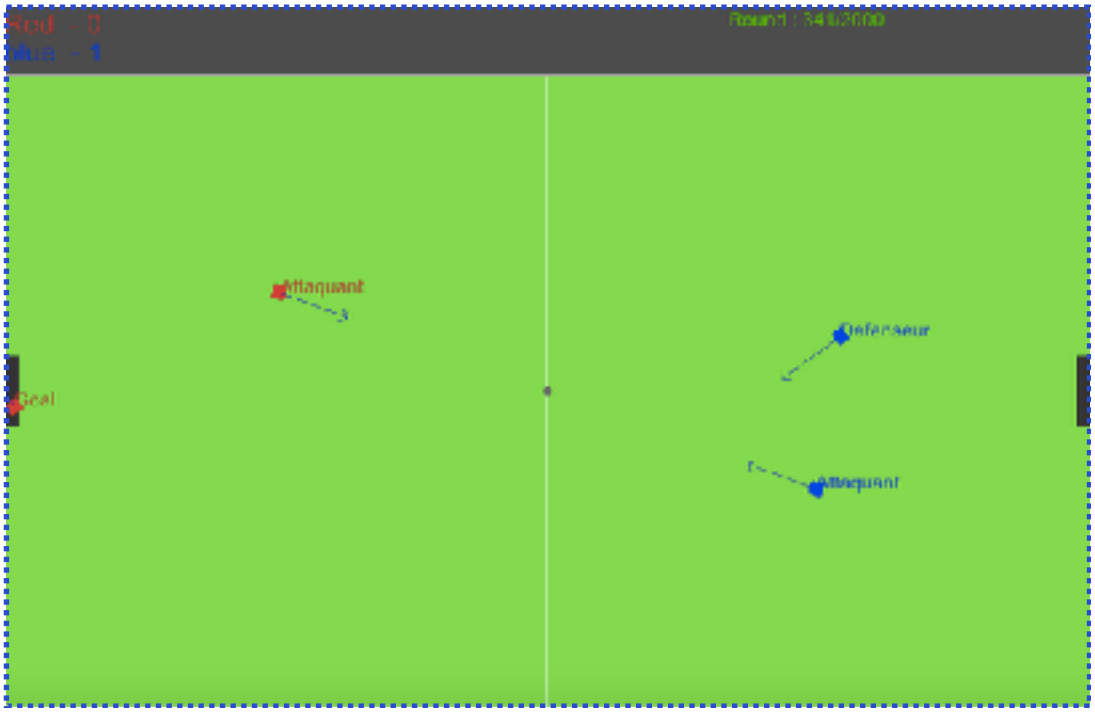
LACOUX Coline
FERRIEUX Théo

2I013
Année 2018

Projet Foot

Rapport final



SORBONNE UNIVERSITÉ
CRÉATEURS DE FUTURS
DEPUIS 1257

Sommaire

Introduction

Lors de cette UE 2I013 projet, nous avons été amenés à mener à bien un projet consistant à créer une équipe ordinateur autonome capable de jouer intelligemment un match (attaquer, se défendre, marquer des buts) sur un simulateur de foot.

C'est la première fois que nous avons eu à travailler sur un projet aussi concret mais aussi imposant, et durant tout un semestre. Il a donc fallu nous organiser au fil des semaines pour réfléchir à de nouvelles stratégies, travailler en autonomie pour mettre en place la théorie que nous voyons en cours, et proposer des équipes toujours plus performantes.

L'objectif de ce rapport est de synthétiser tout ce que nous avons mis en place pour mettre en oeuvre notre projet. Comment nous, sommes nous, organisé, quels ont été les éléments importants dans la conception de nos joueurs ?

Dans un premier temps, nous avons du mettre en place une organisation de groupe et réfléchir à l'élaboration de nos stratégies. Puis nous verrons comment nous avons mis en place notre code au cours des semaines. Ensuite, il a fallu optimiser nos stratégies afin d'en tirer le meilleur. Pour conclure, nous verrons que ce projet nous a apporté tout au long du semestre beaucoup de nouvelles connaissances et capacités.

