# III/ Une application des algorithmes génétiques : jeu de football

Le but de cette expérience était de réaliser un jeu simple en collaboration avec un ami, puis d'y faire s'affronter nos 'intelligence artificielles' programmées indépendamment. Nous avions envisagé de faire s'affronter une IA 'génétique' et une IA 'réseau de neurones', malheureusement l'IA 'réseau de neurones' n'a jamais vu le jour par manque de temps. L'IA 'génétique', quand à elle, nous a tout d'abord surpris par son efficacité, avant que nous soyons forcé de reconnaître ses limitations, qui tiennent à notre implémentation des méthodes d'algorithmique génétique plus qu'à ces méthodes en elles même. Toutefois nous nous permettons maintenant d'émettre quelques réserves quand à l'application de ces méthodes sur des problèmes du type de celui que nous allons décrire. Autant sommes nous forcés de reconnaître l'imperfection de notre implémentation, autant l'éradication de ces imperfections nous semble très complexe en restant purement dans le cadre d'algorithmes génétiques.

## A/ Description du jeu

Le jeu met en compétition 2 joueurs, représentés par des disques dans un espace bidimensionnel. L'espace contient un terrain carré dans lequel évoluent ces joueurs et un ballon, lui même représenté par un disque. Un 'moteur physique' gère l'interaction entre les joueurs, la balle et les limites du terrain. Les forces physiques présentes sont :
L'**inertie** des joueurs et de la balle
Des forces de **répulsion** entre les joueurs et la balle, entre les joueurs entre eux, entre les joueurs et les limites du terrain, et entre la balle et les limites du terrain. Ces forces de répulsion entrent en action quand la distance entre deux éléments sujets à ces forces sont proches l'un de l'autre.
Des forces de **viscosité** s'opposant aux mouvements des joueurs et de la balle.

Le but d'un joueur est de pousser la balle en dehors de la limite droite du terrain, le but de l'autre joueur étant de pousser la balle en dehors de la limite gauche du terrain.

A chaque pas de temps, le moteur physique fournit à chaque joueur la description du terrain (sa position, la position de la balle, la position de l'adversaire, les vitesses de ces éléments), et requiert de chaque joueur une 'intention'. L'intention est un vecteur qui représente la direction suivant laquelle se propulsera le joueur. On donne aux joueurs la possibilité de se propulser comme s'ils disposaient d'un petit 'réacteur' dont la puissance est limitée et qui est orientable dans une direction (la direction du vecteur intention). Cette direction n'est pas la direction selon laquelle va réellement se déplacer le joueur, car il sera sujet à son inertie qui tendra à l'entraîner dans la direction vers laquelle il se déplaçait précédemment, et à des forces d'interaction avec d'autres éléments du monde. Cette direction n'est qu'une force supplémentaire, une accélération qui s'ajoute à d'autres accélérations. Ainsi le contrôle d'un joueur n'est pas trivial, à la manière dont on pourrait imaginer qu'il n'est pas trivial de contrôler une pierre de curling sur une patinoire à l'aide de petits réacteurs qui seraient fixés dessus.

Capture d'écran d'une partie en cours (les traits partant du centre des joueurs représentent leur intention)

**B/ Le génotype**

Comme nous l'avons vu précédemment l'utilisation d'algorithmes génétiques suppose avant tout l'élaboration d'une technique de transcription entre une chaîne de bit constituant le génotype et l'ensemble des solutions. Ici nos solutions sont des comportements de jeu. Nous avons développé une méthode de transcription qui nous semblait offrir suffisamment de liberté pour ne pas trop influencer sur le type de comportement vers lequel la convergence s'effectuerait, qui idéalement ne devrait pas dépendre de la méthode de transcription. Malheureusement il est impossible de dissocier le type de solution trouvé de la méthode de transcription. L'ensemble des solutions acceptables étant infini et l'ensemble des solutions codables sous forme de génotype étant fini, la méthode de transcription utilisée oriente déjà la convergence par une contrainte dans l'exploration de l'espace des solutions.
Il est important d'évaluer un compromis entre un espace des génotypes vaste qui élimine le moins possible de solutions non codables et un espace de génotypes restreint qui rendra la recherche de solution plus aisée et plus rapide.
Nous nous somme arrêtés sur un codage sur 950 bits du comportement des joueurs, divisé en 10 sous-comportements codés sur 95 bits affectés chacun de poids. Nous avons appelé ces sous-comportement des 'gènes'. Notre génotype est donc constitué de 10 gènes codés sur 95 bits. La structure d'un gène est la suivante :

