la particularité principale de ce projet. C'est pourquoi je souhaitais dans cette partie présenter plus en détail le fonctionnement de cette estimation. Pour cela, j'expliquerai dans un premier temps les principes mathématiques théoriques utilisés et dans un second temps je détaillerai l'implémentation qui a été faite pour ce projet, en spécifiant les méthodes utilisées pour améliorer les qualités de résultats.

3.1 Principe de triangulation

La méthode qui a été utilisée pour déterminer la position de la fléchette dans l'espace est la triangulation. L'idée de base est de déterminer la position d'un point P dans l'espace à partir de plusieurs vues prises par des caméras placées à des positions différentes. Chaque caméra va donc capturer l'image du même point P , mais sous un angle différent. En connaissant les positions relatives des caméras ainsi que la position du point P sur chacune des images, il est possible de retrouver la position de ce point dans l'espace.

En notant u_i et v_i les coordonnées du point dans l'image i on peut déterminer la position relative $X_{rel}, Y_{rel}, Z_{rel}$ de P avec :

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = A_i \begin{bmatrix} X_{rel} \\ Y_{rel} \\ Z_{rel} \\ 1 \end{bmatrix}$$

avec A_i la matrice de projection de la caméra i . Cette matrice est définie suivant l'équation :

$$A_i = K_i [R_i \ t_i]$$

où K_i est la matrice intrinsèque de la caméra i et $[R_i \ t_i]$ sa position relative (R_i : matrice de rotation, t_i vecteur de translation).

La matrice intrinséque d'une caméra est définie en fonction de ses paramètres comme la focale, les distorsions optiques et le point principal. Elle s'écrit :

$$K_i = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

où f_x et f_y sont les facteurs de mise à l'échelle selon les axes x et y , et c_x et c_y sont les coordonnées du centre de l'image.

Pour trouver la position relative du point P en 3D, il suffit de résoudre ce système d'équations. Dans le cadre de ce projet, on a travaillé avec deux caméras en prenant la première comme référence pour la triangulation. Dans notre cas, ce système d'équation s'écrit donc :

$$\begin{cases} \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = K_1 \begin{bmatrix} I_3 & 0_{\mathbb{R}^3} \end{bmatrix} \begin{bmatrix} X_{rel} \\ Y_{rel} \\ Z_{rel} \\ 1 \end{bmatrix} \\ \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} = K_2 \begin{bmatrix} R_2 & t_2 \end{bmatrix} \begin{bmatrix} X_{rel} \\ Y_{rel} \\ Z_{rel} \\ 1 \end{bmatrix} \end{cases}$$

Les coordonnées du point P sont ainsi calculées dans le référentiel de la caméra 1. Pour les repasser dans un autre (comme par exemple celui du centre de la cible) on applique les translations et rotations nécessaires. Enfin, il est important de noter que les points (u_i, v_i) sont donnés en prenant en compte de la distorsion de la caméra. Si on souhaite travailler avec des coordonnées en pixels (que l'on notera (u_{pi}, v_{pi})), il faudra effectuer une transformation à l'aide des coefficients de distorsion radiale et tangentielle de la caméra. En notant D_i l'ensemble des coefficients de distorsion de la caméra i on peut écrire :

$$(u_i, v_i) = F(K_i, D_i, (u_{pi}, v_{pi}))$$

avec F la fonction qui gère cette distorsion, qui ne sera pas présentée dans ce rapport mais qui est intégrée dans n'importe quelle bibliothèque de traitement d'images (pour OpenCV il s'agit de *undistortPoints*).

3.2 Solution implémentée

L'explication précédente met en lumière les deux parties qui permettent le fonctionnement de cet estimateur de position. La première est relative à la définition des matrices et coefficients des caméras : K_i, R_i, t_i et D_i . La seconde concerne la détermination des pixels que l'on souhaite trianguler $((u_{pi}, v_{pi}))$.

3.2.1 Calibration des caméras

Pour cette partie, j'ai majoritairement suivi l'excellent tutoriel de Monsieur TEMUGE BATPUREV (github.com/TemugeB/python_stereo_camera_calibrate). Je vous recom-

mande de lire ces explications si vous souhaitez plus de détails.

