

AgentObstacles

Rouge = infos à vérifier

Stratégies :

A chaque instant, vérifier le type du prochain objet devant nous.

Si c'est un obstacle, appeler une méthode qui se charge d'éviter cet obstacle.

Si c'est un bonus ou un nitro, appeler une méthode qui se charge de se diriger vers l'objet.

Éviter l'obstacle :

Si on a un shield, on renvoie le dictionnaire d'actions sans rien changer.

On récupère le steer actuel et on le compare aux coordonnées X du prochain obstacle. Si on se dirige actuellement vers l'obstacle, on réduit ou on augmente le steer d'une petite valeur. Sinon, on ne change rien.

Se diriger vers les bonus :

On récupère le steer actuel et on le compare aux coordonnées X du prochain bonus. Si on se dirige actuellement vers le bonus, on ne change rien et on renvoie le dictionnaire d'actions. Sinon, la coordonnée X du bonus devient notre steer.

Pseudo-codes :

3 méthodes :

- obs_next_item(self, obs, action)
- evite_obstacle(self, obs, action)
- dirige_bonus(self, obs, action)

Ainsi que la méthode choose_action corrigée.

```

obs_next_item(self, obs, action) :
    nextitem = obs["items_type"][0]
    si nextitem = 0 ou 2 ou 3
        #0 : BONUS BOX, 2 : NITRO BIG, 3 : NITRO SMALL
            return self.dirige_bonus(obs, action)
    ou si nextitem = 1 ou 4
        #1 : BANANA, 4 : BUBBLEGUM
            return self.evite_obstacle(obs, action)

evite_obstacle(self, obs, action) :
    si obs["attachment"] = 6 et obs["attachment_time_left"] > 5
        #6 : BUBBLEGUM SHIELD
            return action
    vecitem = obs["items_position"][0]
    si vecitem[2] < 40
        #si on est à une distance raisonnable, pour pas tourner trop
        #tôt
        #vectitem[2] = coordonnée Z
            si abs(vecitem[0] - action["steer"]) < 0.2
                #si on se dirige trop proche de l'obstacle
                action["steer"] = action["steer"] + 0.2
    return action

dirige_bonus(self, obs, action) :
    vecitem = obs["items_position"][0]
    si vecitem[2] < 40
        #si on est à une distance raisonnable, pour pas tourner trop
        #tôt
        si abs(vecitem[0] - action["steer"]) < 0.2
            return action
    sinon :
        action["steer"] = vecitem[0]
    return action

```

AgentItems

Stratégies

A chaque instant, vérifier si on a un item à disposition.

Si on est très proche d'une boîte à items, en général, activer l'item possédé.

Actions pour chaque item :

Bubblegum :

Si on est très proche d'un obstacle, activer le shield et suivre le chemin de base sans dévier l'obstacle.

Si une balle de basket ou une bombe gâteau nous fonce dessus (**si on a l'info**), activer le shield.

Cake :

Si on n'est pas premier et si on peut **l'activer à n'importe quelle distance du joueur devant nous**, alors activer directement.

Sinon, attendre d'atteindre la distance minimale et activer.

Bowling ball :

S'il y a un ennemi en face de nous et que la variation de x par rapport à notre kart n'est pas très élevé, diriger légèrement le kart vers lui et activer.

Vérifier si on peut lancer la balle derrière nous (dans ce cas là on devra s'orienter à l'envers).

Zipper :

Si on est sur une ligne droite et qu'il n'y a aucun obstacle, activer.

Essayer de foncer sur les karts ennemis.

Peut-être plus tard optimiser ça avec des raccourcis ?

Plunger :

Si on a un virage devant nous à la distance d'activation, activer.

Si on a un joueur devant nous à la distance d'activation, diriger légèrement le kart vers le joueur en face et activer.

Switch :

Si on a un avantage (boîte d'item ou nitro) devant nous, NE PAS ACTIVER.

Sinon, activer directement.

Peut-être attendre qu'un joueur ennemi soit en face d'un avantage pour qu'il ne puisse pas l'esquiver ?

Swatter :

Si un ennemi est à moins de 10m de nous, activer.

