



**PROJET D'INGENIEURIE SIMULTANÉE :
BABY-FOOT STRATÉGIE**

PAR :

ALEXANDRE DI PIAZZA

ET

MAXIME LUBRANO-LAVADERA

SUPERVISEUR :
CHRISTOPHE SALZMANN

Table des Matières :

1.	Introduction	1
2.	Élaboration de la Stratégie.....	2
2.1	Blocage par coup sec	2
2.2	Blocage par amortissement.....	2
2.2.1	Amortissement	2
2.2.2	Amortissement suivi	3
2.2.3	Amortissement Itératif (Méthode choisie)	3
2.2.4	Tableau récapitulatif	4
3.	Exécution de la Stratégie	5
3.1	Données utilisées et convention	5
3.1.1	VI Strategy.vi	5
3.1.2	Coordonnées de la position	7
3.1.3	Vitesse	7
3.1.4	Angles	8
3.2	Division du baby-foot en zones	8
3.3	Angles dans les zones d'anticipation.....	10
3.4	Angles dans les zones d'anticipation.....	11
3.4.1	Principe général	11
3.4.2	Mesures et Bornes	12
3.4.3	Hystérèse	13
3.5	Suivi de la balle	14
3.6	Schéma récapitulatif de la Stratégie	15
4.	Méthode prédictive de la position de la balle	16
4.1	Méthode existante et nouvelle méthode	16

4.2 Régression linéaire pour anticiper la trajectoire	16
4.3 Implémentation de la régression linéaire sur LabView	18
5. Performance du code LabView.....	19
5.1 Strength/Courant dans les moteurs à friction	19
5.2 Offset moteur.....	19
5.3 Amélioration.....	20
5.4 Analyse vidéo	20
6. Conclusion	21
7. Annexes.....	22

1. Introduction

Ce projet a eu pour but d'améliorer la stratégie du baby-foot automatique. Le projet du baby-foot automatique est né en 2012, et est amélioré chaque année au cours des différents projets de Bachelor et de Master.

Initialement, la stratégie était de tirer chaque fois que la balle était près des joueurs contrôlés. Le sujet de l'amélioration de la stratégie étant vaste, nous avons décidé de trouver un objectif précis. Les bons joueurs de baby-foot arrivent à contrôler la balle en l'arrêtant avec les joueurs, pour ensuite pouvoir enchaîner sur un tir précis. C'est la stratégie d'attaque principale entre joueurs humains. Nous avons donc essayé d'implémenter cette dernière, en arrêtant la balle en mouvement avec les attaquants.

Tout d'abord, il y a eu un temps de compréhension et d'annotation du code LabView déjà présent, puis nous avons testé différentes stratégies pour l'arrêt de la balle. Nous avons vite réalisé qu'il fallait être très précis pour réaliser un arrêt de la balle, notamment avec la matière lisse du terrain et de la balle. Nous avons donc réfléchi à l'algorithme qui nécessitait le moins de précision, afin qu'il puisse être adapté à une vraie partie. De plus, avec la stratégie déjà présente, les joueurs avaient beaucoup de difficulté à intercepter des tirs en diagonale, il était donc impossible de penser l'arrêter. Nous avons réfléchi à une méthode permettant de mieux intercepter les tirs diagonaux.

2. Élaboration de la Stratégie

Dans cette partie nous expliquons les différentes stratégies envisagées pour l'arrêt de la balle.

2.1 Blocage par coup sec

Au baby-foot plusieurs méthodes existent pour arrêter la balle ; la plus courante est de la bloquer en dessous du joueur. Le joueur attend que la balle se trouve juste en dessous du joueur avec lequel il veut l'arrêter, ce dernier étant levé à un certain angle pour permettre à la balle de passer en dessous. Quand la balle est juste en dessous du joueur choisi, il faut effectuer un mouvement sec et bloquer la balle.

Cette méthode d'immobilisation de la balle est le plus souvent utilisée quand la balle va à une vitesse assez faible. Il est aussi nécessaire que les frottements sur la balle soient suffisants pour qu'elle ne glisse pas. La balle du baby-foot utilisée étant relativement lisse et le terrain aussi, cela semble difficile. En comparant avec le deuxième baby-foot présent dans la salle avec celui utilisé, sa balle est beaucoup plus rugueuse et le terrain moins lisse : on bloque la balle avec beaucoup plus de facilité qu'avec le baby-foot automatisé.

De plus, la difficulté dans ce mouvement vient aussi du fait qu'il faut prédire où la balle sera à l'intersection, ainsi qu'enclencher le mouvement de blocage au bon moment. Après plusieurs essais manuels sur le baby-foot automatisé on a considéré que la marge de manœuvre pour bloquer directement la balle était trop réduite pour la poursuivre comme stratégie.

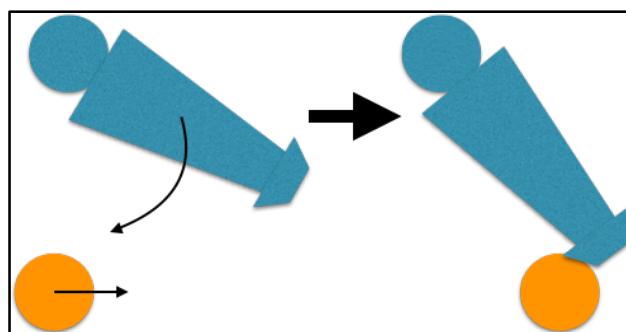


Figure 1 Principe de l'arrêt sec

2.2 Blocage par amortissement

Alternativement, au lieu de l'arrêter d'un seul coup on pourrait « contrôler » la balle et réduire sa vitesse pour ensuite effectuer un tir ou une passe.

2.2.1 Amortissement

Comme au football, on peut effectuer un contrôle et amortir la balle. Cela pourrait se faire de deux façons différentes. La première est la plus simple : positionner un joueur à un certain angle d'amortissement et le laisser se faire percuter par la balle. Ceci pourrait donc au mieux arrêter la balle ou au pire diminuer sa vitesse.

2.2.2 Amortissement suivi

La deuxième méthode est plus complexe, il faut suivre le mouvement de la balle avec le joueur du baby-foot pour l'accompagner dans son mouvement et l'amortir petit à petit jusqu'à l'arrêter. Comme nous avons une vision assez limitée avec une précision de 2mm par pixels, et que les valeurs de vitesses ne sont pas assez précises pour bien prédire le mouvement on choisit la première méthode mais avec une modification importante.

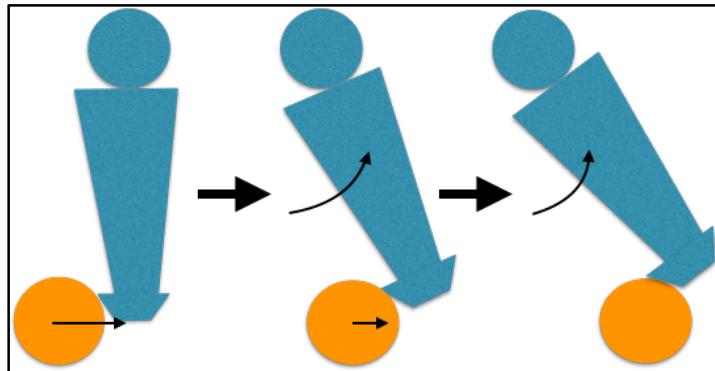


Figure 2 Amortissement suivi

2.2.3 Amortissement Itératif

Comme la première méthode était la plus simple et celle qui semblait avoir le plus de chance d'aboutir on l'a choisie. Cependant on l'a modifiée légèrement : on utilise la méthode pour amortir la balle de manière itérative. Ainsi, au début, la balle percute le joueur qui est à un angle optimal afin de l'amortir. Si la balle ne s'arrête pas après le premier amortissement, un second amortissement est mis en place de l'autre côté et ainsi de suite jusqu'à ce que la balle soit "immobile".

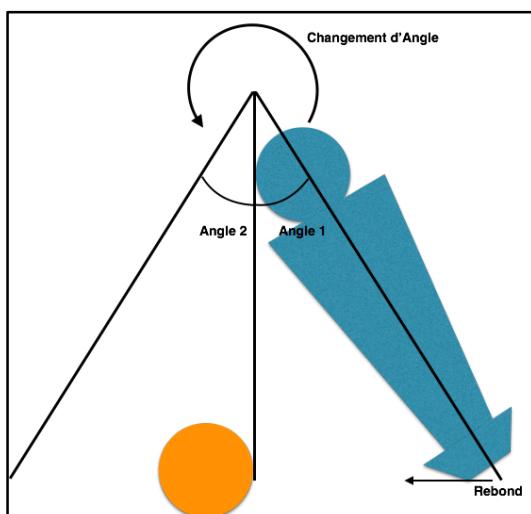


Figure 3 Amortissement itératif

On change ainsi entre l'Angle 1 et l'Angle 2 afin d'amortir la balle, en passant par-dessus pour ne pas la percuter.