Commençons par expliquer comment déterminer les matrices intrinsèques des caméras (K_i et D_i). Certains constructeurs donnent directement les valeurs de ces dernières mais pour la grande majorité des cas (y compris dans le cadre de ce projet) il faut pouvoir les re-calculer. L'idée repose sur l'acquisition d'images d'un motif d'échiquier aux dimensions connues, prises sous différents angles et positions. On utilise ensuite un algorithme qui détecte les coins des cases noires et blanches, qui servent de points de référence pour établir la relation entre les coordonnées des points dans l'image et leur position réelle dans l'espace. En résolvant ce problème par des méthodes d'optimisation, on obtient les matrices K_i et D_i .

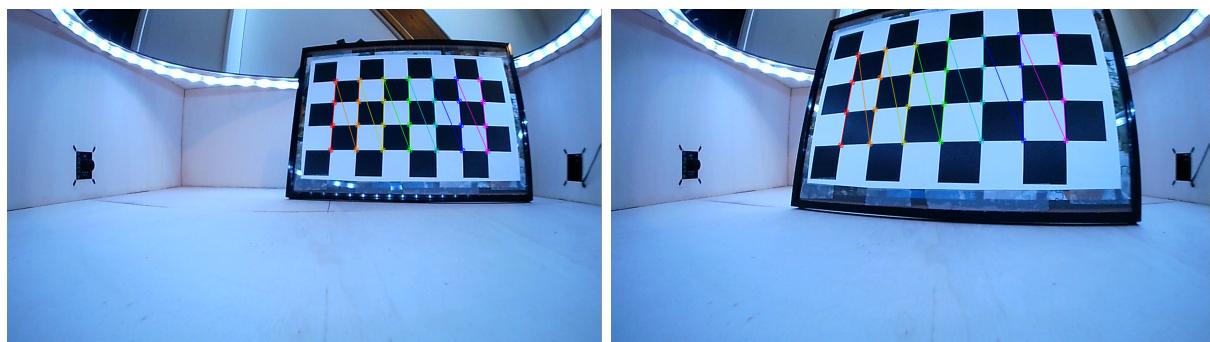


FIGURE 12 – Exemple de détection de l'échiquier pour la calibration de la caméra

Pour déterminer les matrices de rotations R_i et les vecteurs de translation t_i on utilise une méthode similaire. En utilisant un motif d'échiquier, et en capturant simultanément des images pour les différentes caméras, on peut déterminer les correspondances entre les points détectés dans les images permettant ainsi d'établir une relation géométrique entre les caméras.

3.2.2 La détermination des pixels d'intérêts

Pour l'estimation de la position d'une fléchette, le pixel d'intérêt est celui représentant l'impact de la fléchette sur la cible. En effet, en connaissant ce point et en triangulant sa position selon les principes expliqués précédemment, il est possible de connaître ses coordonnées dans le repère lié à la cible. Il est ensuite assez direct de déduire le score associé.

J'ai mis en place un algorithme permettant de maximiser la précision et la robustesse de la détection de ce point.

La première étape repose sur l'étude de deux images : une avant l'arrivée de la fléchette, et une autre après (voir figure 13). Ces dernières sont obtenues à l'aide des gestionnaires de fléchette et de caméra présentés précédemment.

En comparant ces images pixels par pixels on en déduit une image binaire représentant la différence entre ces dernières. Si d'un point de vue théorique la seule différence est l'arrivée de la fléchette, dans la pratique des différences de luminosité (même mineures) sont détectées (voir figure 14).