Partie 1 : Théorie du projet

L'UE 2I013 projet application aura été l'opportunité pour nous de travailler pour la première fois pendant notre licence informatique sur un objectif constant au cours d'un semestre entier. En effet, là où les autres cours proposaient seulement un travail final dans les dernières semaines, nous avons pu dès la première séance commencer à manipuler l'environnement sur lequel tout notre travail repose. Nous avons pu donc très rapidement nous organiser sur la façon de procéder dans les semaines suivantes.

A - Organisation

Après avoir formé le binôme, la première étape a été la configuration de l'outil Github. En effet, nous avons pu travailler à distance grâce à celui-ci tout en s'assurant que le travail de chacun serait compatible. En complément, nous avons également pris les dispositions nécessaires afin de pouvoir se contacter mutuellement : cela permettait à chacun d'informer l'autre de ses avancements sans pour autant mettre à jour le dépôt commun.

Les séances du lundi servaient principalement à rassembler et expérimenter sur notre code, avec apport de nouvelles stratégies élaborées au cours de la semaine passée. L'environnement des salles d'informatiques était optimal pour ces tâches. De plus, la présence des professeurs permettait d'obtenir des réponses à des questions souvent au-delà de nos propres compétences d'étudiants. À l'issue de chaque séance, nous en profitions pour mettre à jour le dépôt distant, regroupant nos travaux jusqu'à ce jour.

Le travail en dehors des séances relevait surtout de l'optimisation du code déjà existant (correction de problèmes, simplification du code) et de la réflexion. En effet, c'est durant la semaine que nous élaborions chacune des stratégies sous l'aspect théorique, avant de la matérialiser la séance suivante.

À l'approche des dernières semaines, nous avons également pris la décision de travailler plus régulièrement dans l'enceinte de l'université, même en dehors des séances de TP et TME.

B – Réflexion

La première source d'inspiration qui nous a servis dans l'élaboration des stratégies a été nos propres connaissances (certes relativement limités) sur le football. Le simulateur étant directement hérité de ce sport, il nous paraissait évident qu'établir des comportements également inspirés de la réalité serait la première étape dans notre travail.

Toutefois, il y a un vaste écart entre le comportement d'un joueur réel et celui d'un carré de pixels guidé par des lignes de code. Ainsi, le comportement "défense" d'un joueur s'est transformé en simple "le joueur se positionne sur le vecteur des cages à la balle" dans les premières semaines, et s'est progressivement orienté vers des directives plus complexes.

Une autre source d'idées quant à l'élaboration de nouvelles méthodes a été les autres simulations de football déjà existantes. En effet, ces jeux sont dans l'idée très similaires à notre propre projet, mais à une échelle bien plus grande et dans un environnement bien plus sophistiqué. Toutefois, on pouvait quand même y discerner différents comportements, notamment le dribble basique, la feinte afin de contourner un adversaire, l'action de se démarquer afin de recevoir une passe etc.

C – Compétition

La plus grande particularité de ce travail au cours des semaines a été l'affrontement hebdomadaire des différentes intelligences artificielles créées par les joueurs. En effet, tous les étudiants travaillant sur un même projet, il était possible de comparer le travail de chaque groupe. Bien qu'assez triviale à première vue, elle est vite devenue une source de motivation et d'inspiration pendant tout le reste du semestre.

Motivation, car c'est bel et bien l'esprit de compétition qui pousse à s'investir plus encore dans ce travail. Être en bas du classement pousse à vouloir grimper, et se retrouver plus haut que la semaine précédente semble être une récompense de l'investissement fourni. Bien évidemment, chaque groupe pouvant escalader le classement, d'autres groupes perdaient ainsi leur place.

C'est donc dans ce cadre-là que la compétition devenait également source d'inspiration : c'est les défaites qui permettaient de réfléchir à de nouvelles stratégies. Quelqu'un qui ne perd jamais aura moins d'idées afin d'améliorer son code alors qu'au contraire un perdant cherchera à contrer la stratégie de son adversaire à la séance suivante.