P0 in [(P1 IV1) R1 (P2 IV2)] TV [(P3 IV3) R2 (P4 IV4)] [(P5 Iv5) (P6 Iv6) (P7 Iv7)]
Poids ---------- CONDITION ---------- ----------INTENTION ------

P0 est le poids pondérateur du gène complet. Un gène code un comportement du type : Si un ensemble de conditions sont réalisées alors une intention est produite. L'intention finale du joueur sera la somme des intentions de chacun des gènes pondérées par leurs poids respectifs.

Les différents éléments du gène explicité plus haut correspondent à :

| | | |
|---|---|---|
| P0 | Poids global du gène | 6 bits |
| in | Bit Inutile | 1 bit |
| P1 | Poids 1 de variable 1 de sous-condition 1 | 6 |
| IV1 | indice de variable 1 de sous-condition 1 | 7 |
| R1 | Relation : > ou < de la sous-condition 1 | 1 |
| P2 | Poids 2 de variable 2 de sous-condition 1 | 6 |
| IV2 | indice de variable 2 de sous-condition 1 | 7 |
| TV | Tableau de vérité liant sous-condition 1 et sous-condition 2 | 4 |
| P3 | Poids 3 de variable 1 de sous-condition 2 | 6 |
| IV3 | indice de variable 1 de sous-condition 2 | 7 |
| R2 | Relation : > ou < de la sous-condition 2 | 1 |
| P4 | Poids 4 de variable 2 de sous-condition 2 | 6 |
| IV4 | indice de variable 2 de sous-condition 2 | 7 |
| P5 | Poids 5 de vecteur de sous-action 1 | 6 |
| Iv5 | indice de vecteur de sous-action 1 | 4 |
| P6 | Poids 6 de vecteur de sous-action 2 | 6 |
| Iv6 | indice de vecteur de sous-action 2 | 4 |
| P7 | Poids 7 de vecteur de sous-action 3 | 6 |
| Iv7 | indice de vecteur de sous-action 3 | 4 |
| | **Total** | **95** |

La condition d'application du gène est évaluée comme suit :
On dispose d'un ensemble de 128 variables réelles descriptives du terrain de jeux et des éléments sur le terrain (positions, vitesses, produits scalaires entre vitesses…), ainsi que quelques constantes. Ces variables sont construites par le module IA du programme à partir des informations basique que le moteur physique fournit à l'IA, elle sont regroupées dans un tableau qui est mis à jour à chaque pas de temps.
La sous-condition 1 est vérifiée si le poids P1 multiplié par la variable IV1 est supérieur (ou inférieur selon la valeur du bit R1) Au poids P2 multiplié par la variable IV2.
Le même mécanisme est appliqué pour évaluer la sous-condition 2, avec des poids, des variables et une relation différente.
La condition d'application du gène est vérifiée si la valeur du tableau de vérité TV correspondant aux valeurs de la sous-condition 1 et de la sous-condition 2 est vraie.
Par exemple :
La sous-condition 1 peut exprimer « La coordonnée X de mon joueur est supérieur à la vitesse en Y de la balle »
La sous-condition 2 « Le produit scalaire de la vitesse de mon joueur avec la vitesse du joueur adverse est supérieur à 0 »
Si à l'instant où l'IA doit fournir une intention, la sous-condition 1 est vérifiée et la sous-condition 2 est fausse, et que la valeur du tableau de vérité codé dans le gène est :

| **TV** | sous-condition 1 vrai | sous-condition 1 fausse |
|---|---|---|
| sous-condition 2 vrai | **0** | **0** |
| sous-condition 2 fausse | **1** | **0** |

Alors la condition globale du gène est vérifiée, et il contribuera à la construction de l'intention du joueur.