Contrairement à l'amortissement suivi et l'arrêt sec, l'amortissement itératif n'a pas besoin

d'anticiper la position exacte de la balle car on peut procéder par anticipation. Il suffit que le joueur soit en face de la balle au moment où il la percute et qu'il change d'angle avant qu'il ne soit trop tard s'il y a un rebond. Ainsi, on peut contourner la prédition de la position de la balle de si on arrive à être à l'angle d'amortissement avant que la balle percute le joueur

2.2.4 Tableau récapitulatif

Table 1: Tableau comparatif des méthodes envisagées

	Besoin de prédire la position	Dépendance sur la vitesse	Changements d'angle
Amortissement Itératif	Non on peut procéder par anticipation.	Pour savoir si la balle est immobile on doit connaître la norme de la vitesse. Pour savoir si la balle a été amortie ou si elle a rebondi on a aussi besoin de la direction de la vitesse.	Pour que la balle soit bien amortie et ne passe pas en dessous les deux angles d'amortissements doivent être précis.
Amortissant Suivi	Oui et avec une très bonne précision pour amortir la balle en suivant sa trajectoire.	Les mesures de vitesses doivent être très précises pour bien prédire la position.	Pour un amortissement suivi, on doit pouvoir contrôler la vitesse du changement d'angle.
Arrêt Sec	On doit pouvoir prédire la position de la balle à l'intersection de la balle et du joueur.	Les mesures de vitesses doivent être très précises pour bien prédire la position.	Pour bien bloquer la balle, il faut que le changement d'angle soit à la bonne vitesse.

3. Exécution de la Stratégie

Pour mettre en place cette stratégie, plusieurs VI ont été rajoutés dans la partie Strategy.vi du code LabView. Plus précisément, on s'est concentré sur la ligne des attaquants. Ainsi, si on exécute le code de la version « Strategy2019 » ce sera la ligne des attaquants qui essaiera d'arrêter la balle.

3.1 Données utilisées et convention

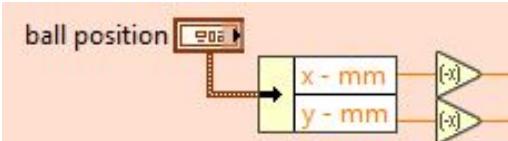
Dans le cadre de ce projet nous avons travaillé dans le sous-VI appelé Strategy.vi. Les VI et le code rajoutés modifient plusieurs inputs et outputs. Dans cette partie, on expliquera les différentes conventions nécessaires pour comprendre le fonctionnement des VIs et la stratégie.

3.1.1 VI Strategy.vi

Nous avons travaillé dans le cadre de ce VI qui est exécuté comme une boucle tant que le baby-foot est calibré. Les principaux inputs à influencer sont l'angle et la position en Y de la rangée des attaquants.

On utilise plusieurs inputs notamment la vitesse et la position de la balle, ces dernières sont calculées dans d'autres parties du code et sont des inputs de Strategy.vi. Le tableau ci-dessous explique comment les différents inputs et outputs sont obtenues :

Table 2 Tableau Explicatif des Input et Output de Strategy

Input/Output	Emplacement dans <u>Strategy .vi</u> et Explication
Position	 <p>La position de la balle est donnée comme input en forme de cluster, qui est ensuite séparé en x et en y. Afin d'avoir les bons axes on inverse les signes des coordonnées (voir <u>Figure 4</u>).</p>

Vitesse	<p>The diagram shows a VI for calculating velocity. It takes a cluster input "Position Vitesse Incertitude" and extracts the velocity component. This is then processed through a filter "Incertitude filtre" (TF) and a multiplier block. The output is the "Vitesse en X" (Velocity in X) and the "norme vitesse" (velocity norm). The velocity norm is calculated using a square root block.</p>
Angle et Linear Output	<p>The diagram shows a VI for generating angle and linear output. It takes "Angular position ref" and "Linear position ref" as inputs and outputs them into a cluster. This cluster then replaces the "References" part of the "Control cluster in" input. The output is the "Control cluster out".</p> <p>Les valeurs des angles et de la linear position (Y) sont insérées dans un cluster vide pour chaque ligne : attaquants, milieux, défenseur et goal (image à gauche). Puis, tous ces clusters remplacent les valeurs respectives (References) de l'input <u>Control cluster out</u> et donnent le <u>Control cluster in</u> (l'output), qui a les valeurs de Y et l'angle qui vont être exécutées par d'autre VI.</p>

3.1.2 Coordonnées de la position

Les coordonnées de la position sont données en input sous forme du cluster ball position à Strategy.vi à chaque itération, ces dernières sont en mm et selon le repère suivant :

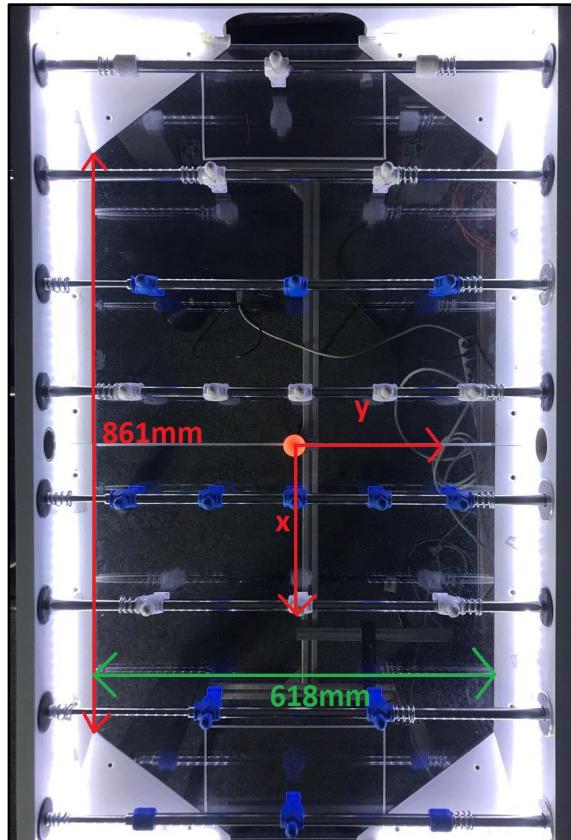


Figure 4 Coordonnées

3.1.3 Vitesse

La vitesse est donnée dans le cluster Position Vitesse Incertitude. La valeur de la vitesse est obtenue par la méthode des différences finies. Cependant, ces valeurs sont plus indicatives que précises : elles ne permettent pas de connaître la vraie vitesse. En effet, une partie de Strategy.vi utilise ces valeurs afin de prédire la position en Y des attaquants et ne marche pas.

De plus, comme la résolution de la camera est d'environ 2 mm par pixels , la norme de la vitesse n'est pas toujours nulle même si la balle est immobile. La valeur de la position oscille entre deux positions et on calcule une vitesse même si la balle est immobile. Cette valeur est cependant très faible, elle est d'environ 1 mm/s en x et en y. Il y aura donc un bruit constant au niveau des mesures de vitesse. On enregistre la norme de la vitesse lorsque le bruit est négligeable et on trouve la valeur de 5 mm/s comme limite. On considère donc seulement les vitesses avec des normes supérieures à 5 mm/s comme valide. Cela reste une vitesse très lente. Si la balle est à cette vitesse, elle est très souvent quasi-immobile ou sur le point de s'arrêter.

Cependant, les valeurs de la vitesse sont précises au niveau de la direction globale de la balle.

3.1.4 Angles

Les angles sont mesurés en degrés. On donne donc des valeurs entre $[-360^\circ, 360^\circ]$; le signe détermine le sens de rotation.

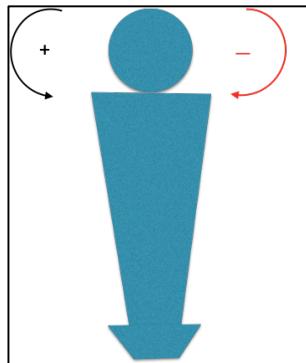


Figure 5 Convention pour le sens de Rotation des Joueurs

3.2 Division du baby-foot en zones

Afin que les angles d'amortissement optimaux soient les bons lorsque que la balle s'approche des joueurs, on divise le baby-foot en différentes zones. On catégorise les zones en deux types : celles d'anticipations et les zones critiques

Lorsque la balle se trouve dans les zones d'anticipation, les joueurs devront se mettre à l'angle de premier amortissement. Ainsi le premier amortissement dépendra que du fait que le joueur soit bien devant la balle. Il n'y aura pas vraiment de question de « timing », car il sera à l'angle de premier amortissement bien avant.

Lorsque la balle se trouve dans les zones critiques il faut d'abord savoir s'il y a eu un rebond et si on nécessite un second amortissement. La Figure 6 ci-dessous définit les différentes zones.

Le premier VI ajouté sur la partie Stratégie est If Blockable.vi, dont le block-diagram est en Annexe. Il détermine d'abord si on se trouve dans une zone critique ou non critique. Les bornes extérieures des zones critiques correspondent à la barre centrale des attaquants et l'interception aux angles optimaux.

La Zone critique est là où on va amortir petit à petit la balle. Les sous-zones de cette zone critique vont nous aider à déterminer les angles d'amortissements dans les prochain VIs.