Enfin, la compétition a également donné lieu à de nouvelles méthodes de travail. En premier lieu, l'élément de surprise qui peut intervenir lors d'une mise à jour à la dernière minute des comportements de chaque joueur. Cette méthode cependant est à usage unique, la stratégie restant après cela accessible à quiconque.

Toutefois, dans un cadre plus large, cela nous a également permis de découvrir un outil dédié à Github qui nous sera très utile dans nos futures études : le mode éducatif. En effet, celui-ci permet aux étudiants, entre autres, de créer des répertoires privés sans avoir à payer un tarif spécial. Bien que peu utile dans ce cadre, il permettra tout de même à l'avenir d'éviter le plagiat.

Partie 2 : Technique

Après réflexions et mise en place théorique de notre projet, nous avons mis en place nos stratégies de manière concrète en language python.

A - Structure, architecture logicielle :

Pour organiser notre projet, nous avons créé un module (dossier modulesocc) contenant les fichiers suivants :

- _init_.py : C'est le fichier servant à configurer le module, c'est à dire à signaler à python que notre dossier est un package.

- etat.py : Ce fichier contient la classe Etat, qui encapsule la classe state contenue dans le simulateur. On y retrouve des fonctions très simples permettant d'obtenir des informations propres à l'état courant, par exemple la position de la balle, d'un joueur ou des cages, le numéro du joueur allié le plus proche, la distance entre deux positions, le numéro du joueur le plus proche de la balle. On y retrouve aussi des fonctions plus complexes, telles que trajectoire_adv(self, id_t) qui renvoie un booléen indiquant si la balle va passer à proximité d'un joueur adverse, ou encore atteindre_balle(self, id_t, id_p, nb_t) qui renvoie un booléen indiquant si un adversaire va atteindre la balle avant le joueur.

- actions_simples.py : Ce fichier contient diverses fonctions permettant de faire des actions simples, telles que courir vers la balle grâce à la fonction dirballe(etat, id_t, id_p, norme, k) qui calcule la position de la balle dans k tours, courir vers un position grâce à dirpos(etat, id_t, id_p, norme, pos), faire une passe, tirer…

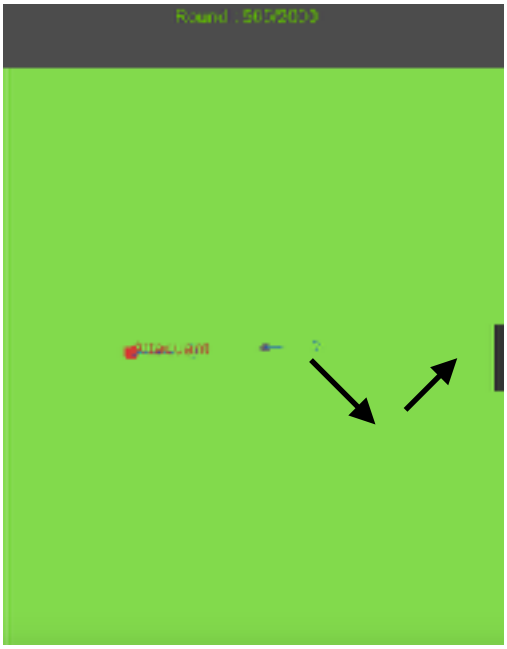
- Strategies.py : C'est le fichier contenant toutes les stratégies que nous appliquons à nos joueurs.

B - Les stratégies

◉ 1v1

C'est d'abord lors de parties en 1v1 que nous avons commencé à élaborer notre première stratégie, celle du fonceur. C'était au départ une stratégie très basique qui consiste à courir après la balle et tirer vers les cages dès que possible.

Rapidement, nous avons souhaité complexifier cette stratégie en créant un fonceurZigzag, qui plutôt que de foncer droit vers les cages, est capable d'anticiper la position de la balle, et va tirer un coup sur un côté du terrain, un coup au centre, afin de déstabiliser l'adversaire qui ne pourra pas anticiper ses gestes.