L'intention fournie par le gène est calculée comme suit :
De la même manière que l'on construisait un tableau de 128 variables descriptives du terrain à un instant donné, on construit un tableau de 16 vecteurs descriptifs du terrain à un instant donné (un grand nombre des variables contenues dans le tableau de variables sont issues de décompositions des vecteur du tableau de vecteurs ou de produit scalaires de ces vecteurs).
Un vecteur intention est produit en faisant une somme de 3 vecteurs choisis parmi ce tableau pondérés de poids.
Par exemple, un vecteur d'intention fourni par un gène peut être de la forme :
0.5*(Ma vitesse) + 0.8*(La vitesse de la balle) + 0.2*(Vecteur X unitaire de l'abscisse)

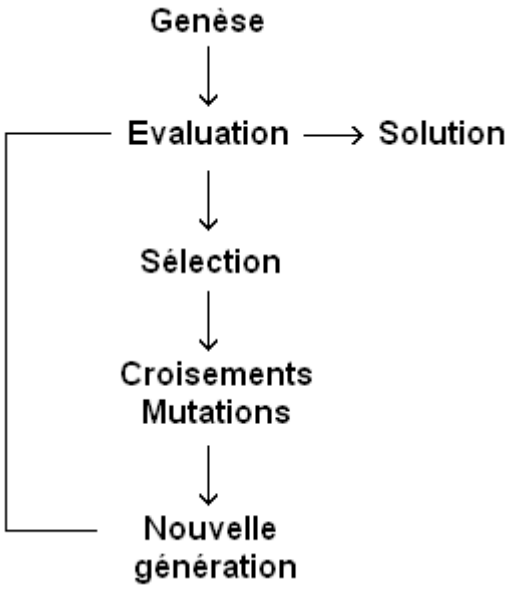
Ainsi si l'on récapitule les exemples cités plus haut, un gène peut avoir une contribution qui se lit de la sorte :
« A chaque fois que La coordonnée X de mon joueur est supérieur à la vitesse en Y de la balle et que Le produit scalaire de la vitesse de mon joueur avec la vitesse du joueur adverse n'est pas supérieur à 0, alors mon intention est de me diriger dans la direction du vecteur 0.5*(Ma vitesse) + 0.8*(La vitesse de la balle) + 0.2*(Vecteur X unitaire de l'abscisse) »

L'intention globale du joueur sera la somme pondérée de l'intention de tous ses gènes. (Dans notre programme un comportement est codé sur 10 gènes).

Nous avons ainsi codé un comportement complexe qui offre à la fois une grande variété de styles de jeu et un codage assez compact de ces styles de jeu. Notons qu'il existe des redondances dans la transcription des génotypes : 2 génotypes différents peuvent donner des comportement similaires (in version des bits de relation R1 et R2 et du tableau de vérité TV par exemple), mais cela ne pose pas de problèmes outre mesure.

## C/ L'Evolution génétique

Reprenons le schéma récapitulatif d'un algorithme génétique, et expliquons la manière dont nous avons réalisé chacune de ces étapes :