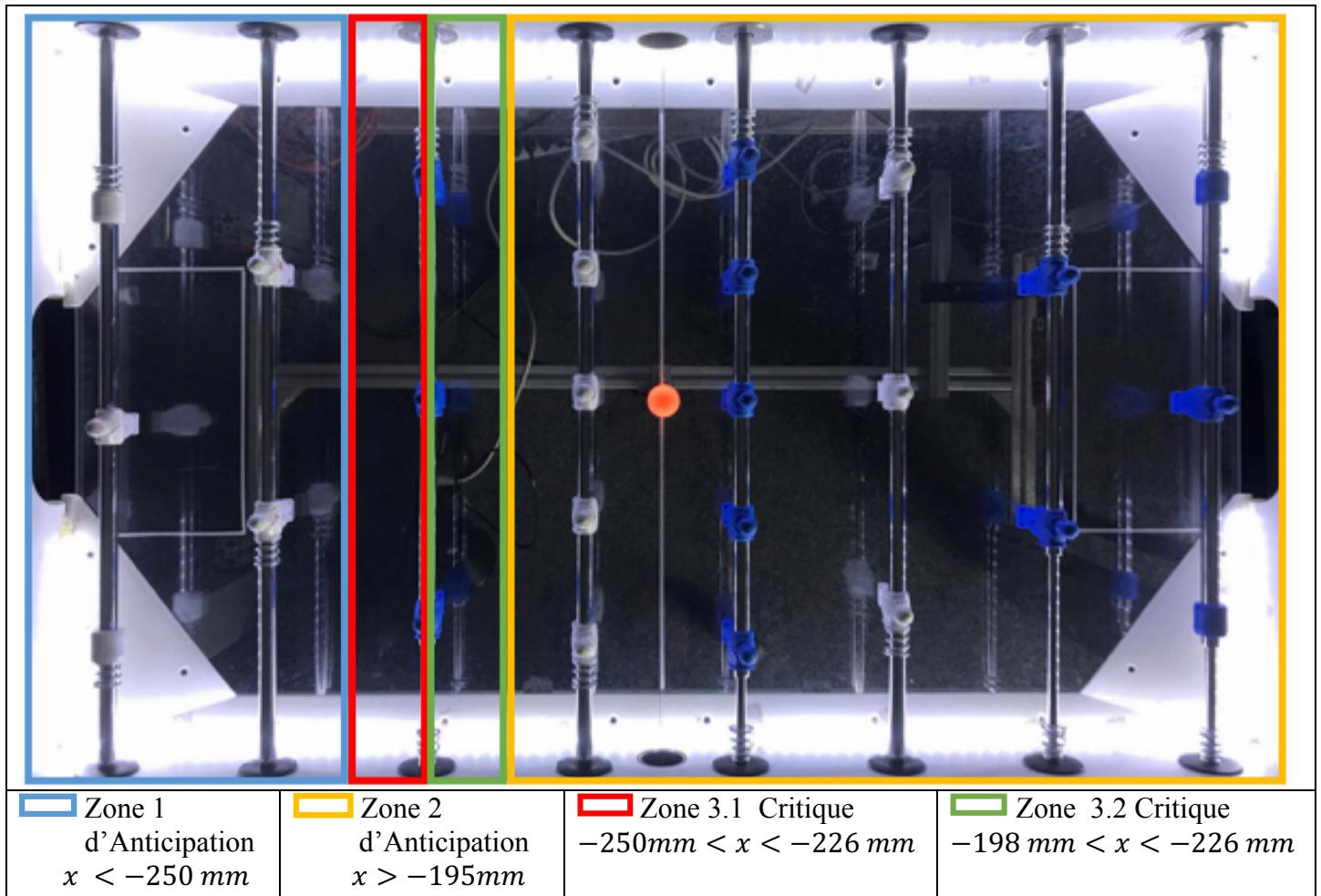


Figure 6 Zones

3.3 Angles dans les zones d'anticipation

Seules deux valeurs d'angles sont prises par les joueurs pour amortir la balle. La première valeur est de 40° et amortie la balle quand celle-ci se dirige vers les attaquants en partant du goal adverse. La deuxième valeur est de -40° et amortie la balle quand celle-ci se dirige vers les attaquants en partant de son propre goal.

On a choisi les valeurs des angles après plusieurs essais et afin de minimiser l'offset qui a lieu dans les moteurs à frictions. En effet, pour des valeurs légèrement plus hautes (-45° et 45°) on ratait plus souvent la balle après plusieurs itérations. Pour des angles plus bas (-36° et 36°) l'amortissement n'était pas assez effectif. De plus, prendre une valeur intermédiaire permet de minimiser l'impact de la calibration ou l'on doit donner la position verticale des joueurs.

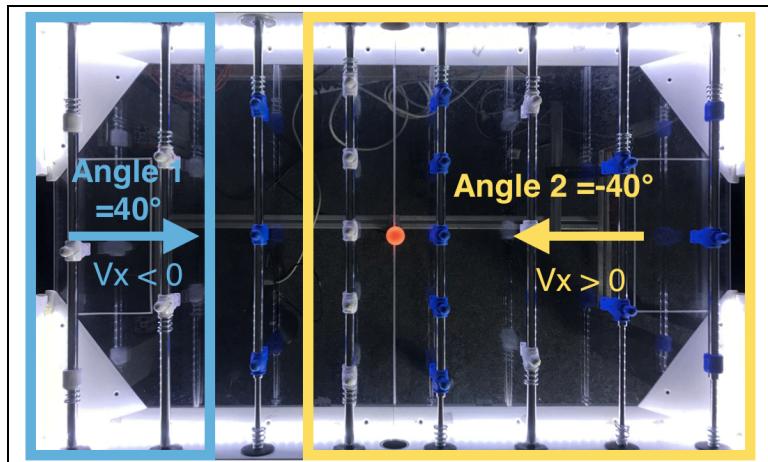


Figure 7 Premier angle d'amortissement

Le VI déterminant les angles dans les zones d'anticipation est Premier_Angle.vi, dont le block diagram est en Annexe. Il détermine le premier angle d'amortissement selon la direction de la vitesse et la position comme sur la figure ci-dessus. En effet, le premier angle d'amortissement est initialisé seulement quand la direction de la vitesse est la bonne et la balle est dans la bonne zone, sinon il garde la valeur précédente.

3.4 Angles dans les zones critiques

3.4.1 Principe général

Comme vu précédemment l'angle du premier amortissement peut prendre que deux valeurs : -40° ou 40° . Selon l'angle du premier amortissement le changement d'angle pour un éventuel deuxième amortissement sera différent. Ce changement d'angle aura lieu seulement si la balle n'a pas été assez amortie, c'est-à-dire qu'elle a changé de sous-zone critique après le premier amortissement et que sa vitesse en Vx a changé de direction. En effet, comme vu précédemment, si on élimine les cas où la vitesse calculée est en fait celle d'une balle immobile, on peut utiliser la condition pour connaître la direction de la balle. Il faut aussi veiller à ce que le changement d'angle se fasse dans le bon sens pour ne pas percuter la balle. Un schéma ci-dessous illustre comment le changement d'angle a lieu selon les deux cas.

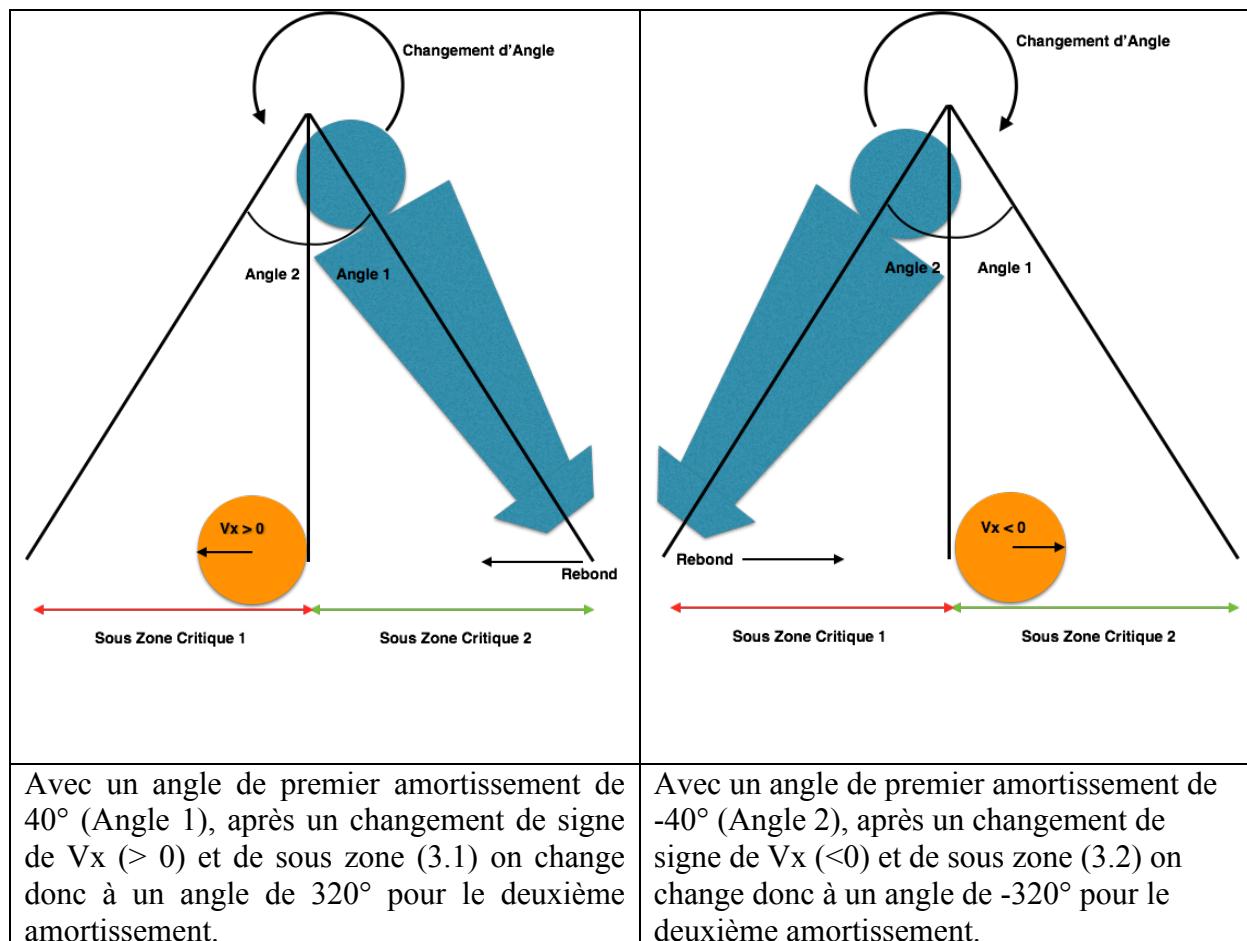


Figure 8 Schéma pour deuxième angle d'amortissement

Si on a un troisième rebond, l'angle revient à l'angle précédent car à chaque changement on change l'angle de premier amortissement dans le sous-vi Switch.vi.

