

Documentation des tâches

30/01 :

- **Changer la map de test de notre script. Map : Abysses antédiluviennes**

Affichage des maps disponibles en allant dans le fichier `pystk2_gymnasium/envs.py`. En regardant dans la classe `BaseSTKRaceEnv`, qui est une classe parente à `STKRaceEnv` et `STKRaceMultiEnv`, on voit une variable `TRACKS` initiée au début est une liste vide, puis qu'elle est remplie ligne 167-168 :

```
if not BaseSTKRaceEnv.TRACKS:  
    BaseSTKRaceEnv.TRACKS = self._process.list_tracks()
```

En posant un `print(BaseSTKRaceEnv.TRACKS)` juste après cette ligne, on peut alors accéder aux noms des maps disponibles quand on lance une course.

```
MAPS : ['abyss', 'black_forest', 'candela_city', 'cocoa_temple', 'cornfield_crossin  
g', 'fortmagma', 'gran_paradiso_island', 'hacienda', 'lighthouse', 'mines', 'minigol  
f', 'olivermath', 'ravenbridge_mansion', 'sandtrack', 'scotland', 'snowmountain', 's  
nowtuxpeak', 'stk_enterprise', 'volcano_island', 'xr591', 'zengarden']
```

Pour modifier la map dans notre script de test, il faut se rendre dans `testagent.py` dans le dossier de notre équipe. Il faut chercher dans la définition de la fonction `create_race()` (se situe ligne 97 au moment où je rédige ces lignes), plus précisément les 5 lignes suivantes, où la variable `env` est définie.

Dans la première condition "if" on peut préciser la variable `track` quand on crée notre environnement, ce qui correspond à la map sur laquelle nos agents vont rouler. Par défaut, si on laisse la variable vide alors la map est choisie au hasard. Dans le code qui nous a été donné, la map choisie était "xr591". On remplace par "abyss" qui correspond à la map qu'on a choisie : **Abysses antédiluviennes.**

- **Créer une nouvelle classe `AgentStraight` pour séparer l'agent qui roule tout droit de l'agent de base**

En Python quand on fait de la programmation orientée objet (POO), les classes peuvent être des classes dites “enfant” qui héritent de tous les attributs et toutes les méthodes d’une autre classe dite “parent”. On se sert de ça dans notre code pour définir plusieurs classes `Agent` au fil des modifications qu’on apporte. Ainsi, la dernière classe (`Agent1`) sera la plus aboutie et héritera de toutes les précédentes qui auront des fonctions plus basiques (d’abord rouler tout droit, ensuite suivre la piste au centre, etc...).

On a d’abord créé deux nouvelles classes :

`AgentBase(KartAgent)` : la classe `Agent1`, avant toute modification, héritait d’une classe `KartAgent` qui est une classe importée. Notre classe `AgentBase` correspond à l’agent initial du début du projet, qui roule au hasard sans modifications. On initie dans cette classe toutes les variables dans son `__init__(self, env, path_lookahead=3)` ainsi que les 3 méthodes `reset(self)`, `endOfTrack(self)`, `choose_action(self, obs)`. Ces 3 méthodes pourront donc être utilisées pour les classes qui hériteront d’**`AgentBase`**.

`AgentStraight(AgentBase)` : définition d’un agent qui roule tout droit. Pour ça on ne change rien sauf 2 variables : `action[“acceleration”]` est défini à 1 pour que l’agent roule toujours à fond, `action[“steering”]` est défini à 0 pour que l’agent ne tourne ni à gauche, ni à droite. Ce dictionnaire est défini dans la méthode `choose_action(self, obs)` qu’on a ajouté à nouveau dans cette classe pour faire ce changement. Cette classe contient donc uniquement `choose_action(self, obs)` comme méthode.

A ce stade, notre classe **`Agent1`** hérite de la classe **`AgentStraight`** et n’a aucune méthode ni attribut en plus.

La prochaine étape est de créer une classe **`AgentCenter`** qui héritera d’**`AgentStraight`** et dont le but sera de suivre le centre de la piste.