**L'évaluation :**
La méthode d'évaluation adaptée est clairement ici celle du tournoi. Plusieurs types de tournois ont été utilisés lors de tests.
En faisant jouer chacune des IA contre toute les autres d'une population et en comptabilisant victoires et défaites, nous pouvons classer une population d'IA selon leur performance de jeu.
En faisant jouer chacune des IA d'une population contre une IA de référence extérieur à la population évolutive, nous classons la population en fonction de l'aptitude des joueurs à battre un adversaire bien particulier.
Un grand nombre de matchs entre chaque joueurs d'une population affine l'évaluation : on pourra plus justement décider de qui est le meilleur de deux joueurs si l'on comptabilise les victoires sur 10 matchs plutôt que si l'on décide sur un unique match. Mais ce n'est pas forcément une bonne chose de trop affiner l'évaluation des joueurs, car il est parfois souhaitable que des joueurs ayant divergé du comportement le plus performant à un moment donné ait une chance de survivre et de continuer à explorer des comportements qui à court termes sont moins performants mais pourront se révéler plus performants après une évolution à long terme.
Nous imposions généralement une part d'aléatoire dans le déroulement des matchs par les positions de départ des joueurs aléatoire, variant autour d'une position type. Cette position type était toujours du mauvais coté de la balle (c'est à dire que l'on plaçait les joueurs entre la balle et le camps dans lequel il devaient pousser la balle) pour complexifier la tâche des joueurs qui devaient faire le tour de la balle plus vite que l'adversaire avant de commencer à pousser cette balle vers leur but.

**La sélection :**
La sélection est effectuée une fois la population évaluée et classée. Nous avons encore une fois expérimenté plusieurs méthodes de sélection. La plus couramment utilisée a été de supprimer le dernier joueur du tournois et de le remplacer par un clone du gagnant (qui existe alors en double dans la population avant son entrée dans les opérations de mutation et de croisement : un de ces clones restera intact et l'autre subira mutations et croisements).

**Mutation et croisements :**
Les croisements et mutations ne sont pas appliquées sur les individus que nous souhaitons gardés identiques (d'où la réplication du vainqueur dans l'opération de sélection pour permettre à son génotype d'entrer en jeu dans ces opérations). Un nombre de mutation définit à l'appel de la méthode de construction d'un nouvelle population (et qui peut donc varier au cours de l'évolution) est appliquée : ces mutations sont des inversions de bits pris aléatoirement dans un génotype.
Le nombre de croisements est également définit à l'appel de la méthode de construction d'une nouvelle population (et peut donc varier au cours de l'évolution). L'opération de croisement que nous avons utilisé n'est pas une opération 'naturelle' de croisement : nous l'avons appelé 'croisement biaisé' car lors du mélange de 2 génotypes pour en créer deux nouveaux qui prendront leur place, le parent le mieux classé à l'évaluation transmet plus d'information.

De manière générale, trouver l'équilibre entre évolution performante et sélective qui implique une uniformisation de la population et évolution laxiste, moins performante mais plus diversifiée, est un vrai casse tête. Il semblerait préférable d'opter pour un mélange entre les deux philosophies, en commençant par une évolution laxiste puis en augmentant les exigences d'efficacités au fur et à mesure de l'évolution.

**Genèse : initialisation de la première population :**
Plusieurs remarques sont intéressantes quand à l'initialisation de la population sur laquelle commencer à applique l'algorithme génétique.
L'augmentation du nombre de joueurs est très coûteuse en temps de calcul, pour un tournoi classique (match allé et retour) le nombre de match à jouer étant proportionnel au carré du nombre de joueurs. Nous avons utilisé des populations dont la taille variait entre 10 et 100 individus.
Partir d'une population de comportement aléatoire est pratiquement impossible : les premier tournois se soldent par des tableaux de points vides, chaque match se finissant en match nuls. Dans ces conditions il est très difficile de faire la différence entre un comportement idiot et un comportement qui commence à devenir 'intelligent' mais n'est pas encore assez performant pour faire la différence. Pour cela nous avons commencé nos évolutions à partir d'une population de comportements capables de gagner des matchs. Ces comportements étaient du type : « si je suis du bon coté de la balle de vais vers la balle, sinon je retourne vers mon camp ». Notons que les comportements obtenus ensuite par évolution battent tous facilement ce comportement de base.

**Fin de l'algorithme :**
Nous avons fait tourner l'algorithme typiquement sur 1000 générations (ce qui selon le nombre de match et le type de tournois, pouvait prendre de 1 à une centaine d'heures).

