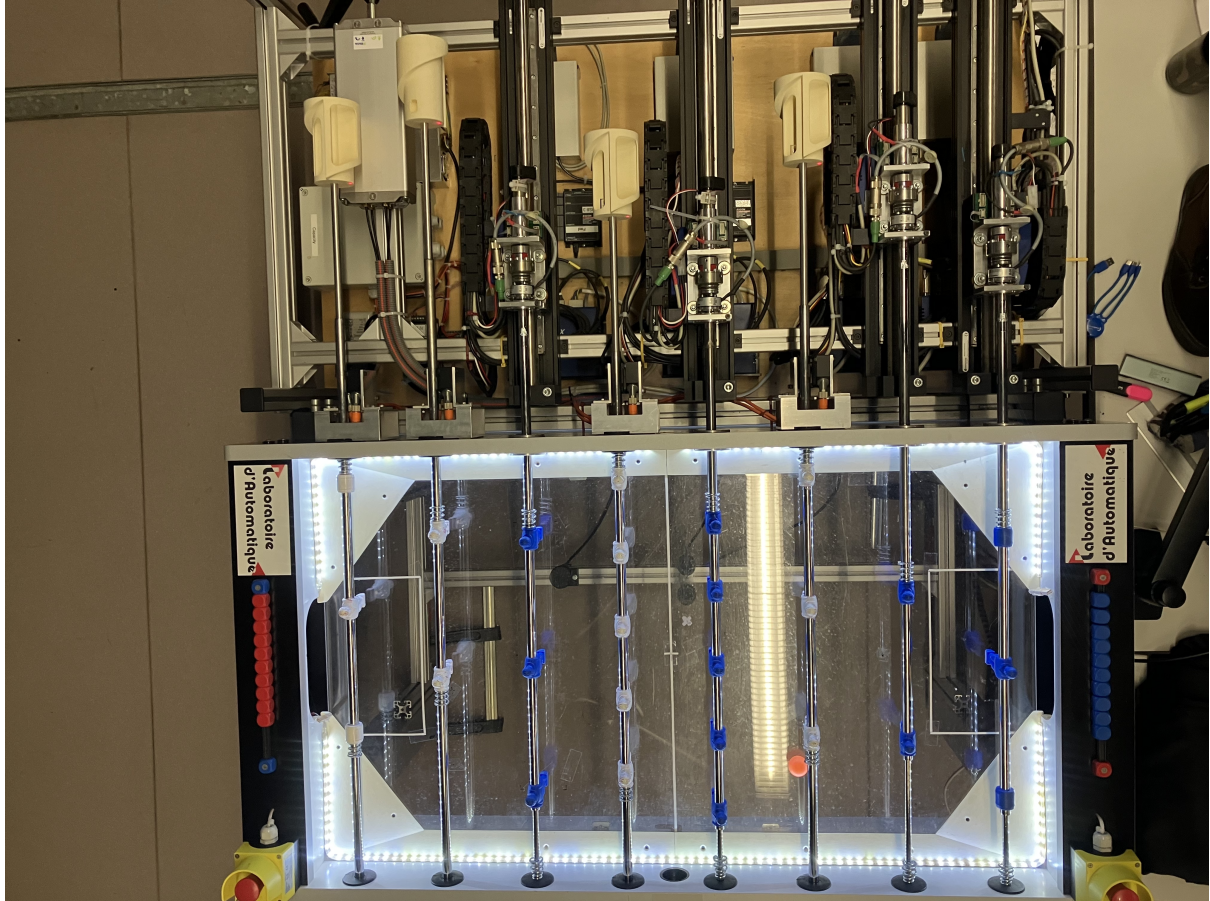


Rapport de projet de semestre



Babyfoot EPFL: Visualisation, jonglage et angle de contrôle

Michaël FARINELLI

28 janvier 2026

Résumé

Ce rapport présente le travail réalisé durant le semestre d'automne 2025 dans le cadre d'un projet de semestre de master portant sur le baby-foot automatisé de l'EPFL. Ce système a atteint un niveau de complexité rendant difficile la compréhension immédiate de son fonctionnement interne et de ses stratégies de contrôle. Le projet s'articule autour de trois contributions principales. La première concerne le développement d'un outil de visualisation en temps réel permettant d'afficher l'état du système de contrôle ainsi que les stratégies actives et les conditions associées. Cet outil améliore significativement la lisibilité du code LabVIEW existant et a permis d'analyser en profondeur le sélecteur de stratégies, conduisant à de légères optimisations et à une augmentation du nombre de jonglages observés.