Le VI Switch est activé lorsque la balle est dans la zone critique ; c'est lui qui se charge de déterminer si le changement d'angle a lieu. Ce dernier et sa description sont en annexe.

3.4.2 Mesures et Bornes

La stratégie pour arrêter la balle est basée sur le fait que la coordonnée en x est constante le long d'une ligne horizontale, c'est-à-dire qu'une trajectoire horizontale du baby-foot donne bien des mesures parfaitement horizontales après le traitement de la caméra. Ainsi, les bornes des zones et sous-zones sont considérées constantes ; on les définit comme des intervalles aux bornes fixes.

Afin de vérifier si cette hypothèse est valable on analyse sur Matlab les données récoltées lorsque la balle effectue des trajectoires en forme de lignes horizontales.

Sur la figure ci-dessous, à première vue la trajectoire de la balle est bien décrite comme une ligne droite horizontale pour chaque trajectoire le long des trois lignes de joueurs (attaquants, défenseurs et milieux).

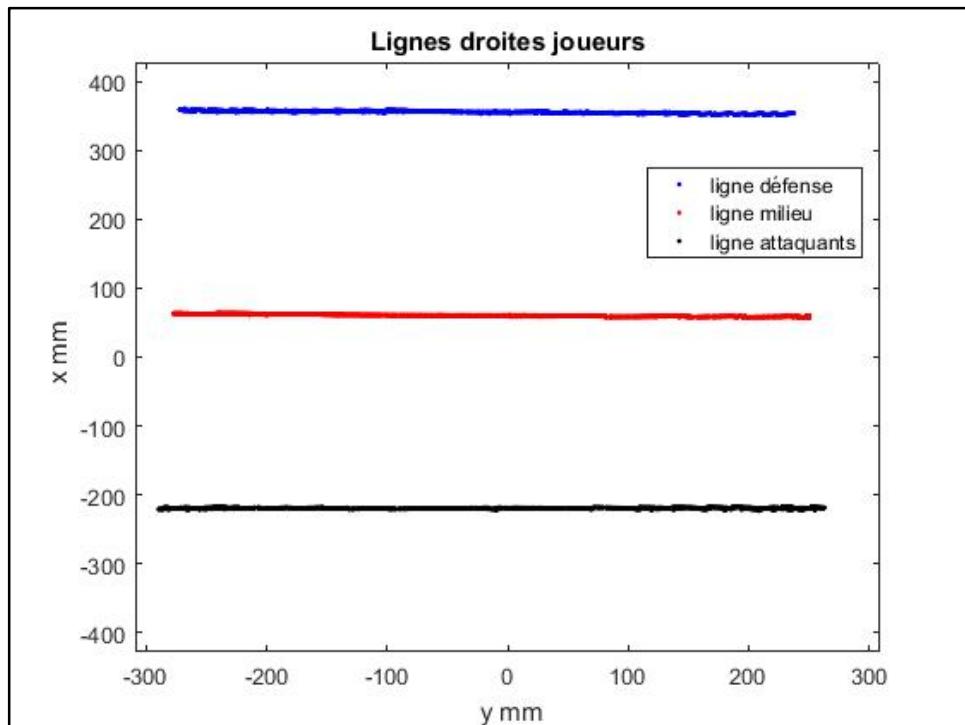


Figure 9 Cordonnées de la balle le long d'une ligne horizontale le long des défenseurs, milieux et attaquants

Nous avons ensuite analysé plus précisément une ligne droite. Afin d'avoir les mesures plus précises et de ne pas être gêné par la barre des joueurs, nous avons fait des lignes droites à $x = -208.8$ mm, la ligne des attaquants se trouvant plus bas sur le graphe à $x = -226$ mm. On effectue plusieurs allers-retours afin d'avoir assez de données. Les résultats, après un zoom, sont présentés dans la figure ci-dessous : les données ne traduisent pas une ligne droite parfaite.

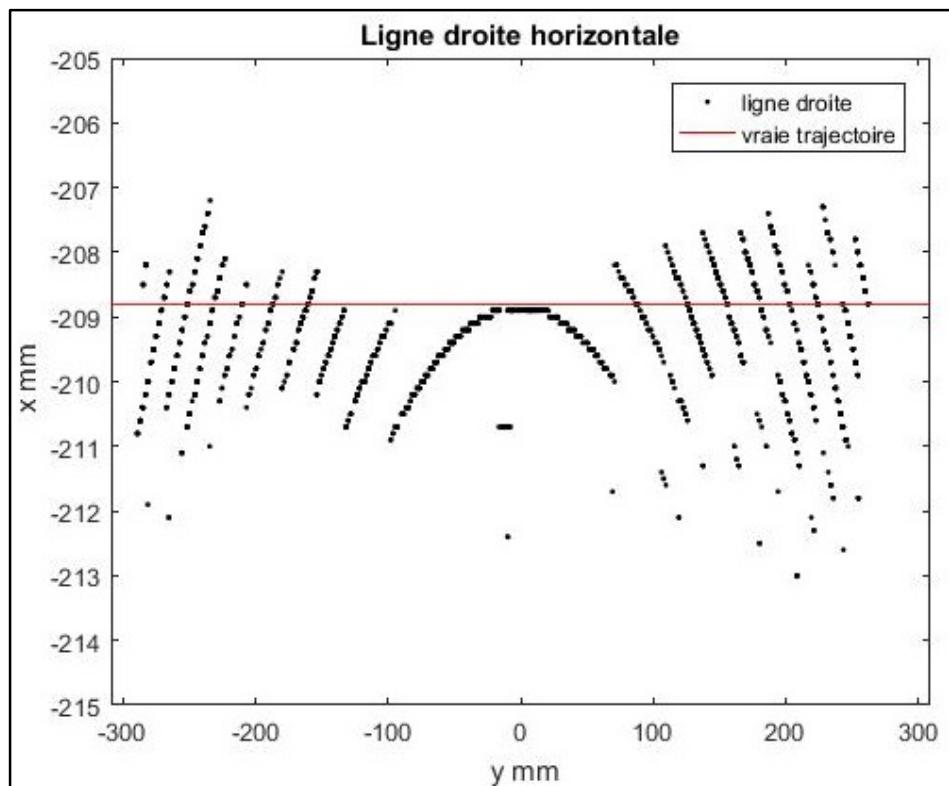


Figure 10 Graphique d'une trajectoire horizontale

Quand la balle est de 10mm de part et d'autre du centre, la ligne est parfaitement droite. Plus on s'écarte du centre, plus l'erreur entre la vraie trajectoire et la valeur mesurée augmente. On observe au maximum une déviation de + 3mm vers le haut, et une déviation de -2mm vers le bas, le long de l'axe x. Cette non symétrie s'explique par le fait que la trajectoire a été faite dans la partie supérieure du baby-foot, et non au milieu. Plus on s'éloigne du milieu, et donc de la camera, moins les mesures sont précises.

Ces écarts de 5 mm sont significatifs notamment au niveau de l'ordre de grandeur des sous-zones. Si une erreur de quelques mm parvient au bord d'une sous-zone, un changement d'angle fautif peut avoir lieu. L'utilisation de zones précises dans notre stratégie peut donc être grandement influencé par ses écarts de mesures.

3.4.3 Hystérèse

Afin de prendre en compte le fait que les valeurs des sous-zones ne soient fixes on s'inspire du principe d'hystérèse pour minimiser l'erreur et élargir les zones du rebond. Suite à des mesures on a déterminé que la postions du milieu de la sous-zone critique oscillait entre [-224 mm, -230 mm] au lieu de la valeur fixe. Ainsi dans le VI Switch la condition ne se résume plus à la l'appartenance à la sous-zone et la condition de vitesse, mais prend en compte cet intervalle du milieu.

3.5 Suivi de la balle le long de la coordonnée Y

Pour que l'amortissement ait bien lieu il faut qu'au moment où la balle traverse la ligne des joueurs, le joueur assigné soit bien en face de la balle. Pour cela deux méthodes existent :

- Soit on prédit à quel position la balle sera quand elle sera au niveau des joueurs et on y positionne le joueur.
- Soit on constraint le joueur assigné à avoir la même coordonnée Y que la balle tout le temps.

Pour cela c'est le VI find_player_forward .vi qui guide les joueurs à la position Y désirée qu'il reçoit en input.

Chaque joueur est séparé par 200mm et à une portée de 95mm de chaque côté. Si la balle se situe entre les coordonnées -95mm et 95mm, on joue avec le joueur central. Ce dernier se situe est au centre de la ligne à la position (-225 mm, 0 mm) après la calibration. Donc la barre doit seulement se décaler de la valeur de la postions de la balle en Y pour que le joueur soit en face.

Cependant si la balle se situe à une position inférieure à -95mm c'est le joueur à gauche du joueur centrale qui devra être en face de la balle, on décale donc la balle de 200mm en plus que la position actuelle de la balle pour que le bon joueur soit en face d'elle. De même, si la coordonnée en Y de la balle est supérieure à 95mm on décale la barre de -200mm en plus que la position de la balle.

3.6 Schéma récapitulatif de la Stratégie

Ci-dessous est le schéma récapitulatif de la stratégie finale implémentée par LabView.



4. Méthode prédictive de la position de la balle

Afin d'améliorer la performance du babyfoot on a cherché à prédire la position de la balle.

4.1 Méthode existante et nouvelle méthode

Si on aligne le joueur avec la position immédiate de la balle l'un des principaux problèmes est que le joueur rate la balle la plupart du temps quand cette dernière se déplace en diagonale. Il faut donc prédire son intersection avec les attaquants.