**Résultats :**
Nous présentons dans la page qui suit une 'bande dessinée' d'une fin de match entre 2 joueurs possédant la même IA. Le joueur bleu, qui essaye de pousser la balle à gauche a réussis, en un temps assez long (environs 20 fois le temps représenté dans la BD) à prendre l'avantage sur le rouge (cet avantage étant ici du aux positions de départ non symétriques des 2 joueurs au début de la partie, car les 2 joueurs ayant les mêmes IA leurs comportements aurait été parfaitement symétrique tout au long du jeu sans cette asymétrie d'origine). Les vecteurs partant du centre des joueurs correspondent à leurs intentions à un instant donné.
Cette représentation n'est pas très convaincante graphiquement, pour avoir une meilleure idée des comportements obtenus je pourrai joindre à ce rapport un exécutable de démonstration.

**Critiques :**
Nous nourrissions l'espoir de voir apparaître des comportements visiblement différents pour deux évolutions différentes partant de la même population. Or il semble que la population converge irrémédiablement vers un comportement type. Ce comportement consiste à se placer derrière la balle et de la pousser vers le but. La position de l'adversaire a généralement peu d'influence sur l'intention d'un joueur, même si nous avons pu voir apparaître ce qui nous a semblé être les prémisses d'un comportement d'anticipation sur la trajectoire de l'adversaire, ce comportement n'est ni assez franc ni assez sophistiqué pour être apparemment bénéfique. Il serait prématuré de l'affirmer, n'ayant pas effectué plus d'une dizaine d'évolutions dont on a fait varier les paramètres, néanmoins il nous semble plausible que notre algorithme ne trouvera jamais de comportement radicalement différent du comportement type. Il est vrai que ce comportement type est plutôt performant, mais il nous laisse un peu sur notre faim. Il aurait été plaisant par exemple de voir émerger un comportement purement défensif ou d'anti-jeu visant à empêcher l'adversaire de marquer plutôt que d'essayer de marquer. Mais l'émergence d'un tel comportement radicalement différent nous semble impossible dans l'état actuel de notre programme : en effet notre population est initialisée à un comportement de base offensif simple. La déviation de ce