La seconde contribution porte sur la détermination expérimentale d'un angle d'interception optimal pour l'arrêt de la balle, en fonction de la vitesse et de l'angle de tir. Une méthodologie de tests systématiques a été mise en œuvre sur le système réel, complétée par une analyse des données sous Matlab. Les résultats ont conduit à l'identification d'un angle optimal de $\phi_{opt} = 144^\circ$ pour les conditions étudiées, offrant un compromis fiable entre amorti, rebond contrôlé et prévention du passage de la balle sous le joueur.

Enfin, la stratégie de jonglage a été analysée et plusieurs limitations ont été mises en évidence, notamment des fenêtres de contrôle trop restrictives et une forte sensibilité à l'inertie des moteurs. Différentes pistes d'amélioration ont été explorées, mettant en lumière la complexité inhérente à cette stratégie.

Dans l'ensemble, ce travail fournit des outils pratiques, des résultats expérimentaux et une documentation des limitations actuelles, dans le but de faciliter la poursuite et l'amélioration future du baby-foot automatisé de l'EPFL.

Table des matières

1	Introduction	4
1.1	Remerciements	5
2	Visualisation	5
3	Angle d'interception et analyse angle de tir	7
3.1	Méthode	7
3.2	Résultats	8
4	Jonglage	9
5	Bug solving	10
6	Conclusion	11

1 Introduction

Le projet d'automatisation d'un baby-foot est développé au sein du laboratoire d'automatique de l'EPFL depuis 2012. Chaque année, il est enrichi et amélioré par des étudiants dans le cadre de projets de fin de bachelor ou de master. Ma contribution s'inscrit dans cette continuité et a été réalisée durant le semestre d'automne 2025, dans le cadre de mon projet de semestre de master.

Le baby-foot automatisé repose sur plusieurs composants principaux, dont la liste suivante, non exhaustive, présente les éléments essentiels :

- Deux caméras placées sous la surface de jeu en plexiglas transparent, permettant de détecter et de localiser la balle en temps réel.
- Un ruban de LED disposé le long du périmètre interne du terrain, assurant un éclairage homogène de la balle.
- Des moteurs dédiés à la translation et à la rotation des joueurs bleus.
- Des capteurs fournissant au logiciel les informations relatives à la position et à la rotation des joueurs blancs, manipulés par des joueurs humains.

L'ensemble du programme assurant le contrôle du baby-foot est développé sous LabVIEW.

Ayant été développé sur plusieurs années et par différentes équipes, une difficulté à comprendre exactement le code s'est faite ressentir. C'est donc pour ça que des objectifs concernant la visualisation en temps réel de l'état du système de contrôle ont été définis. Les objectifs du projet ont donc été les suivants :

1. Créer un système permettant de visualiser en direct les changements de stratégie, avec les différentes conditions devant être remplies.
2. Définir clairement l'angle optimal pour arrêter la balle, en fonction de la vitesse et de l'angle de tir.
3. Optimiser la stratégie de jonglage.

Les définitions suivantes sont utilisées dans le rapport :

- ϕ défini l'angle des joueurs. 0 correspond à un joueur les pieds en haut, le sens positif est le même que pour un tir vers l'avant.

— θ angle de tir. 0 correspond à un tir droit.

1.1 Remerciements

Merci au professeur C. Salzmann de l'opportunité de travailler sur ce projet. Et surtout de son soutien et ses conseils tout au long du projet.

2 Visualisation

Une des grandes difficultés actuelle du Babyfoot est de comprendre où le programme en est. Or, c'est absolument nécessaire afin de pouvoir améliorer les stratégies ou les conditions pour être dans une stratégie. Le subVI : Visual_FARI, a été créé pour pallier cette difficulté. Il prend en entrée la position et vitesse de la balle, la limite admissible en Vx pour jongler ainsi que l'état de la stratégie et sort un tableau qu'il est possible de brancher directement pour avoir un affichage graphique. Bien que créé et implémenté dans le case "Fall_2025_complete_light" il est possible, en comptant quelques modifications mineures, de l'utiliser pour analyser d'autres stratégies.