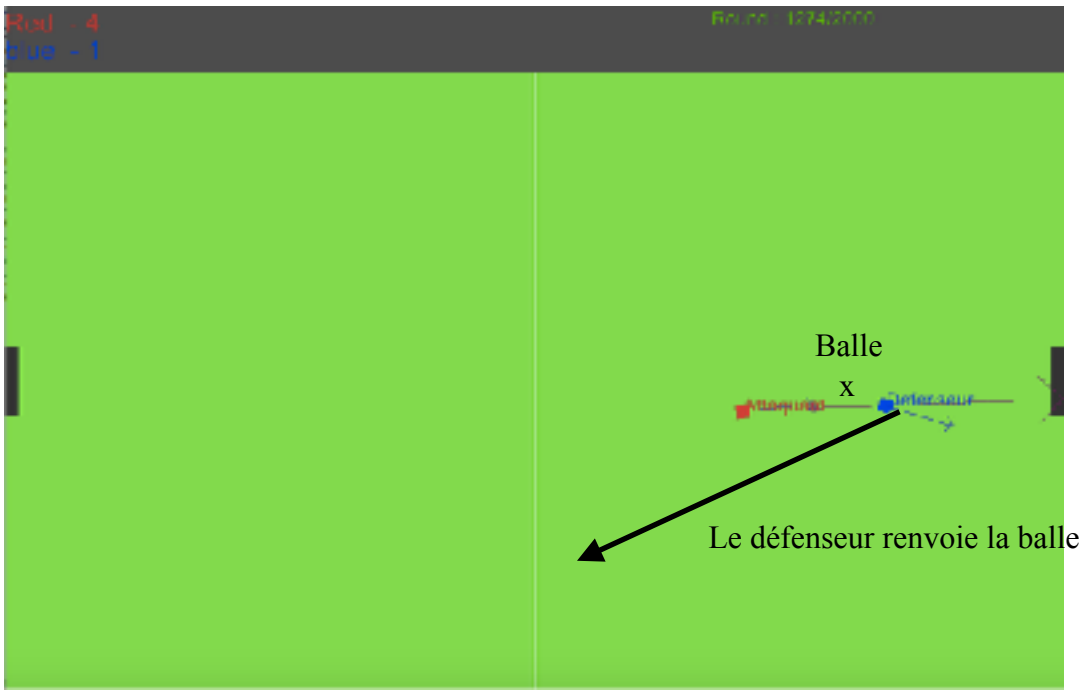


Nous avons avec le temps abandonné cette stratégie et opté pour une stratégie liant à la fois l'attaque et la défense : selon sa position sur le terrain et celle de l'adversaire, le joueur va avoir un comportement agressif ou défensif.

◉ 2v2

C'est lorsque nous avons abordé le mode 2v2 que nous avons crée notre première une vraie stratégie de défense. En 1v1, nous nous concentrions majoritairement sur l'attaque, alors qu'en 2v2 nous pouvons séparer une stratégie offensive d'une stratégie défensive.

Le défenseur viens se positionner entre les cages et le joueur adverse, près à intercepter la balle. Il est capable d'anticiper la position de la balle dans 10 tours ce qui lui permet de l'intercepter quand elle se rapproche trop des cages et de l'envoyer vers l'attaquant.