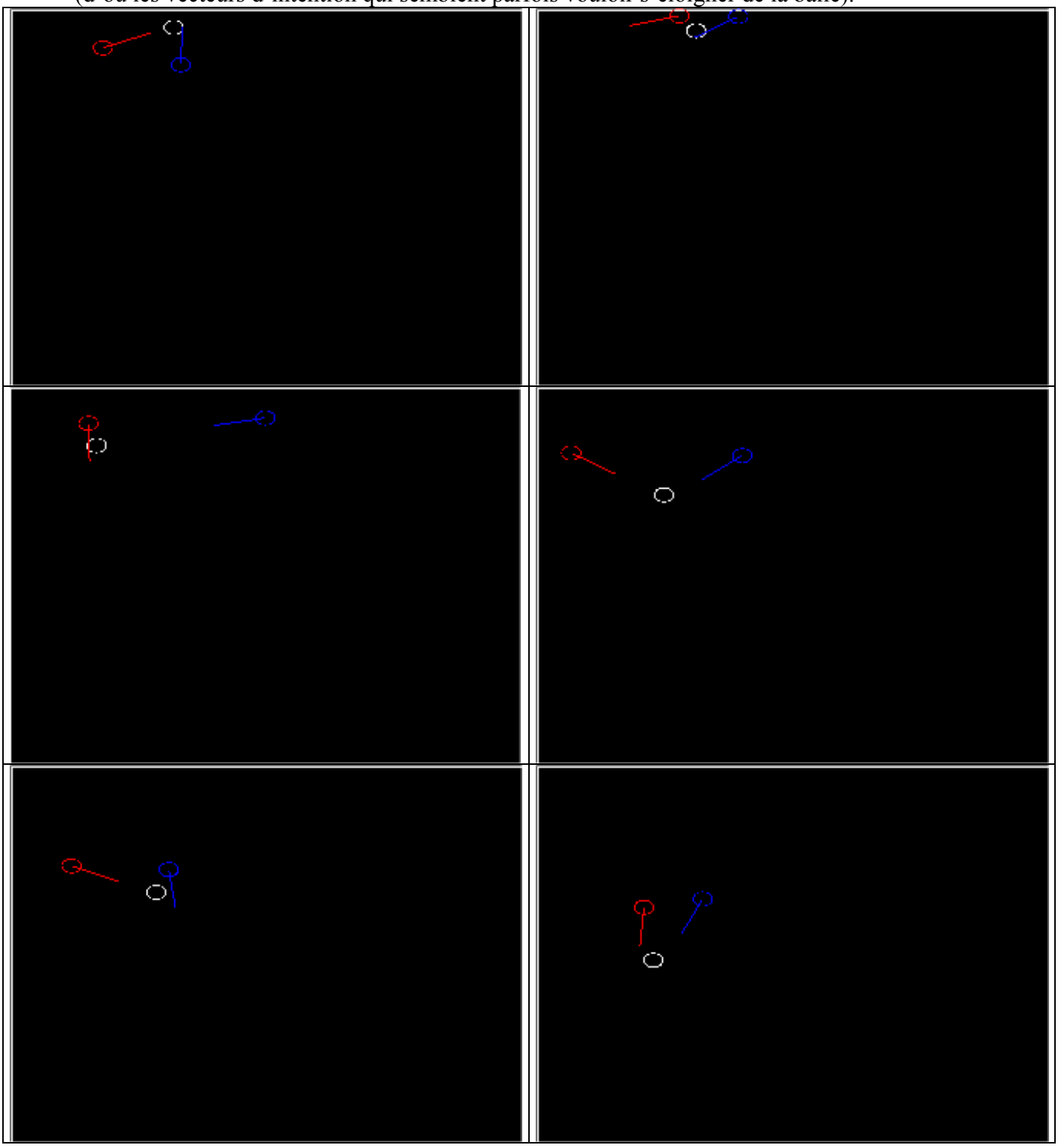
comportement vers un comportement très différent se fait nécessairement en plusieurs mutations, voire en un nombre très grand de mutations qui seront forcément négatives à court terme. Elle seront forcément négatives à court terme car il n'existe pas de 'chemin' de transformations élémentaires menant du comportement offensif de base à un comportement défensif élaboré qui passe uniquement par des comportements intermédiaires assez performants pour survivre. C'est précisément sur ce point que nous voyons les limites de l'application d'un algorithme génétique à notre problème. Evidement ce problème découle en partie du fait que notre initialisation oriente notre convergence, mais comme nous l'avons vu une initialisation aléatoire nous a semblé irréaliste tant l'apparition au seins de la population d'un premier comportement capable d'introduire une différence au niveau des évaluations semble improbable dans une population de comportements générés aléatoirement.