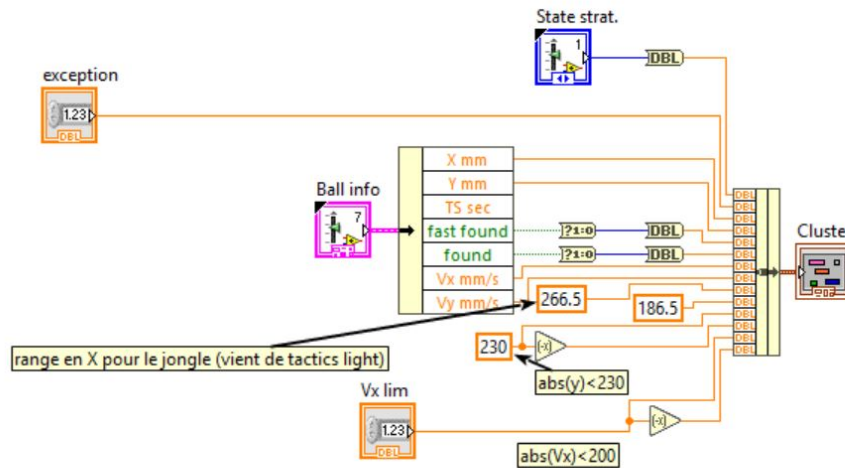


FIGURE 1 – Vi permettant la visualisation de l'état en temps réel

En utilisant le subVI, il a été possible de comprendre en profondeur le fonctionnement

du subVI `tactics.light_selector` et de l'optimiser légèrement. Grâce à ces modifications, beaucoup plus de jongles sont observables.

Dans Figure 2, l'utilité du VI est illustrée. En effet, le jonglage est clairement visible. De plus, les conditions sont aussi bien visible.