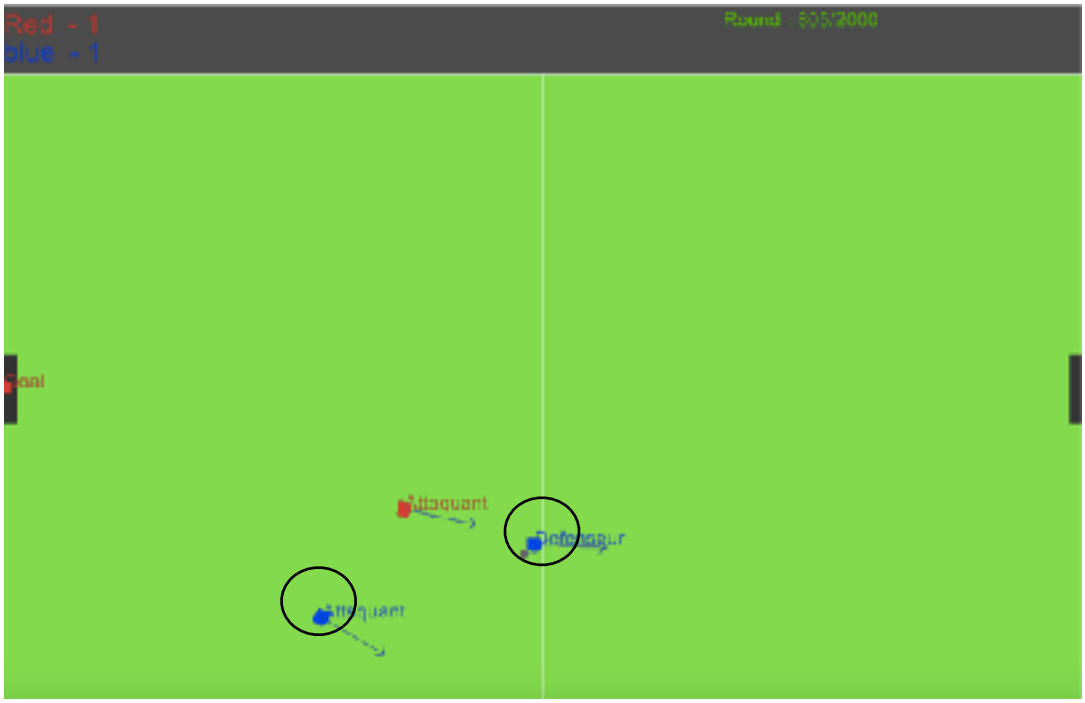


L'attaquant quant à lui se met en position d'attaque lorsqu'il n'a pas la balle, c'est à dire qu'il va se démarquer de manière à ce que le défenseur puisse lui faire une passe, puis dribbler vers les cages en évitant les adversaires en faisait des feintes, puis tirer lorsqu'il est assez près des cages.

A la fin du semestre, nous avons eu l'idée d'ajouter une condition, celle du score actuel, à notre attaquant. Si l'équipe a l'avantage, l'attaquant, plutôt que de prendre des risques en essayant de tirer, va temporiser et tout faire pour garder le ballon le plus loin possible de ses cages. Nous avons eu l'idée d'essayer de coincer la balle dans un coin à l'aide de nos deux joueurs, auquel cas il serait impossible pour l'adversaire de récupérer la balle, mais cela n'est pas possible avec ce simulateur car les collisions entre joueurs n'existent pas et donc l'adversaire peut passer au travers .

Nous avons donc opté pour un autre solution : quand l'attaquant peut frapper, il tire la balle très fort dans un coin adverse afin de ralentir l'adversaire.

Sur cette image, on voit l'attaquant démarqué et le défenseur prêt à lui faire la passe :



◉ 4v4 et améliorations avec les mini-stratégies

Lors des dernières semaines, nous avons revu presque intégralement nos stratégies dans le but de mettre en place un algorithme d'apprentissage et une KeyBoardStrategy. Nous avons pour cela crée un grand nombre de mini-stratégies répondant à un besoin simple.

Foncer_centre, Tempo_centre, Intercepter_centre, Aller_goal, Aller_defense, Aller_passe, Dribble, Feinte, Tir, Defense, Sortir_goal, Degager sont des exemples de ces mini-stratégies.

Nos nouvelles stratégies se sont donc composées de la manière suivante :

- En 1v1, nous avons constitué un mélange de micros-stratégies tirées des 2v2 ainsi que de quelques nouvelles. Elles dépendent de plusieurs conditions (notamment, le fait que la balle va vers les cages ou non, le fait qu'un joueur adverse est en train de dribbler avec la balle, le fait qu'un joueur adverse atteigne la balle plus rapidement etc.) et sont au nombre de 5.

- En 2v2, nous avons ajouté deux comportements bien différents de notre équipe 2v2 habituelle. En effet, elle changera complètement de stratégie en fonction des scores. En cas d'avantage dans le score, elle va essayer de jouer beaucoup sur la défensive et de gagner un maximum de temps. En cas d'égalité, elle va essayer de jouer très agressif afin de prendre l'avantage. Enfin, en cas de désavantage, elle va jouer de manière modérée, utilisant notre comportement 2v2 habituel optimisé par apprentissage artificiel (voir partie 4). Au total, la stratégie en 2v2 comportes 18 micro-stratégies, réparties sur 3 situations différentes pour 2 joueurs différents.