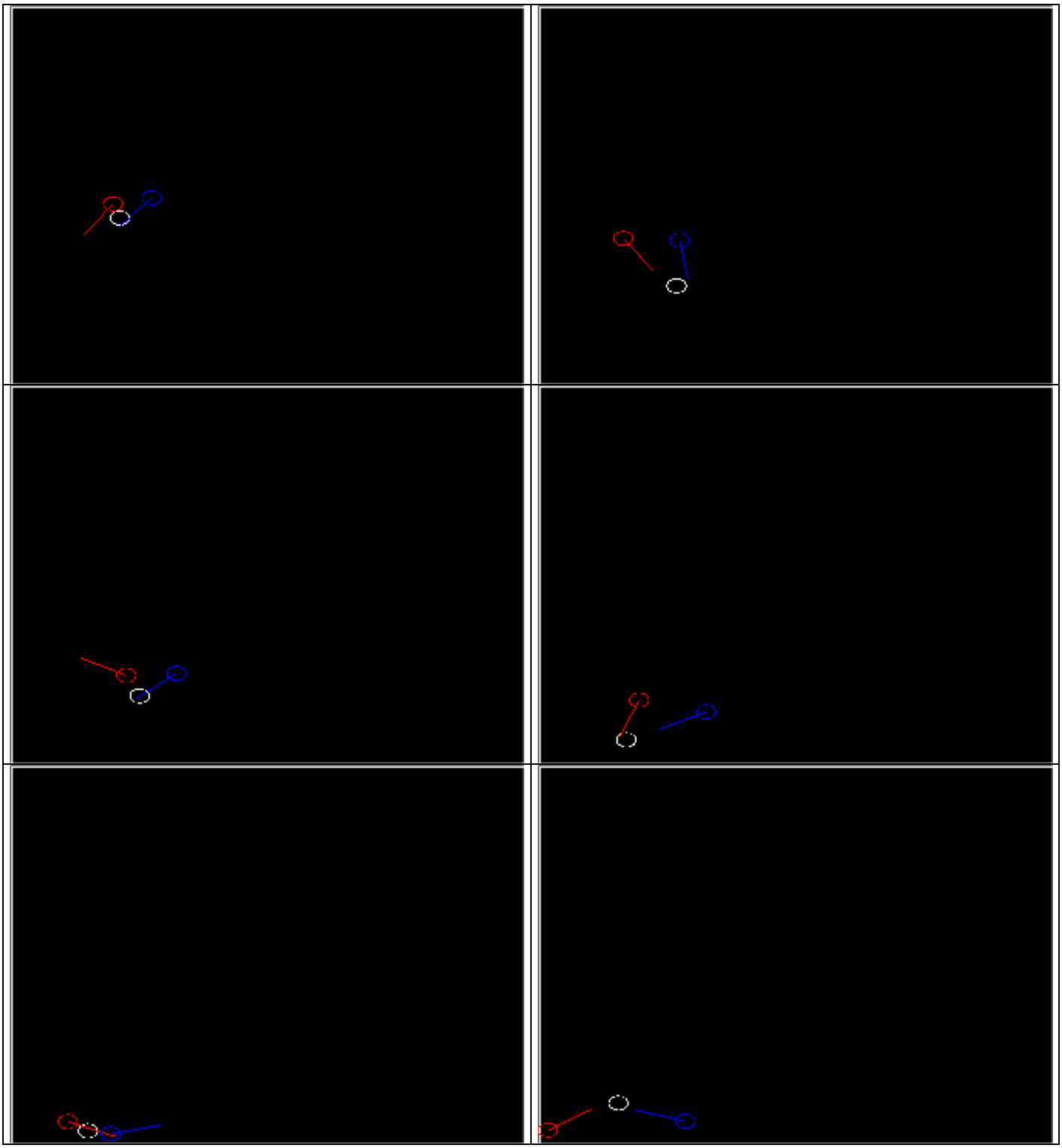
**Développements futurs :**

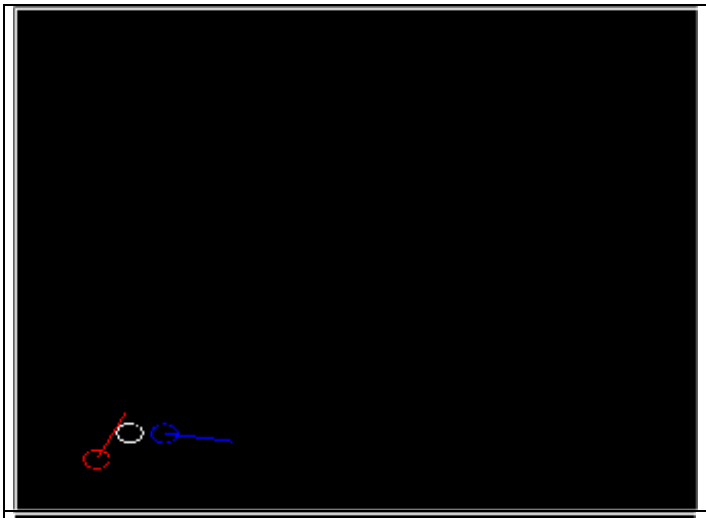
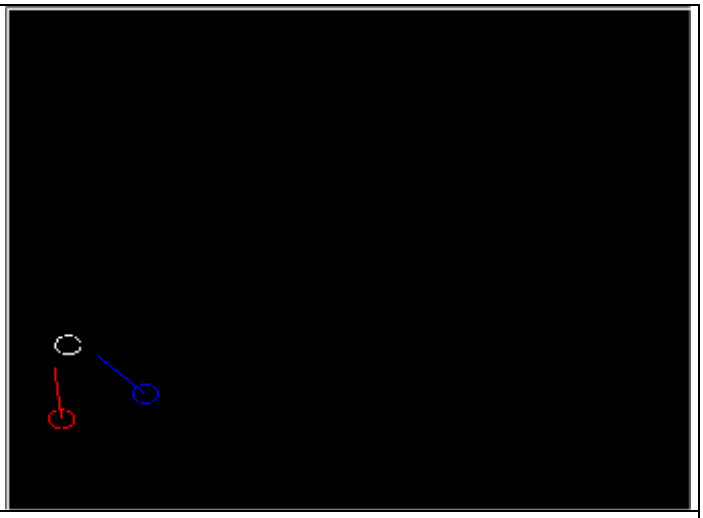
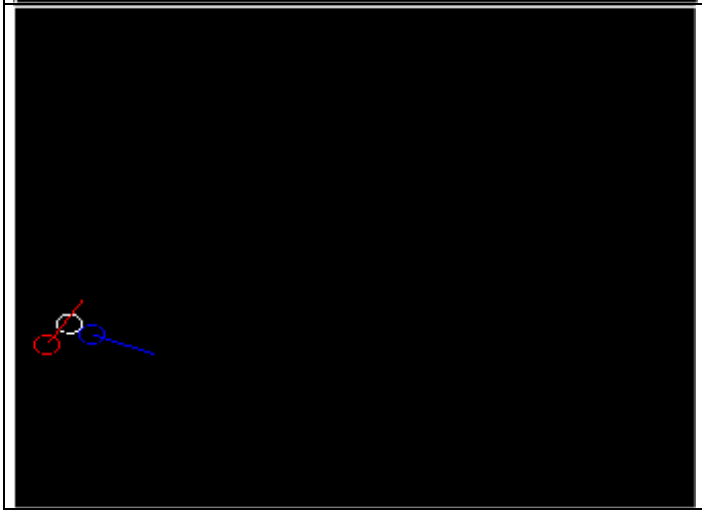
Ce projet fera l'objets de développements futurs, quand les emplois du temps respectifs des développeurs auront des plages disponibles.

Il nous semble intéressant d'essayer de faire émerger des comportement hétérogènes de joueurs au sein d'une équipe : une spécialisation des joueurs à une tâche particulière et complémentaire de celle d'autre joueurs dans une équipe. Pour cela nous allons commencer par faire une version du jeu à 2 joueurs par équipe. Dans une population d'individus, tous les matchs qui opposent deux individus de la population à deux autres individus de la population seront réalisés, et la récompense de la victoire d'un match sera attribuée au deux équipiers. Nous espérons ainsi voir apparaître des comportements 'spécialistes' qui, plutôt que d'être capables de gagner seuls, sont capable de faire gagner un coéquipier : par exemple en gênant de manière efficace le jeu des adversaires...

Exemple d'une phase de jeu (lisible de gauche à droite puis de hauts en bas, captures d'écran toutes les secondes à peu près). Ces images rendent assez mal les aspects 'rebond' des joueurs sur la balle (qui a une masse légèrement inférieur à celle des joueurs mais comparable), ni le fait que les joueurs, sujet à leur inertie 'glissent' sur le terrain et doivent donc dans une certaine mesure compenser ce glissement dans leur intention de déplacement (d'où les vecteurs d'intention qui semblent parfois vouloir s'éloigner de la balle).





| | |
|---|---|
|  |  |
|  | Match gagné par le joueur bleu qui pousse le ballon hors de la limite gauche du terrain. |