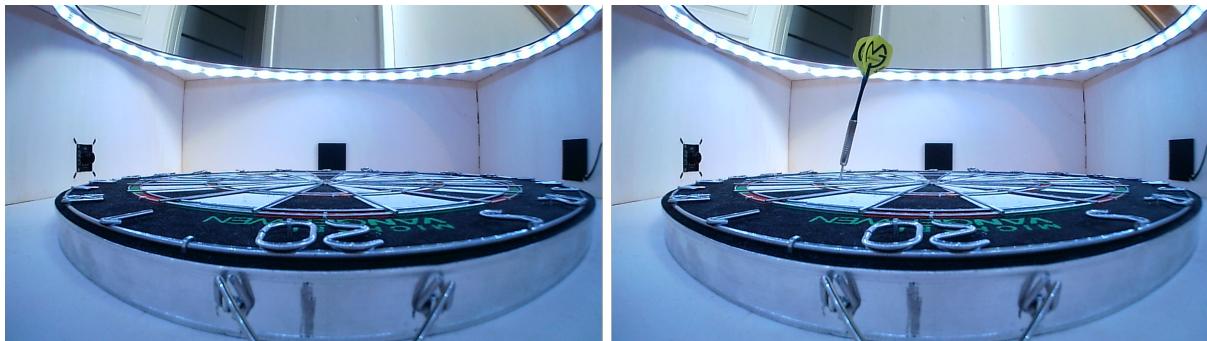


FIGURE 13 – Images avant et après l'arrivée d'une fléchette

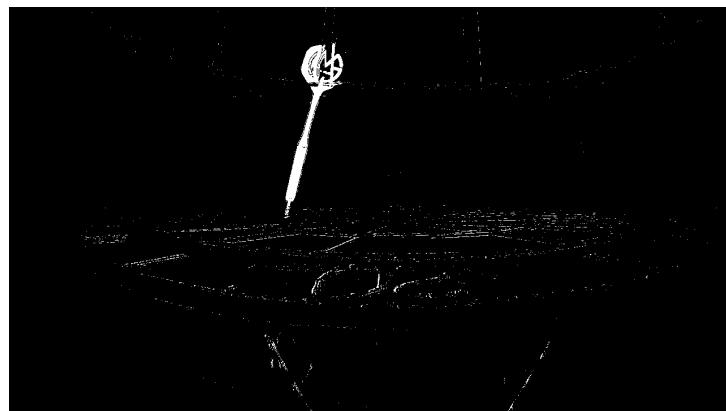


FIGURE 14 – Image binaire représentant la différence entre les images de la figure 13

La prochaine étape consiste donc en un moyen de filtrer ce bruit. Le principe repose sur le constat que la quasi-totalité de ces points parasites se situent dans le tiers haut et bas de l'image binaire. En effet, le tier haut représente l'extérieur de la structure qui est naturellement le plus sujet à des variations de luminosité, et le tier bas y est également assez sensible à cause des tiges métalliques de la cible (qui délimitent les zones des différents scores) qui peuvent aussi réfléchir cette lumière. Le tier central, lui, comprend quasiment que les points du corps de la fléchette. L'idée consiste donc à travailler avec cette partie centrale.

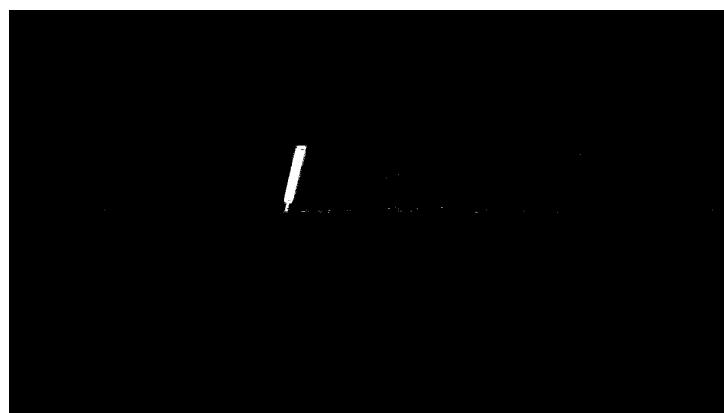


FIGURE 15 – Extraction du centre de l'image binaire

Une fois l'extraction de cette partie effectuée, avec un algorithme de détection de contours, on établit l'équation de la droite traversant le milieu du corps de fléchette (qui est le plus grand des contours détectés).