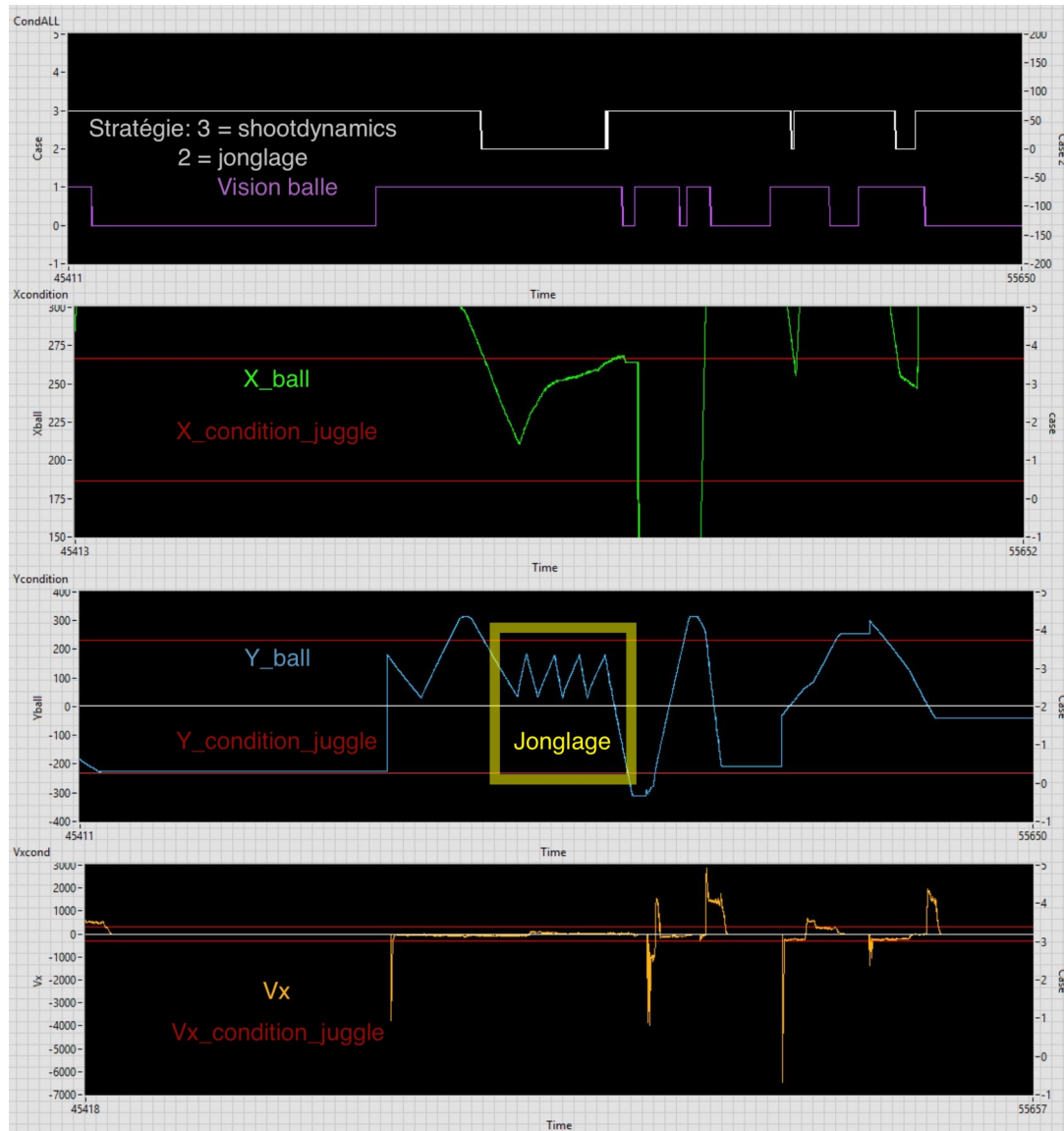


FIGURE 2 – Front panel de Visual, avec partie en cours

Le subVI pourrait être amélioré afin d'être rendu plus modulaire. En l'état, il prend toutes les données et ressort toutes les données. C'est ensuite à l'utilisateur de cocher les données qu'il veut voir affichées sur les graphiques ou non. De plus, si les valeurs limites des conditions changent, il est nécessaire de les modifier aussi dans le subVI. La création d'un tableau avec toutes ces constantes faciliterait grandement l'utilisation du programme.

3 Angle d'interception et analyse angle de tir

Il est trivial de montrer que pour un angle ϕ proche de 180° un rebond est assuré, et pour amortir il faut diminuer l'angle. De plus, une vitesse élevée exigera une moins grande diminution de l'angle, car la balle aura plus de force pour soulever le joueur. Considérant qu'un passage au travers - la balle à assez de force pour soulever le joueur et passer en dessous - est inacceptable, mais qu'un rebond l'est, il en découle qu'il faut définir un angle d'interception pour un tir rapide. C'est donc ce qui a été fait et les résultats se trouvent dans la sous-section suivante : 3.1.

3.1 Méthode

La méthodologie suivante a été utilisée afin d'obtenir un angle d'interception optimal : ϕ_{opt} .

1. Tir droit, vitesse élevée, ϕ variable. (Table 1)
2. Tir droit, vitesse variable, ϕ variable. (Table 2)
3. Tir de test. Variation de θ , de la vitesse de tir, et de la fenêtre visée.

Dans cette section, tous les tirs ont été effectués depuis les défenseurs bleus. Les mots-clés suivants ont été utilisés pour quantifier la qualité du contrôle de balle par les attaquants blancs : "rebond" signifie que la balle revient en direction des défenseurs bleu ; "amorti" veut dire que la balle rebondi, mais reste dans le rayon d'action des attaquants ; "traversé" que la balle a pu forcer le passage sous le joueur et continuer sa trajectoire ; et finalement "arrêt" est utilisé pour les cas où la balle est écrasée par le joueur. De plus, partant de l'hypothèse que les amateurs seraient les utilisateurs les plus courants, les tirs effectués ont eu pour but de ressembler à ce qu'on pourrait trouver dans une partie. C'est-à-dire, pas de "Snake" ou autres techniques avancées pour tirer très fort. Partant de ce principe, les tirs les plus forts étaient aux alentours de 4000mm/s, contre au moins 8000mm/s pour un "Snake".

Concernant l'implémentation labVIEW : une première version du case se basait sur "man in the mirror", un ancien case dans lequel le robot miroitait exactement les joueurs humains. Après l'avoir légèrement modifié pour qu'il suive la balle et non pas les joueurs, les deux premières étapes de l'analyse ont pu être réalisées. Le problème qui est survenu

était que seul le joueur central suivait la balle, donc pour des angles élevés cette stratégie s'est révélée inefficace. Pour pallier ce problème, la deuxième version s'est basée sur "DEF ALL PLAYER". Contrairement au précédent, celui-ci permet de suivre la balle avec tous les joueurs. De plus, un subVI permettant d'enregistrer la valeur maximale de différentes variables et de les afficher pendant que le programme tourne a été ajouté. Ce dernier permet de faire plusieurs tirs sans devoir réinitialiser tout le programme.

3.2 Résultats

Pour ce premier test le but était de déterminer un premier intervalle de valeur pour ϕ . Les résultats se trouvent dans Table 1), il en sort qu'un angle $\phi \in [142, 144]$ semble optimal.

ϕ°	150	145	144	143	142	140
Contrôle ?	rebond	rebond	rebond et amorti	amorti et rebond	traversé et amorti	traversé

TABLE 1 – Tir droit : $\theta < 6^\circ$, vitesse élevée : $V_{moy} = 3953 \text{ mm/s}$

Pour affiner la recherche, un nouveau test a été effectué, cette fois avec des tirs à vitesse variable, mais avec encore une déviation maximale de 6° . Les résultats se trouvent dans Table 2.