Deux méthodes différentes ont été mises en place pour prédire la position de la balle. La première fut d'utiliser un modèle prédictif avec les valeurs de vitesse calculées par les différences finies et l'autre fut d'utiliser une régression linéaire. La méthode avec les différences finies était préexistante et ne marchait pas. Elle utilisait les composantes en x et en y de la vitesse pour calculer la position prédictive de la balle avec très peu de succès.

La plupart du temps, les trajectoires de la balle peuvent être approximées par des droites. On choisit donc d'utiliser des régressions linéaires pour approximer la trajectoire de la balle, en réinitialisant à chaque changement de trajectoire.

4.2 régression Linéaire pour anticiper la trajectoire

Nous avons tout d'abord analysé des données de trajectoire de la balle sur Matlab afin de comprendre les différents problèmes et bien pouvoir visualiser. Nous parcourons les 2 vecteurs de coordonnées, et calculons une régression linéaire chaque fois que l'on parcourt un nouveau point, afin d'être cohérent avec la réalité où nous avons une paire de coordonnées l'une après l'autre :

```
1) for i=1:length(x)
2) %% Régression linéaire F(x)= b+m*x
3) L=length(x);
4) X = [ones(L,1) x ]; %% X et Y doivent etre colonnes
5) a=(X.'*X)\(X.'*y);
6) b=a(1); %% intersection avec les y
7) m=a(2); %% coefficient directeur
8) end
```

Cependant, il est important de réaliser s'il y a un changement de trajectoire afin de commencer une nouvelle régression linéaire. Lorsqu'il y a un changement de direction, il faut donc oublier les données précédentes et ne considérer que les nouvelles données.

La méthode qui s'est avérée la plus performante est expliquée dans le schéma suivant :

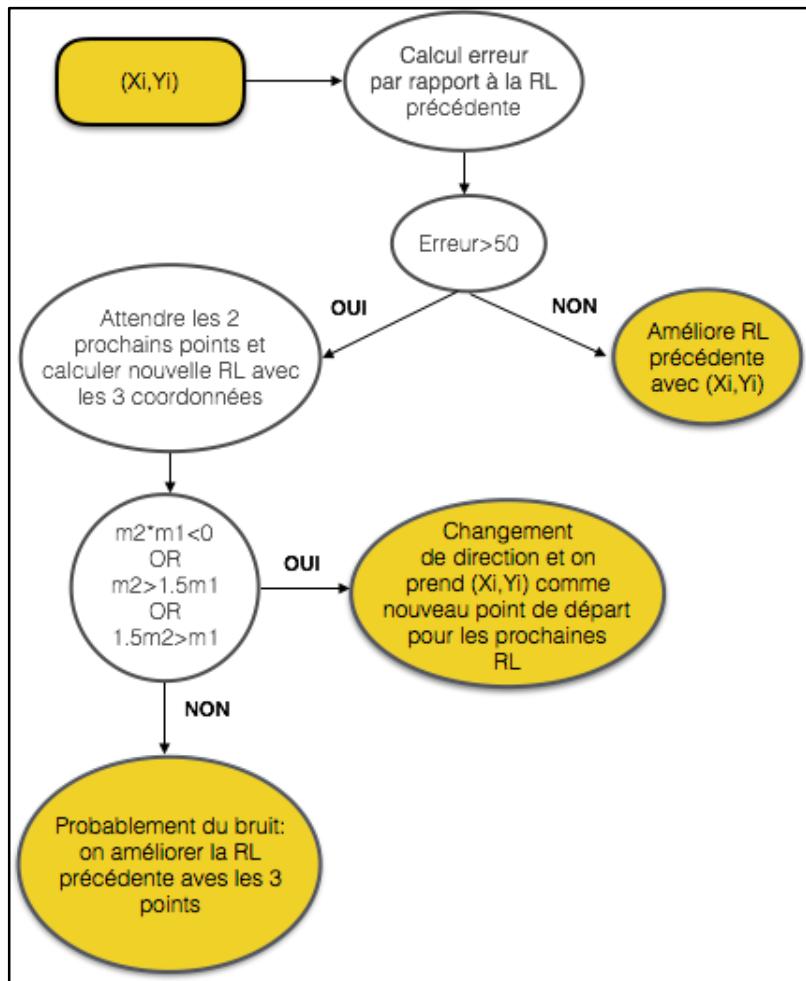


Figure 11 Schéma de la Régression Linéaire adaptative

Ici, m_2 signifie le nouveau coefficient directeur calculé, et m_1 l'ancien.

L'erreur limite (50) ainsi que les facteurs limites pour les coefficients directeurs (1.5) ont été trouvé de manière empirique en testant de nombreuses valeurs. Le code Matlab complet est donné en annexe, avec des graphiques permettant de visualiser le résultat.

L'inconvénient de cette méthode est que l'on a besoin d'au moins 3 points avant de détecter un changement de direction. Si la balle est très rapide, elle parcourt beaucoup de distance durant ces 3 points et la méthode est donc peu précise.

Une méthode permettant de calculer la différence en x et en y entre les différents points afin d'en déduire un changement de direction a été étudiée, mais ne s'est pas avérée performante à cause de la présence de bruit.

4.3 Implémentation de la régression linéaire sur LabView

D'abord, on a implémenté une régression linéaire sans prendre en compte les changements de trajectoire afin de voir son efficacité sur LabView, et si elle permettait de prédire la position pour arrêter un tir diagonal.

Soit $y = ax + b$ la trajectoire approximée de la balle, avec les coefficients a et b trouvés par la régression linéaire. On peut donc calculer à chaque itération la valeur de y pour un certain x et ainsi obtenir la valeur qu'aura y lorsque la balle croisera la ligne des attaquants (la ligne des attaquants ayant une coordonnée en x fixe), afin d'intercepter la balle.

On a comparé cette méthode prédictive avec celle utilisée. On rappelle que la méthode utilisée place le joueur à la même position en y que la balle à chaque instant. Les 2 méthodes avaient finalement les mêmes problèmes : elles n'arrêtent pas les tirs trop rapides en diagonales. Lors d'un tir rapide du défenseur adverse, la distance entre ce dernier et la ligne des attaquants étant faibles, seulement quelques points de la trajectoire de la balle sont pris en compte. A cause du bruit de mesures, ces points ne sont pas assez nombreux pour avoir une bonne prédiction à tous les coups. Cependant, pour un tir en diagonale lent, il semble que la régression soit plus appropriée pour arrêter le tir. En effet, comme nous pouvons le voir en comparant les 2 vidéos en annexe, lorsque l'on utilise une régression linéaire, le joueur se positionne là où va aller la balle. Pour la méthode classique, le joueur bouge à chaque itération et a ainsi plus de chance de taper la balle au lieu de l'arrêter.

Cependant, nous avons finalement jugé que l'avantage de la méthode de régression linéaire n'était pas significatif, et que le petit avantage ne compensait pas la difficulté d'implémenter le changement de régression linéaire lors de changements de direction.

5. Performance du code LabView

Les différents VIs mis en place ont une performance mitigée. Plusieurs facteurs extérieurs contribuent à perturber la performance.

5.1 Strength/ Courant dans les moteurs à friction

La strength qui peut être modulée directement du Front Panel détermine le courant qui fera tourner les moteurs à friction. Plus la strength est importante plus les mouvements en rotation des joueurs et donc le changement d'angle seront rapides et brusques. Cette dernière est aussi une output de Strategy.vi : on peut donc la modifier directement dans la partie stratégie.

Plusieurs tests ont été effectués pour que le changement d'angle ne soit pas trop brusque ou trop fréquent, mais assez rapide. On a choisi une plage de courant qui permet d'arrêter la balle assez souvent. Cet intervalle est de [1,70 ; 1,90]. On pourrait encore plus augmenter le courant notamment jusqu'à 2.00 et au-delà, mais à cause de l'offset produit par les moteurs à friction on reste en dessous. En effet, plus le courant est important plus l'offset apparaît rapidement. Ainsi, même si un courant supérieur permet de mieux arrêter la balle initialement l'offset le rend moins performant assez rapidement.

5.2 Offset moteur

Comme le moteur est un moteur à friction, un certain offset commence à apparaître au niveau des angles après plusieurs rotations. On remarque que ce dernier fait que l'Angle 1 augmente et l'Angle 2 diminue. Ainsi, l'amortissement est réduit pour l'Angle 2 et la balle passe dessous pour l'Angle 1.

Quand l'angle augmente à cause de l'offset, la balle peut même être mesurée au bord de la zone critique. La position de la balle oscille entre l'intérieur et l'extérieur de la zone. L'angle va varier rapidement entre l'angle de premier amortissement et second amortissement, donnant l'impression que le joueur tape la balle. Cela peut aussi se produire si la calibration est mauvaise ou suite à une erreur de mesure.

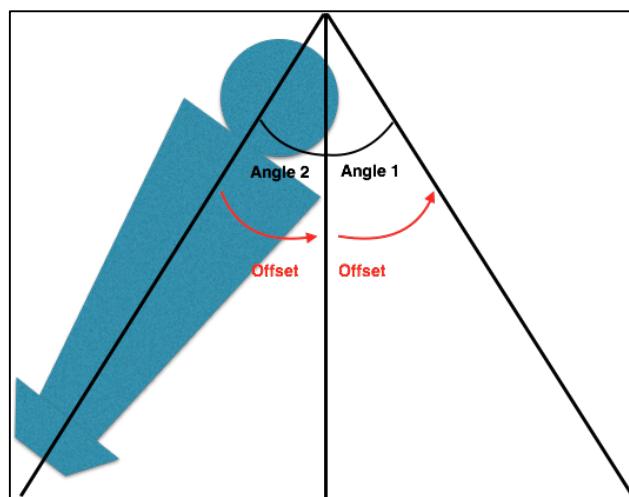


Figure 12 Effet de l'offset sur les Angles

5.3 Amélioration

Le code LabView fonctionne mais les incertitudes de mesures et le mode de contrôle du moteur limite sa performance. Des idées d'amélioration ont été explorées, comme rajouter des hystérèses au niveau de la limite entre les zones d'anticipation pour contourner les erreurs de vision mais ces dernières n'ont pas amélioré la performance du code.