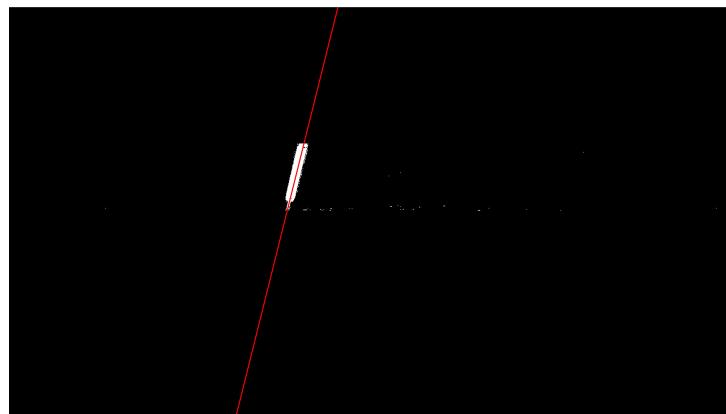


FIGURE 16 – Détermination de la droite traversant le milieu du corps de la fléchette

On utilise ensuite cette équation de droite pour effectuer un filtrage sur l'image binaire originelle (non tronquée du tiers haut et bas). Si les points sont trop éloignés de cette droite alors ils sont filtrés.

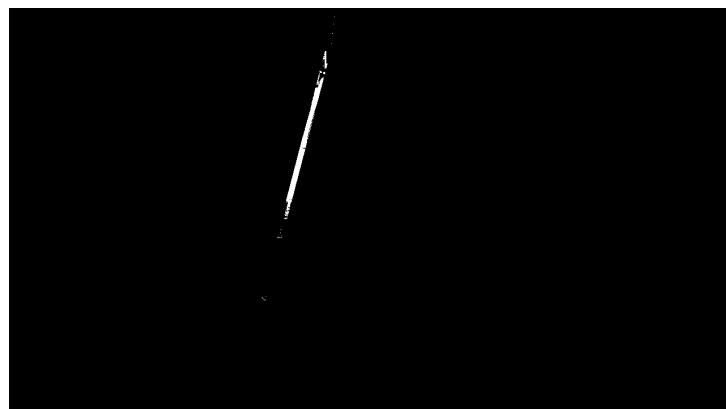


FIGURE 17 – Filtrage suivant l'équation de la droite

Le filtrage est quasiment terminé. Le dernier problème réside dans les points parasites qui peuvent subsister le long de cette droite (voir figure 17). Pour les éliminer, on applique un algorithme de filtrage qui permet de conserver les régions suffisamment grandes et de supprimer les petits bruits (voir figure 18).

Le pixel représentant l'impact de la fléchette sur la cible est le point le plus bas de cette image filtrée (voir figure 19).

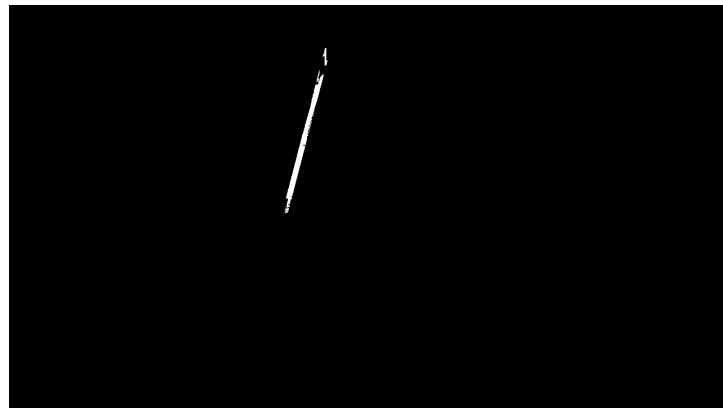


FIGURE 18 – Filtrage par nombre d’éléments connectés



FIGURE 19 – Détection finale du point d’impact (en vert) dans l’image originale

4 Budget

Comme précisé plus tôt dans ce rapport, le coût de reproduction du projet était un élément important du cahier des charges. La figure 20 présente donc ce dernier estimé pour reproduire le système.

	Prix unitaire (€)	Quantité	Total
Caméra	26,4	2	52,8
Raspberry	54,9	1	54,9
Ruban LED	20	1	20
Bois (250*122cm)	32,9	0,5	16,45
Colle/ruban collant	8,99	1	8,99
Total			153,14 €

FIGURE 20 – Estimation du coût du système

Cette estimation confirme bien le fait que ce système peut être reproduit à un prix abordable (on est proche de l’exigence des 150€). De plus, si on considère que ce projet est reproduit dans le cadre de l’association du personnel de l’école, alors on pourrait imaginer que certains coûts (pour la colle et le bois notamment) soient factorisés ce qui réduirait encore plus le prix individuel de chaque reproduction.