$\phi[^\circ]$	145	144	143	142
Limite inférieure vitesse [mm/s]	694	918	348	1144
Limite supérieure vitesse [mm/s]	4020	2459	3432	4148
Contrôle ?	rebond et amorti	amorti	amorti et traversé	arrêt et traversé

TABLE 2 – Tir droit : $\theta < 6^\circ$, vitesse variable

L'angle pour lequel la balle n'a jamais traversé est $\phi = 144^\circ$. C'est donc ce dernier qui sera défini comme angle d'interception optimal : $\phi_{opt} = 144^\circ$. Il est important de noter que cet angle est aussi fonction du diamètre, du poids, et de la matière de la balle, donc si un changement de balle à lieu, il serait bon de refaire une sélection de ϕ_{opt} .

Une représentation en nuage de points des différentes vitesses pour un angle $\phi = 144^\circ$ ce trouve dans Figure 3. Il en découle qu'à partir d'une vitesse horizontale $V_y \approx 300 \text{ mm/s}$

le robot à des difficultés à toucher la balle et qu'à partir d'une vitesse verticale de $V_y \approx 3800 \text{ mm/s}$ il y a des rebonds. Il est important de noter que les rebonds n'avaient pas tous la même intensité. Pour les rebonds faibles, un contrôle serait possible en faisant faire un demi-tour au joueur par exemple.

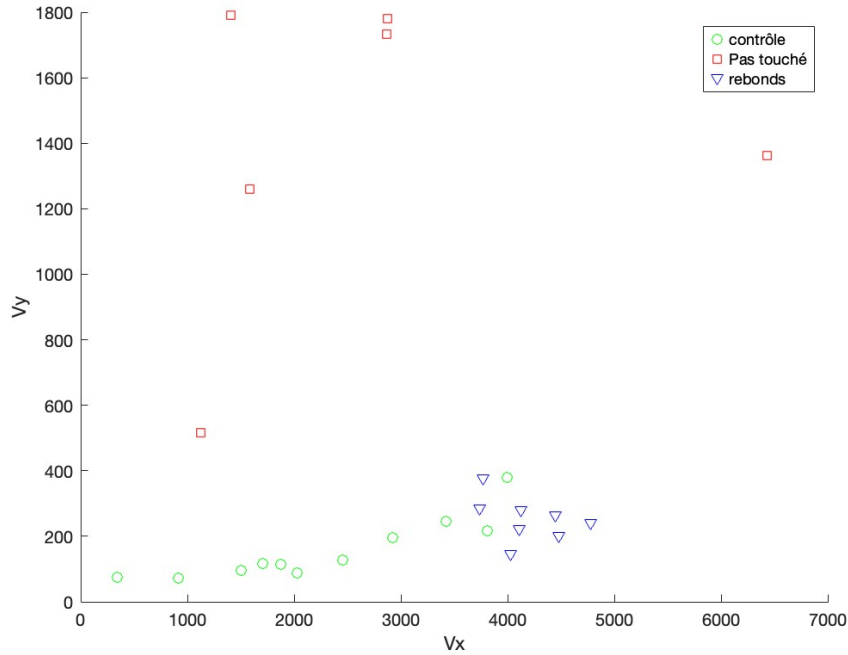


FIGURE 3 – Etat de contrôle en fonction de V_x et V_y , pour $\phi = 144^\circ$

Ces vitesses pourront être utilisées comme condition pour choisir si l'on souhaite jongler, tirer, ou amortir.

4 Jonglage

Le jonglage en jeu avait de grande faiblesse. Notamment une fenêtre de mise en action réduite (Figure 4), une condition sur la vitesse horizontale rendant impossible de commencer à jongler depuis l'arrêt, et une vitesse verticale très lente - une balle simplement déposée du haut du corner allait trop vite pour jongler avec les attaquants - ce qui ne correspond pas à un joueur humain. De plus, le jonglage ne se fait qu'autour d'une position $y=0$ pour les attaquants.