Afin de ne pas rater la balle, nous avons aussi pensé à effectuer le changement d'angle en deux temps et à deux vitesses différentes pour qu'il soit plus rapide, mais cela n'a pas été implémenté à cause de la structure itérative du Strategy.vi. En effet, nous avions un sous VI qui fonctionnait mais ce dernier ne fonctionnait pas quand inséré dans Strategy.vi. A cause d'un manque de temps nous n'avons pas pu aboutir. De plus, même si la rapidité nous permettait de minimiser l'impact des mesures fausses, le changement de vitesse ne nous permettait pas d'avoir un angle assez précis. Il est aussi possible que le changement d'angle à une vitesse supérieure crée un offset et à son tour rende les angles moins précis. Ce VI (Modulation.vi) est expliqué en Annexe.

5.4 Analyse vidéo

Trois exemples d'arrêt de la balle aux conditions optimales sont donnés dans les vidéos: Exemple arrêt balle 1, Exemple arrêt de la balle 2 et Exemple arrêt de la balle 3.

Si le courant est trop bas, le changement d'angle n'a pas lieu assez rapidement. La balle est donc ratée par le joueur comme dans la vidéo Exemple arrêt manqué- lent.

Si la balle est mesurée à l'intérieur de la zone alors qu'elle est en fait à l'extérieur, le joueur la percute et effectue un tir comme dans la vidéo Exemple arrêt manqué- tir.

Ce sont les deux erreurs les plus fréquentes.

6. Conclusion

Le bilan du projet est assez positif même si la performance du code est limitée. Nous avons pu avancer considérablement dans plusieurs aspects, notamment dans l'annotation du code et la compréhension du fonctionnement du baby-foot. Cette tâche fut assez fastidieuse, mais indispensable afin de pouvoir au mieux intégrer notre stratégie dans le code existant et faire en sorte qu'il soit réutilisable. Ainsi, nous avons annoté tous nos VI afin qu'il soit facilement compréhensible pour les prochains groupes.

Malheureusement, des erreurs et problèmes techniques ont retardé de plusieurs semaines l'intégration de notre code dans la partie stratégie du baby-foot. Une erreur de la caméra arrêtait subitement la vision de la balle après seulement quelque seconde, rendant impossible de tester notre VI sur le code. Finalement, avec l'aide de Monsieur Salzmann nous avons pu contourner ce problème après quelques semaines.

Notre VI arrête la balle pour un nombre non négligeable de cas, mais le taux de réussite est assez haut seulement pour des conditions optimales et fragiles. Non seulement ces dernières sont restreintes, mais elles se détériorent au fur et à mesure d'une partie, notamment à cause de l'offset. Dans une vraie partie les conditions seront différentes et l'offset pas aussi prononcé avant un temps considérablement plus long. En effet, on ne mobilise pas autant les attaquants dans une vraie partie. Cela va dépendre de comment on incorpore l'arrêt de la balle dans notre une stratégie plus générale.

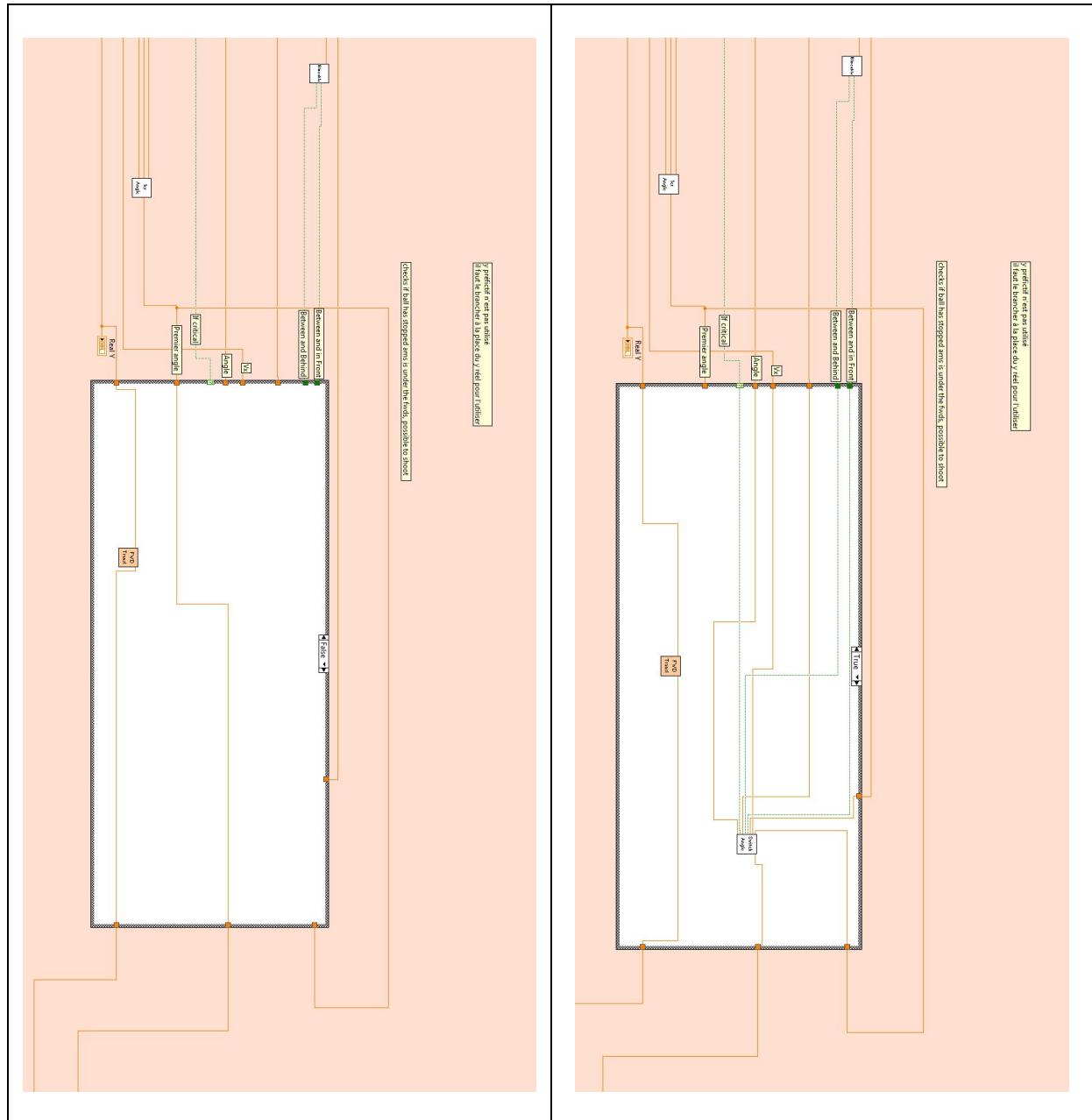
Si la vision et les mesures s'améliorent, on peut affirmer avec une grande certitude que le code sera plus performant. En effet, non seulement les erreurs dues aux sous-zones et zones seront réduites si la caméra a une meilleure résolution, mais on pourrait mieux prédire la direction de la balle pour les tirs en diagonales. Ainsi cela permettrait d'incorporer la prédiction de la balle dans le code et

7. Annexe

7.1 VI Pour arrêter la balle

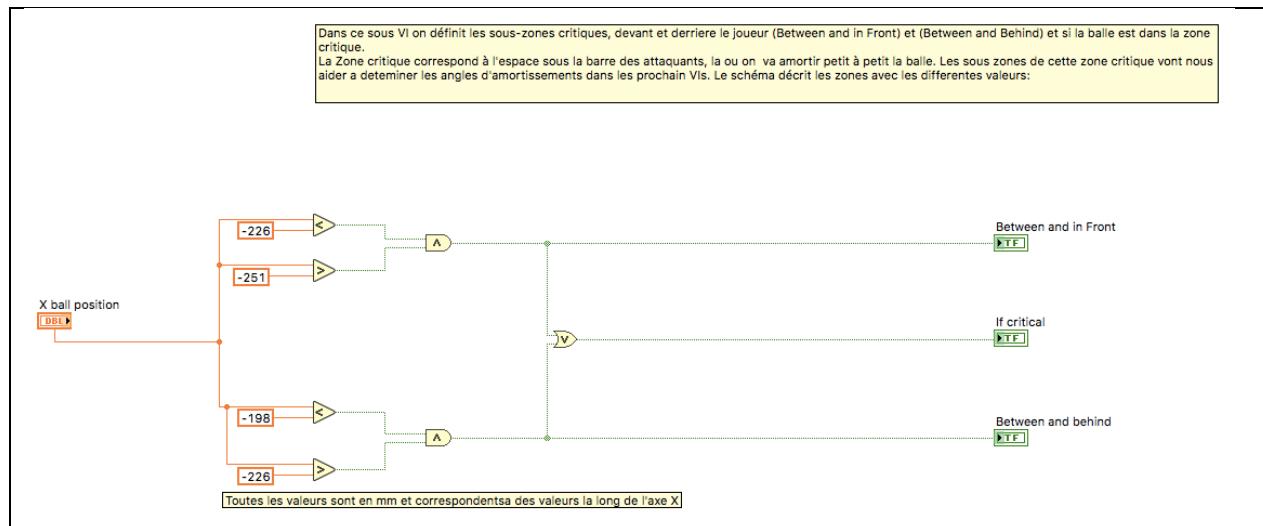
Dans cette partie on décrit les différents sous VI pour arrêter la balle