-4v4 : Nous n'avons que peu approfondit le cas des équipes à 4 joueurs. La stratégie est essentiellement héritée des 18 micro-stratégies en 2v2, certaines arrangées pour les 4v4 sans fondamentalement changer le fonctionnement. La principale stratégie que nous avons crée pour le 4v4 est celle du goal : celui-ci va d'abord se positionner dans les cages, puis lorsqu'un adversaire approche il sa se décaler de manière à se positionner entre la balle et les cages, mais tout en restant très proche des cages. Lorsque la balle arrive vers les cages, il la dégage vers un défenseur. Si la balle est assez proche des cages et qu'il peut l'atteindre avant l'attaquant adverse, le goal peut sortir de ses cages pour intercepter la balle.

Partie 3 : Optimisation

L'élaboration de nouvelles stratégies toujours plus complexes n'est pas la seule méthode permettant de progresser. En effet, certaines stratégies, pourtant assez simplistes dans leur fonctionnement, peuvent dépasser des stratégies bien plus complexes de par leur optimisation. En effet, pour un code inchangé, ces dernières fonctionnent avec les valeurs optimales et sont bien plus performantes que des stratégies certes complexes mais inégales dans leurs conditions et paramètres.

A - Algorithmes génétiques et paramétrisation

La première optimisation des stratégies abordées a été celle des paramètres. Pour ce faire, il fallait commencer par s'assurer que l'intégralité de notre code était paramétrable, et quand ce n'était pas le cas, on changeait des constantes en valeurs modifiables. Il nous a donc fallu revoir l'intégralité de notre code afin de pouvoir paramétrer la moindre fonction, même une fonction aussi simple que celle de la position à viser lors d'un tir vers les cages.

Toutefois, en parallèle de la paramétrisation de notre code, nous avions commencé à établir une nouvelle stratégie plus complexe dédiée aux 2v2, notre précédente stratégie ne pouvant réellement bénéficier des avantages d'une optimisation des paramètres. En effet, si une stratégie est fondamentalement mauvaise, des paramètres optimaux ne suffiraient pas à la rendre performante. Ainsi, nous avons décidé de focaliser notre attention sur cette nouvelle stratégie plus à la place.

La question des algorithmes génétiques et du brassage s'est rapidement ajoutée à nos préoccupations sur la priorité à accorder à notre stratégie même ou à l'optimisation des paramètres de cette dernière. L'idée de faire s'affronter nos propres stratégies avec des ensembles de paramètres différents de manière automatisée, tout en conservant les meilleurs candidats, nous semblait davantage intéressante, mais ne répondait toujours pas au problème de notre stratégie actuelle.

Ainsi, une stratégie issue d'un brassage sera certes la meilleure parmi tous les ensembles de paramètres possibles, mais ne sera réellement optimisée que contre elle-même. Des stratégies tout à fait différentes pourraient sans difficulté dépasser cette dernière étant donnée qu'elle n'a pas été optimisée contre les autres stratégies.

Nous avons donc conservé notre attention sur la stratégie en 2v2 plutôt que l'optimisation de ses paramètres, il s'agissait là d'un pari risqué mais qui a tout de même payé en premier lieu. En effet, une fois cette nouvelle stratégie établie pour la semaine 8, sans même que cette dernière soit optimisée, elle a atteint la deuxième position, manquant de quelques points la première place occupée par une équipe qu'elle a battue lors de l'affrontement. Cet exploit fut de courte durée car nous avons rapidement été rattrapés par la suite, mais il s'agissait tout de même d'un moment peu commun, étant donné notre retard dans le classement les semaines précédant l'arrivée de notre nouvelle stratégie.