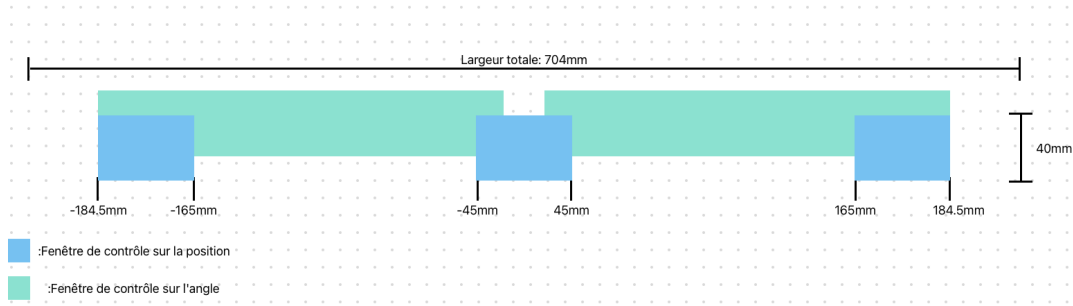


FIGURE 4 – Fenêtre de condition pour se mettre à jongler

Le principal but était de pouvoir augmenter la fenêtre de contrôle sur la position, afin que toutes les balles assez lentes puissent être jonglées. Dans ce but, le case "Fall.2025_juggling_test" a été créé. C'est un duplicata de celui du printemps, où la fenêtre de jonglage a été déplacée. Le but était, à terme, de pouvoir mettre en entrée la position des joueurs à l'itération précédente pour avoir une fenêtre de jonglage dynamique. Cependant, à cause de l'inertie des moteurs qui n'est pas aisée à prendre en compte, aucun résultat probant n'a été atteint. Le deuxième test fut de mettre le tout dans une séquence, afin de d'abord mettre les joueurs proches de la balle, et ensuite se mettre à jongler. Cette fois, le problème était que la balle ressortait de la fenêtre de contrôle (Figure 4), elle rentrait dans celle de mise en place et les joueurs se rapprochaient à nouveau de la balle. Ce fonctionnement faisait que les joueurs frappaient la balle beaucoup trop fort et perdaient le contrôle instantanément. Une autre idée qui pourrait fonctionner, serait de créer un case avant de rentrer dans le case de jonglage pour aller chercher la balle.

En conclusion, la stratégie de jonglage s'est révélée difficile à améliorer due à la complexité du système.

5 Bug solving

Un couloir défini par $|y| \in [90, 117]$ où l'attaquant robotisé ne va pas a été découvert. C'est-à-dire que si $|Bally| \in [90, 117]$, le robot se met en position neutre ($y = 0$) (c.f. Figure 5. Après une analyse du VI de contrôle des avants : Y forward.vi, la cause de ce couloir a pu être découverte : une simple erreur lorsque le "dimension" s'est fait *unbundle*. Y minimum du midfield était en entrée du vi pour les forwards. De plus, tous les VI gérant la position Y ont été modifiés afin d'avoir une meilleure gestion des bords.

Avant, lorsque la balle était perdue sur un côté, les joueurs se mettaient en position centrée. Maintenant, ils restent du côté où la balle a été perdue.

6 Conclusion

La prise en main ayant pris beaucoup de temps, tous les objectifs du projet n'ont pu être validés. En particulier le jonglage n'a connu aucune amélioration majeure.

Pour un prochain projet, je me concentrerai donc sur ce sujet, en utilisant notamment le rapport de printemps 2025, et le case "Spring_2025_juggling".

Ce projet a été très enrichissant et m'a permis d'acquérir de solides connaissances sur labVIEW. L'ajout du système de visualisation est une vraie amélioration et l'angle d'interception, une fois intégré aux stratégies, pourra être très utile.

ANNEXE

Visualisation du couloir cité dans section 5. Seule une moitié du terrain est représentée en raison de la symétrie.

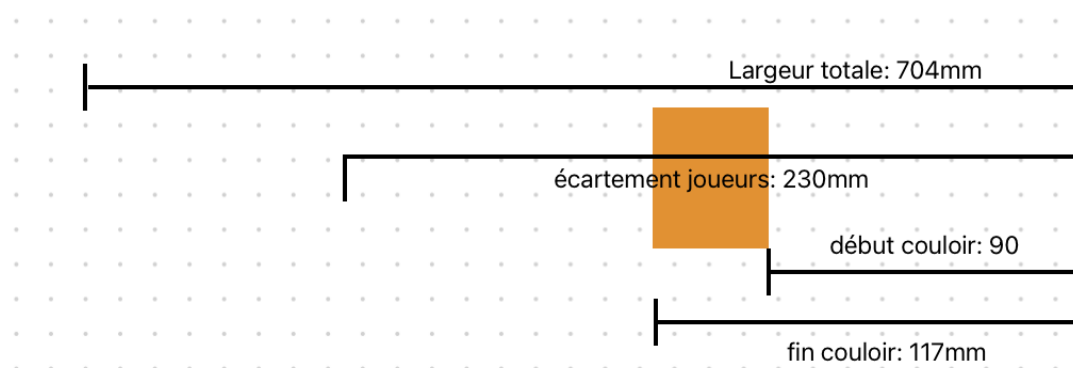


FIGURE 5 – Couloir pour lequel les joueurs vont en position neutre ($y=0$)

Explication front pannel

Tout les ajouts de ce rapport sont regroupés en haut à droite du front pannel du vi principal dans un cadre, comme visible dans Figure 6. A l'initialisation le premier graphique représente la stratégie sur l'axe de gauche ainsi que la vision ou non de la balle. Il est possible d'ajouter d'autres fonction en cochant sur le sélecteur situé sur la droite. Les trois autres graphiques représentent chacun une variable ainsi que les conditions pour changer de stratégie sur l'axe de gauche, et sur l'axe de droite l'état de la stratégie. En bas à droite se trouve la partie front du case "V_intercept".