7.1.1 Emplacement dans Stratégie :

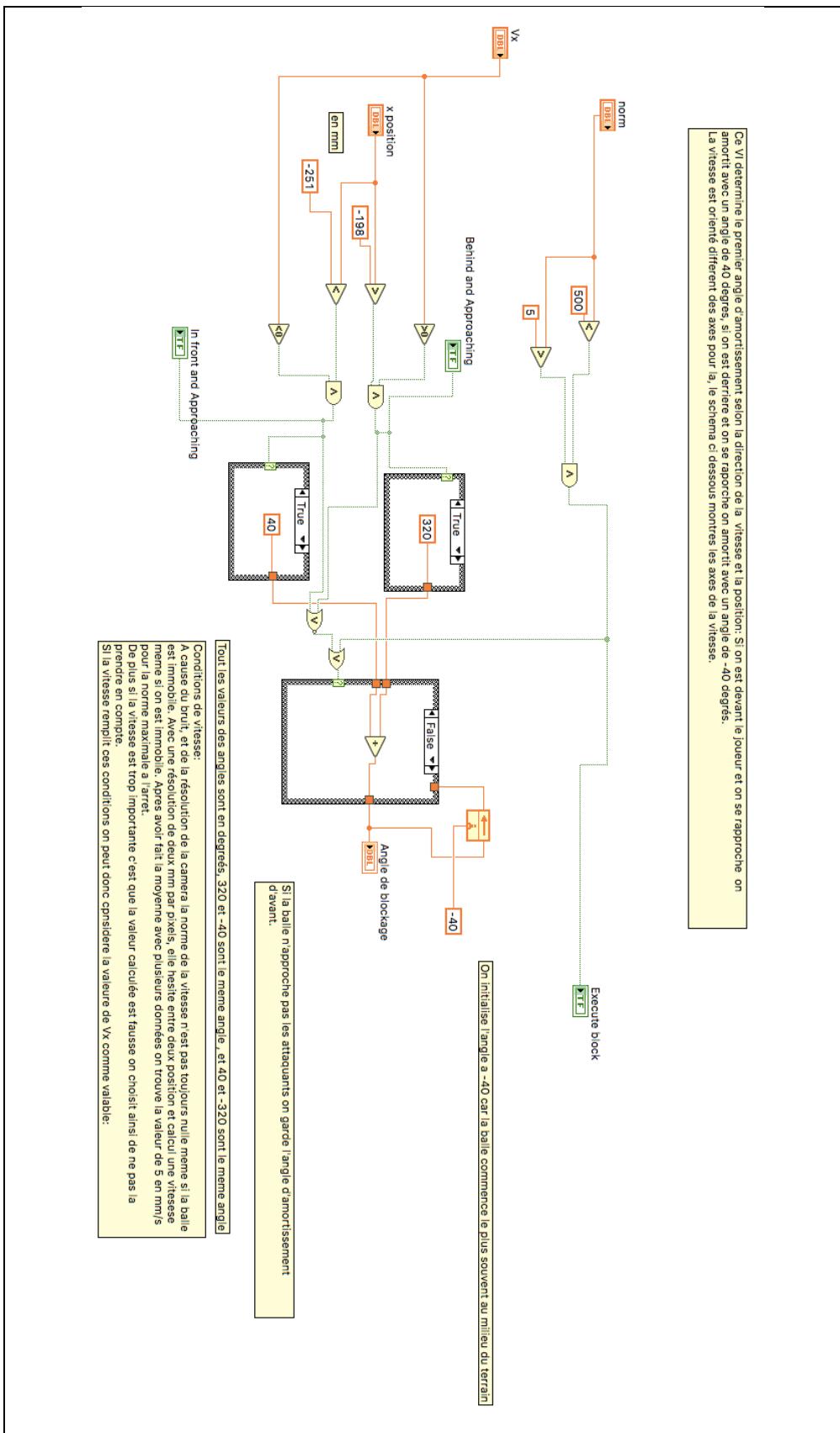


7.1.2 Contenu des sous VI pour trouver la balle

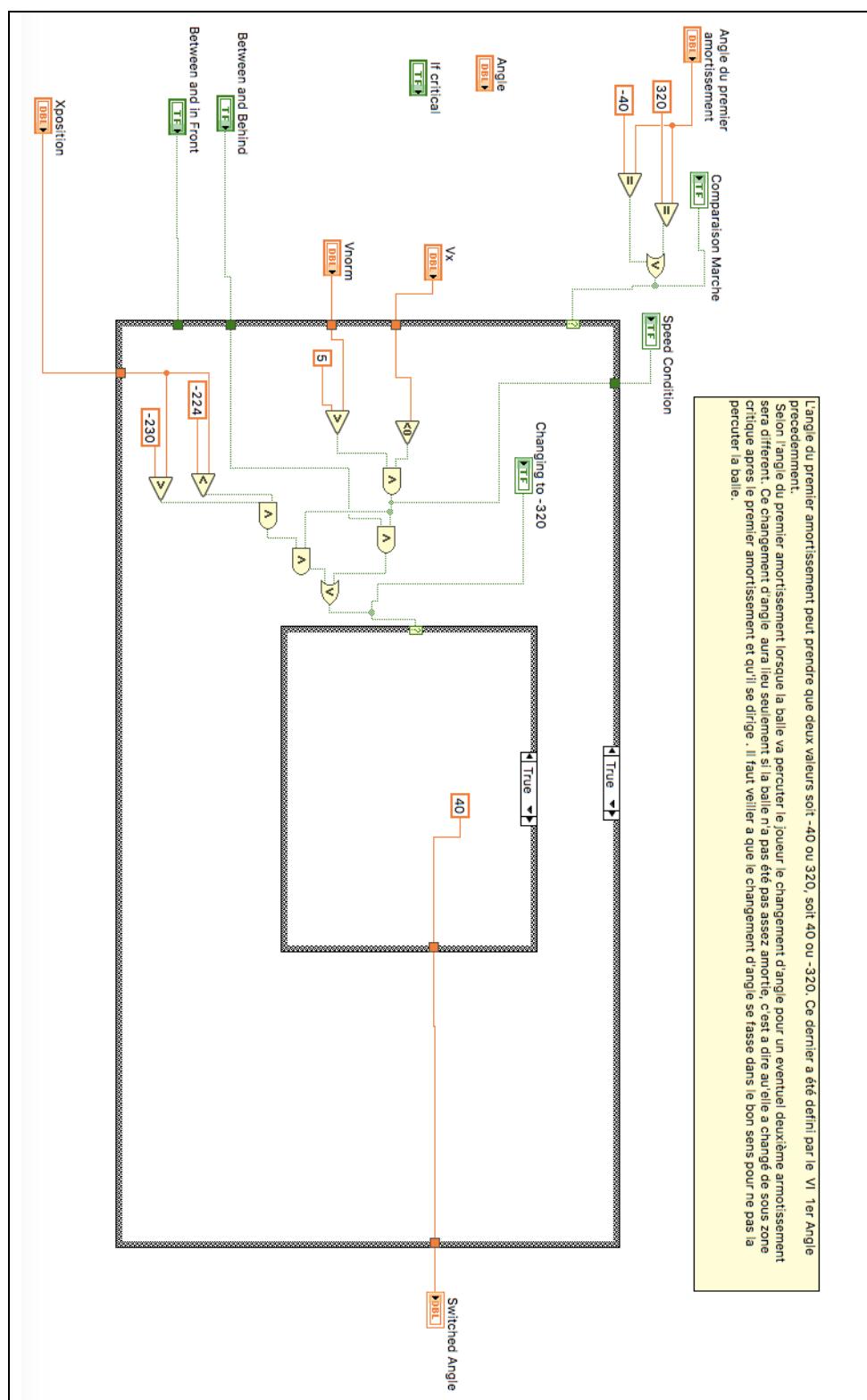
7.1.3.1 If Blockable.vi



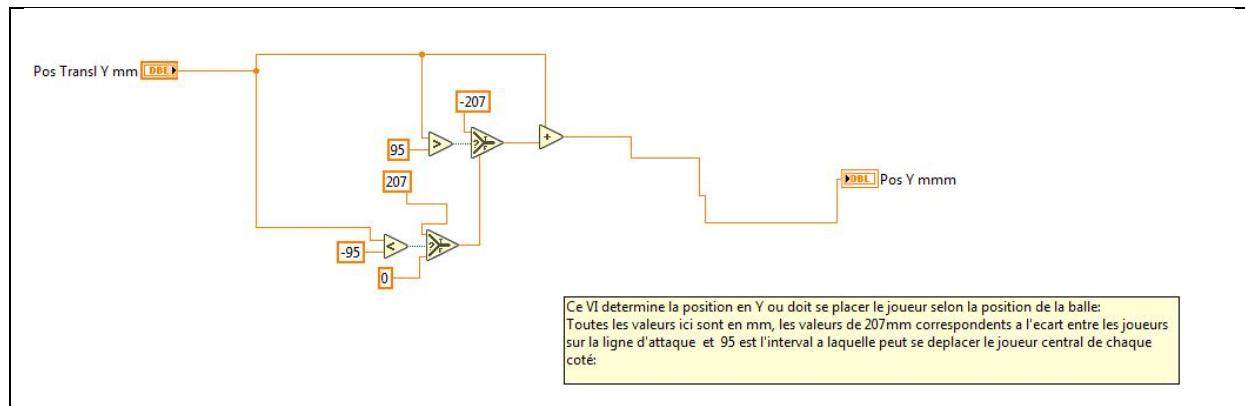
7.1.3.2 Premier_Angle.vi



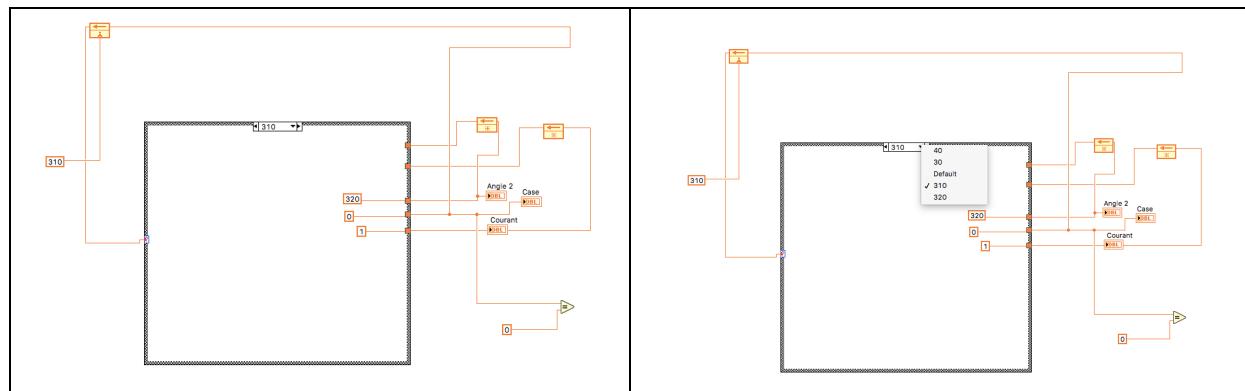
7.3.1.3 Switch.vi



7.3.1.4 Find Player forward



7.3.1.5 Code de modulation de l'angle non utilisée Modulation.vi



Description :

Ce code est supposé diviser en deux mouvements la rotation d'un joueur jusqu'aux angles 320° ou 40°. La première partie de la rotation s'effectue à une valeur importante de strength pour que la rotation soit plus rapide. La première rotation va jusqu'à l'angle choisie moins 10°, donc à 310° pour 320° et 30 ° pour 40°. Puis, dans un deuxième temps jusqu'à l'angle finale mais cette fois ci à une valeur de strength moins importante.

L'angle finale voulue est la constante donnée en input.

7.2 Regression Linéaire et Mesures

7.2.1 Code Matlab

```
clear all;
close all;
clc;
load('trajectoire')
%% CODE MATLAB: Etudier les régressions linéaires sur les données de la balle
% Les données enregistrées par LabView ne sont pas dans le bon ordre
x=trajectoire(:,2);
y=trajectoire(:,1);
% La première paire de coordonnée est toujours(0,0) et on l'efface
x(1)=[];
y(1)=[];
taille=size(x);
count=0;
count2=0; %% Compter les matrices M
ERR=[];
%% on enlève les coordonnées doubles pour faciliter les régressions linéaires. On ne veut pas étudier
ici le cas d'une vitesse nulle.
for i=2:taille(1)
    i2=i-count;
    if (x(i2)==x(i2-1))&&(y(i2)==y(i2-1))
        x(i2)=[];
        y(i2)=[];
        count=count+1;
    end
end
x2=x;
y2=y;
% Vecteur des intersections b
B=[];
% Vecteur des coefficients directeurs m
M=[];
Xa=[];
Xb=[];
for i=2:(length(x2)-1)

    if i==2
        %% FIT TO F(x)= b+m*x
        iterlim=1
        x=x2([1:3]);
        y=y2([1:3]);
        N=length(x);
        X =[ones(N,1) x ]; %% X et Y doivent être colonnes
        a=(X.'*X)\(X.'*y);
        b=a(1); %Intersection axe y
        m=a(2); %Coefficient directeur
        %% CALCULATE LINE FOR GRAPHICS
        xa=min(x);
        xb=max(x);
        xinterp=linspace(xa,xb,2); % we only need 2 points for a line
        f= b+m*xinterp;
        B=[B b];
        M=[M m];
        Xa=[Xa xa];
        Xb=[Xb xb];
    end

    if i>2
        if m==0
            err=abs(y2(i)-b)
            end
        c=y2(i)-m*x2(i);

        if m~=0
            err=abs((c/m-b/m)*(sin(atan(m)))); %% Erreur entre yi et la régression linéaire
            end
        ERR=[ERR err];
    end
end
if (err>50) %% Valeur arbitraire pour vérifier que la RL approxime bien la data.
```

```

%% Si l'erreur est aussi grande, c'est anormal et il faut étudier le cas. On aura besoin de