Rubberball :

Si on n'est pas premier, activer directement.

Parachute :

Si on n'est pas premier, activer directement.

Peut-être attendre qu'on soit troisième ou quatrième pour l'activer...

Pseudo-codes :

Prend en paramètres : self, obs, action

Si obs[“powerup”] égal 0 ou 10 (car l'anvil n'est ni ramassable, ni utilisable), action[“fire”] = 0, fin de la fonction.

Sinon :

```
si obs[“powerup”] = 1 :  
#BUBBLEGUM  
    si obs[“items_type”][0] == 1 ou 4  
        #1 : BANANA, 4 : BUBBLEGUM  
        vec = obs[“items_position”][0]  
        si z du vecteur < 5 mètres  
            action[“fire”] = 1
```

```

        return action

<Si on a l'info qu'un item nous fonce dessus>
si obs[“items_type”][0] == 0 :
#0 : BONUS BOX
    vec = obs[“items_position”][0]
    si z du vecteur < 5 mètres
        action[“fire”] = 1
    return action

si obs[“powerup”] = 2 :
#CAKE
    premier_kart = obs[“karts_position”][0]
    si premier_kart[2] (sa coordonnée z) > 0 :
        #si le premier kart est devant nous, ajouter condition
de la distance du kart devant, voir si le CAKE peut se lancer de
n'importe où
        action[“fire”] = 1
    return action

si obs[“powerup”] = 3 :
#BOWLING
    pour kart dans obs[“karts_position”]
        si kart[z] < 15 et kart[z] > 0 et abs(kart[x]) < 20
            Diriger légèrement le kart vers l'ennemi
            action[“fire”] = 1
        return action
    Regarder si on peut lancer derrière, à remplir ici

si obs[“powerup”] = 4 :
#ZIPPER
    react = #appel fonction analyse d'AgentTurn
    si react = “ligne droite” :
        si obs[“items_type”][0] == 1 ou 4
            #1 : BANANA, 4 : BUBBLEGUM
            vec = obs[“items_position”][0]
            si z du vecteur < 15 mètres
                action[“fire”] = 0

```

```

        return action
    action[“fire”] = 1
    return action

si obs[“powerup”] = 5 :
#PLUNGER
    pour kart dans obs[“karts_position”]
        si kart[z] < RANGE_PLUNGER et abs(kart[x]) < 20
            Diriger légèrement le kart vers l’ennemi
            action[“fire”] = 1
            return action
    #Appel de la fonction d’AgentTurn pour voir s’il y a un
    #virage
    #S’il y a un virage et que distance[z] < RANGE_PLUNGER
        action[“fire”] = 1
        return action
    si obs[“items_type”][0] == 0 :
        #0 : BONUS BOX
        vec = obs[“items_position”][0]
        si z du vecteur < 5 mètres
            action[“fire”] = 1
            return action

si obs[“powerup”] = 6 :
#SWITCH
    si obs[“items_type”][0] = 0 ou 2 ou 3
        #0 : BONUS BOX, 2 : NITRO BIG, 3 : NITRO SMALL
        vec = obs[“items_position”][0]
        si z du vecteur < 10 mètres :
            action[“fire”] = 0
            return action
    action[“fire”] = 1
    return action

si obs[“powerup”] = 7 :
#SWATTER
    pour kart dans obs[“karts_position”]
```

```

        si kart[z] < 20 et kart[z] > 0
            action[“fire”] = 1
            return action
        si obs[“items_type”][0] == 0 :
#0 : BONUS BOX
            vec = obs[“items_position”][0]
            si z du vecteur < 5 mètres
                action[“fire”] = 1
                return action

        si obs[“powerup”] = 8 :
#RUBBERBALL
            premier_kart = obs[“karts_position”][0]
            si premier_kart[2] (sa coordonnée z) > 0 :
#si le premier kart est devant nous
                action[“fire”] = 1
                return action

        si obs[“powerup”] = 9 :
#PARACHUTE
            premier_kart = obs[“karts_position”][0]
            si premier_kart[2] (sa coordonnée z) > 0 :
#si le premier kart est devant nous
                action[“fire”] = 1
                return action

    action[“fire”] = 0
    return action

```