Utilisation Visual et autres SubVIs

Pour utiliser le VI "visual_FARI", le plus simple est de faire un clic-droit dessus, et de "replace with SubVI contents" de cette manière il sera plus facile de modifier les *input* et les *output*. Sinon toutes les instances du VI seront modifiées. Ensuite vous pourrez créer un nouveau SubVI en sélectionnant ce dont vous avez besoin, et dans le menu "Edit", cliquez sur *Create SubVI*.

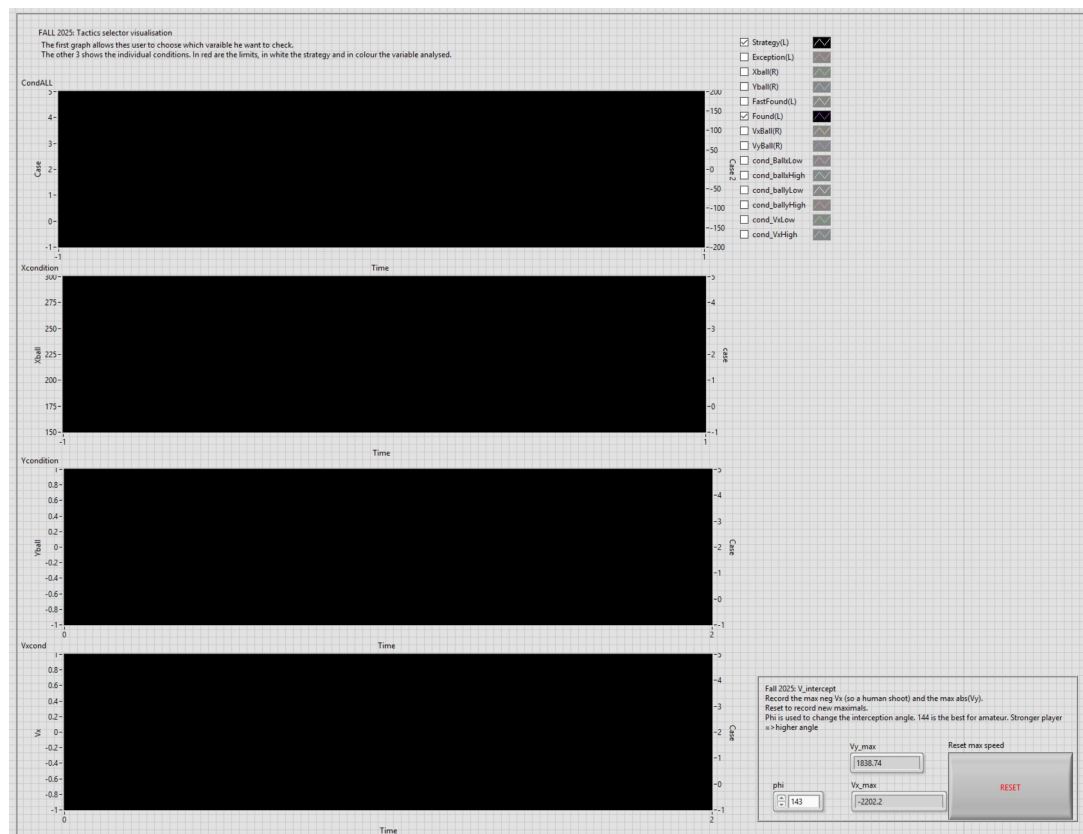


FIGURE 6 – Front pannel

Programme matlab utilisé pour analyser la mémoire

```
clear; clc; close all;
```

```
%% Data import
```

```
M = readmatrix('4tirs.txt'); % matrice des donn es saved
                                par memory du babyfoot
```

```
% 1 = temps; 2 BAllx; 3 Bally
% 8 = Vx; 9 = Vy; 16 = PhiATK
%
%
%
%
%
```

```

%
%
%
%
%
%
%

%% Constant definition
t = linspace(1,40,20000);           % time matrix
x = M(:,2);                         % position x of ball
y = M(:,3);
vx = M(:,8);
vy = M(:,9);
phi = M(:,16);                      % angle of Shooter

%% Juggle analysis forward
% constant
Xatk = 226.5;                       % x position of forward bar
    in mm
Vlimjuggle = 200;                   % As defined in Labview

% Conditions to be in Juggle case
condX = (x >= Xatk-40) & (x <= Xatk+40); % condition on x
    position of ball

% -Must be in
    range of
    forward-

condY = abs(y) < 230;               % -Defined the
    vertical edges of control surface-