5 Gestion de projet

Je souhaitais maintenant faire un rapide point sur la partie gestion du projet.

5.1 Réunions

Au cours de ce projet trois réunions ont été réalisées avec Monsieur BOURDEAU-D'HUY. Une première pour lancer le projet et déterminer les différentes exigences de ce dernier. Une deuxième à environ mi-parcours pour discuter de l'avancée et identifier les pistes à privilégier. Enfin une dernière en fin de projet, pour identifier les dernières étapes sur lesquelles se focaliser. Au cours de chacune de ces réunions, Monsieur BOURDEAU-D'HUY a pu me partager son expertise et sa vision sur le projet ce qui m'a permis de prendre des décisions plus éclairées.

A noter également qu'une réunion a aussi été réalisée avec Monsieur RIADH BRAHAM (responsable de filière). Cette dernière m'a permis d'avoir un regard extérieur sur le projet ainsi que des retours sur mon organisation.

5.2 Outils utilisés

Au cours de ce projet, j'ai principalement utilisé deux outils pour organiser mon travail et suivre ma progression. Tout d'abord, un fichier de timelog, qui m'a permis d'enregistrer le temps consacré à chaque tâche et d'avoir une vision claire de l'avancement du projet. Couplé à un planning, cet outil m'a notamment aidé à mieux planifier les dernières étapes du projet.

Ensuite, GitHub a été essentiel pour la gestion du code en particulier en ce qui concerne son versioning et son partage. Cet outil me permet de rendre mon code accessible à tout le monde. Vous trouverez sur ce lien (github.com/ThibaudPiccinali/Dart-O-Matic) le dépôt GitHub du projet comprenant l'ensemble des codes. Les modélisations 3D ont été réalisé à l'aide du logiciel Onshape, vous pourrez accéder avec le lien suivant à l'ensemble de ces dernières (lien).

6 Ressentis

J'aimerais dans cette partie faire un point sur mon ressenti personnel de ce projet.

6.1 Ressentis vis à vis du travail proposé

Je suis fier du travail que j'ai accompli au cours de ce projet. Même si certains aspects pourraient être améliorés, je suis très satisfait des résultats obtenus. J'ai énormément appris en travaillant sur ce sujet, que ce soit sur le plan technique (serveur web, C++) mais aussi organisationnel. Chaque difficulté rencontrée a été une occasion d'approfondir mes connaissances et de trouver des solutions, ce qui a rendu l'expérience très enrichissante.

6.2 Ressentis vis à vis du projet dans son ensemble

J'ai vraiment adoré réaliser ce projet. L'un des aspects que j'ai le plus apprécié est la grande liberté d'organisation dont je disposais. Pouvoir gérer moi-même les différentes

étapes, expérimenter et choisir mes propres solutions m'a permis de m'investir pleinement et de travailler à mon rythme. Cette autonomie a rendu le projet encore plus intéressant et motivant, et je pense que je garderai une expérience très positive de cette aventure.

7 Conclusion

Je souhaitais pour conclure ce rapport parler des limites et perspectives d'évolution du projet dans son état actuel.

Pour moi la limite majeure est la précision de la détection de la position de la fléchette. Même si cette dernière a grandement été améliorée au cours des différentes versions du projet, elle peut rester insuffisante pour jouer sans perturbations et sans jamais devoir manuellement modifier les scores au cours d'une partie. Également une limite de l'expérience de jeu est le fait qu'il y a un petit temps d'attente obligatoire entre les lancers.

Les améliorations potentielles seraient donc d'améliorer la qualité de détection en ajoutant, par exemple, d'autres caméras pour ainsi faire une triangulation avec plus de point de vue. Utiliser des algorithmes d'intelligence artificielle pourrait également être utile pour avoir une meilleure détection du point d'impact de la fléchette sur la cible.