quelques prochains points pour étudier si on calcule une nouvelle RI ou on améliorer celle d'avant

%% Si la balle va très vite, il faut peut être repérer plus rapidement dans quel cas on est,
plutot qu'attendre les prochaines points

%% FIT TO F(x)= b+m*x

%% itération a partir de laquelle commence la régression linéaire

x=x2([i:i+2]);

y=y2([i:i+2]);

N=length(x);

X =[ones(N,1) x ];% X et Y doivent étre colonnes

a=(X.'*X)\(X.'*y);

b=a(1);

m=a(2);

count2=count2+1

if (m*M(count2)<0) || (abs(m)>1.5*abs(M(count2)))||abs(m)<0.6667*abs(M(count2))) %% On estime
qu'il y a ici changement de direction

%% FIT TO F(x)= b+m*x

%% itération a partir de laquelle commence la régression linéaire

% CALCULATE LINE FOR GRAPHICS

iterlim=i;

xa=min(x);

xb=max(x);

xinterp=linspace(xa,xb,2); % we actually only need 2 points for a line

f= b+m*xinterp;

B=[B b];

M=[M m];

Xa=[Xa xa];

Xb=[Xb xb];



else %% On estime ici au'il n'y a pas de changement de direction et on améliore la version
précédente

%% FIT TO F(x)= b+m*x

%% itération a partir de laquelle commence la régression linéaire

x=x2([iterlim:i+3]); %% après un changement de direction, il faut prendre plusieurs points
pour avoir un bon début de trajectoire

y=y2([iterlim:i+3]);

N=length(x);

X =[ones(N,1) x ];% X et Y doivent étre colonnes

a=(X.'*X)\(X.'*y); %% first numbeer = b, second number y intersction

b=a(1);

m=a(2);

xa=min(x);

```

```

xb=max(x);

xinterp=linspace(xa,xb,2); % we actually only need 2 points for a line
f= b+m*xinterp;

B=[B b];
M=[M m];
Xa=[Xa xa];
Xb=[Xb xb];

end

end

end
end

plot(x2([1:(length(x2)-1)]),y2([1:(length(y2)-1)]),'.b');
hold on;
for i=1:length(M-1)

xinterp=linspace(Xa(i),Xb(i),2);
f= B(i)+M(i)*xinterp;
plot(xinterp,f,'-r','LineWidth',1.5);

end

axis([-300 350 -550 550]);

```

7.2.2 Approximations de la trajectoire par Régression Linéaires

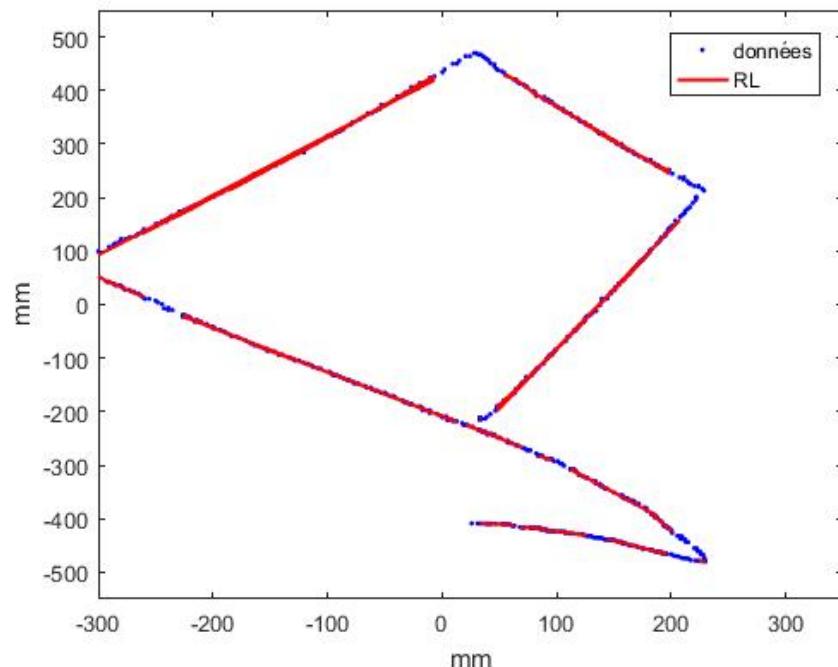


Figure 13 Approximations de la trajectoire par Régression Linéaires

7.2.2 VI Régression Linéaire.vi

On remplit les tableaux X et Y avec les différentes positions de la balle, et calculons une Régression Linéaire avec tous les données du tableau.
 Si le coefficient d'accroissement ne prend pas des valeurs extrêmes (correspondant à une trajectoire parallèle à la ligne de défense, qui ne nous intéresse pas ici),
 ainsi que si la balle n'est pas à l'arrêt (Condition v<10 à cause du bruit), alors on calcule le y prédictif. Sinon, on prend le y correspondant à la position de la balle comme
 d'habitude.
 Un curseur booléen permet de vider le tableau des X et Y afin de faire une nouvelle Régression linéaire dans recommencer le programme lorsque l'on fait des tests.