```

```
condVx = abs(vx) < Vlimjuggle;           % Must be slow
      enough for control

condAll = condX & condY & condVx;         % to enter juggle
      , condAll must be true

% visualize conditions

tiledlayout(4,1);                         % first condition
nexttile
hold on
plot(t(~condX), x(~condX), 'r. ');        % When its met in
      green
plot(t(condX), x(condX), 'g. ');          % otherwise in
      red
ylabel('X ball');
yline(Xatk-40);
yline(Xatk+40);
%ylim([Xatk-40-10,Xatk+40+10]);           % Can be used to
      focus the plot
                                           % on met area

nexttile; % Second subplot                % Second
      condition
hold on                                   % - same
      principle-
plot(t(~condY), y(~condY), 'r. ');
plot(t(condY), y(condY), 'g. ');
yline(-230);
yline(230);
```

```

ylabel('Y ball');
%ylim([-230-10,230+10])

nexttile; % Third subplot
hold on
plot(t(~condVx), vx(~condVx), 'r. ');
plot(t(condVx), vx(condVx), 'g. ');
yline(-Vlimjuggle);
yline(Vlimjuggle);
ylabel('Vx');
%ylim([-Vlimjuggle-10,Vlimjuggle+10])

nexttile; % Fourth subplot % Instant check
    for all conditions
yl = ylim;
for k = 1:length(t)-1
    if condAll(k)
        patch([t(k) t(k+1) t(k+1) t(k)], ...
                [yl(1) yl(1) yl(2) yl(2)], ...
                'g', 'FaceAlpha',1, 'EdgeColor','none');
    else % Comment the
        else statement to get faster results
        patch([t(k) t(k+1) t(k+1) t(k)], ...
                [yl(1) yl(1) yl(2) yl(2)], ...
                'r', 'FaceAlpha',0.1, 'EdgeColor','none');
    end
end
ylabel('All cond for juggle met');
xlabel('Time');

```

```
%% Best stop angle phi
% used for the analysis of shot between white DEFS and blue
  FORWARDS
% as this is the smallest distance, this gives an upper bound
  for the
% stop angle phi.

% constant
Xdef = 376.5; % As defined in Dimension
  .vi
OutRange = Xdef - 45; % Control range of DEF
delT = 5; % Time over which
  velocity is computed

% Time of shooting
% Defined as the speed when the ball goes out of the control
  range
% of the white DEF

idx_tshot = find(x(1:end-1) > OutRange & x(2:end) <= OutRange
  );

% Speed at shooting time

Vx_shot = (x(idx_tshot + delT) - x(idx_tshot))./(t(idx_tshot+
  delT)-t(idx_tshot));
Vx_shot = Vx_shot(:,1);
Vy_shot = (y(idx_tshot + delT) - y(idx_tshot))./(t(idx_tshot+
  delT)-t(idx_tshot));
Vy_shot = Vy_shot(:,1);
```

```

% Time of stopping
idx_tstop = zeros(1,length(idx_tshot));
for i = 1:length(idx_tshot)
    t_shot = t(idx_tshot(i))+0.5;

                                % Stop at least 0.5sec

    after shoot
    Var = find(sqrt(vx(t>t_shot).^2+vy(t>t_shot).^2) < 5);
        % Times where |v|<5mm/s

                                %

                                af

                                Ts

    idx_tstop(i) = Var(1)+idx_tshot(i);
end
% To get better result impose a condition of some dt were the
    ball must be
% static or with a slow velocity; Or check that the x
    position in within
% control range of a rod.

% Angle of player at time of stopping
phiStop = phi(idx_tstop);          % phi = 0 is defined as
    the XXX position

%%
figure()
hold on
yyaxis left

```



```
plot(t,x);                                % Positons in function of
    time
xline(t(idx_tstop))                       % Vertical line: Ball
    stopped
yline(OutOfRange);                       % Out of range of
    defenders
yyaxis right
plot(t,vx);                              % X velocity
scatter(t(idx_tshot),Vx_shot)            % Time and velocity of
    the shot
legend('','vertical: Ball stopped','horizontal: Out of range
    of DEF',Location='best')
hold off
```