Rapport de TP Golf

Semestre 4 - 2022

Dorian Vidal
Lucas Nouhaud
Duthil Thomas

1. Prise en main

1.1 Simulateur

Dans le fichier simulateur, la position actuelle de la balle est stockée dans la variable “ball”, les obstacles quant à eux sont stockés et représentés sous forme d’une matrice. La fonction "get kick target" permet en fonction de l’orientation de balle, de connaître la valeur du sinus et du cosinus. La simulation du déplacement de la balle se fait dans la méthode “kick”, de la ligne 84 à 104, on vérifie pour chaque “pas” de la balle s'il y a un obstacle ou non, quel que soit le résultat, c’est ajouté à la variable “ball_frames”, qui retrace le déplacement total de la balle. Si jamais il y a un obstacle, on vient simuler le rebond de la balle à l'aide de la fonction “hit_test”. Selon l’orientation de la balle lors du rebond, le déplacement se fera plus sur l’axe x ou y. Pour qu’il y ait victoire, il faut que la balle rentre dans le trou à une certaine vitesse. On a la possibilité de décider si l’on souhaite voir le déplacement de la balle pour des raisons de fluidité. Pour l’activer ou le désactiver, il suffit simplement de changer l’état du booléen. La position de la balle est initialisée en 0,0, une fois que la partie commence, on génère une position aléatoire, tout en vérifiant que celle-ci ne se situe pas sur un obstacle, si c’est le cas, on recommence jusqu’à trouver une position convenable.

1.2 Agents

La fonction “*position to state*” permet de convertir la position exacte de la balle en pixel, sous la forme d’un tableau [x,y] en un état sous la forme d’un tuple, qui représente la position “approximative”, permet de simplement dans qu’elle case elle est. La méthode “*build states*”, permet de construire les coordonnées des toutes les cases. Par défaut, l'agent choisit aléatoirement ces actions. Pour choisir, l’agent que l’on souhaite utiliser, il faut dans la méthode “*get_agent*”, décommenter l’agent à utiliser et commenter ce que l’on ne souhaite pas utiliser.

2. Value Iteration

2.1 Compréhension

Les valeurs de la fonction V sont stockées sous la forme d’un dictionnaire, où on associe les cases (states) à des valeurs. On commence d’abord à initialiser toutes les cases à zéro. Ce modèle est représenté à l’aide d’un dictionnaire, qui associe le couple (état,action) à (reward,état final). Le fichier model.data, permet de conserver les données du parcours de golf, cela évite à chaque fois qu’on utilise value itération de tout recalculer.

3. Model-free

3.1 Compréhension

Dans la fonction *pick-action* de la classe *ModelFree*, si on est en mode exploration alors on prend pas la meilleure solution, mais on prend autre chose.Sinon, c'est le même fonctionnement que le pick_action de value itération, on prend la meilleure action. Si jamais on ne trouve pas cette meilleure action, alors on en prend une aléatoirement. Dans la fonction learn, episode = episode[-10:], permet de prendre les 10 derniers épisodes, ceux le plus à droite pour une lecture occidentale. Comme il ne faut au maximum 10 lancer pour atteindre le trou à n'importe quel endroit du parcours on n'en garde que 10.
Les arguments passés en paramètre de la fonction update_q sont , l'état actuel de la balle, son action, la récompense, le prochain état et le return.

On initialise les valeurs à -10 car dans le pire des cas on se retrouve à 10 coups du trou.