B – Apprentissage artificiel

L'optimisation des arbres décisions nous a tout de suite semblé plus intéressante que celle des paramètres, surtout pour nos stratégies qui reposent déjà sur de nombreuses conditions et sous stratégies. Ainsi, il a fallu préparer notre code pour cette fois-ci une méthode d'apprentissage artificiel. Ce dernier serait assisté par les étudiants afin de déterminer le meilleur arbre de décision pour les différentes sous stratégies.

La première étape a donc été de créer de multiples « mini-stratégies », comme expliqué dans la partie 2. La plupart ont été directement tirées de stratégies déjà existantes, tandis que nous avons en plus ajouté des nouveaux comportements spécifiques pour les joueurs. En effet, il s'agissait d'une opportunité de choix pour créer de nouvelles stratégies plus performantes. C'est à ce moment là que nous avons également commencé à aborder les 4v4, que nous n'avions que survolé jusqu'à présent sans réellement concevoir de stratégies spécifiques à ce mode de jeu.

Une fois le concept défini, il nous fallait désormais guider l'algorithme d'apprentissage artificiel. Pour ce faire, nous avons utilisé la KeyboardStrategy mise à notre disposition. Avec cet outil, nous pouvions basculer manuellement d'une stratégie à l'autre en fonction des situations. L'algorithme capture chaque état où l'on change les stratégies, et se constitue une base de données avec les différentes conditions. Lorsqu'il aurait assez de situations, l'algorithme serait capable de générer un arbre de décision sur nos différentes conditions ("est-ce que le joueur adverse va atteindre la balle avant le joueur joué", "est-ce que l'équipe a l'avantage du score sur l'équipe adverse" etc.), et la stratégie serait prête à fonctionner d'elle-même.

Toutefois, même si le code était prêt à être utilisé dans l'objectif de l'apprentissage artificiel, nous ne sommes pas parvenu à utiliser l'algorithme générateur d'arbre de décision correctement, et notre travail est resté sur cette matière est resté de l'ordre de la théorie.

Partie 4 : Apport, expérience personnelle

Travailler sur ce projet tout au long du semestre aura été l'occasion de découvrir de nombreuses méthodes de travail et de programmation.

En premier lieu, il s'agissait de la première fois que nous utilisions git (et ainsi par extension, Github). Il s'agit d'un outil extrêmement pratique dans tout type de projet en programmation. La gestion de branches ainsi que de versions permet d'éviter les pertes, mais également de contribuer à plusieurs sur un seul projet sans risquer d'écraser le travail de chacun. Nul doute que cet outil sera très utilisé dans la suite de nos études en informatique.

De plus, comme expliqué dans notre paragraphe sur l'organisation du travail, nous avons pu débloquer l'accès au mode éducatif de Github, rendant cet outil d'autant plus efficace pour nos besoins personnels.

Ensuite, il s'agissait également d'une première approche de l'intelligence artificielle, même s'il s'agit d'une forme assez primitive de cet aspect de l'informatique. C'est une des notions les plus utilisées actuellement dans l'informatique, elle ne cesse d'évoluer et se rend utile dans presque tous les domaines. Ainsi, il s'agit des premiers pas sur une partie cruciale de la programmation d'aujourd'hui.

Lors de notre projet, nous n'avons pas pu exploiter le potentiel des algorithmes génétiques, toutefois nous avons pu acquérir des notions rudimentaires sur la question, notamment à travers les mécanismes du brassage et la paramétrisation de notre code.

Enfin, c'est surtout l'apprentissage artificiel qui aura joué un rôle-clé lors de notre travail. Cela fait en effet plusieurs années que l'on entend parler de machine learning sans jamais vraiment en voir l'usage, mais ce projet a permis de concrétiser nos quelques connaissances en la matière. Bien que guidés au cours de ce semestre, nous avons tout de même pu acquérir des bases qui nous serons sûrement très utiles, au moins dans les situations d'apprentissage assisté.

Ce projet aura donc eu un apport certains sur nos connaissances et capacités dans la programmation informatique au sens large, touchant à de nombreux domaines. De plus, voir son travail se concrétiser chaque semaine par l'intermédiaire des matchs est un apport non négligeable, tout l'inverse des quelques situations abstraites que nous avons pu rencontrer par le passé à entrer des lignes de code sans réelle influence.