

Tutoriel de la plateforme de simulation

Nicolas Baskiotis
Sorbonne Université

2019

1 Manipulation élémentaire

1.1 Importer le module

- Importer les objets de base : `from soccersimulator import Vector2D, SoccerState, SoccerAction`
- Importer les objets pour un match :
`from soccersimulator import Simulation, SoccerTeam, Player, show_simu`
- Importer la stratégie de base : `from soccersimulator import Strategy`
- Importer toutes les constantes : `from soccersimulator import settings`
- vous aurez besoin du module math : `import math`

1.2 L'objet `Vector2D`

```
#Coordonnees cartesiennes
v = Vector2D(1.,1.)
# Coordonnees polaires
w = Vector2D(angle=3.14/2,norm=1)
#Vecteur aleatoire
z = Vector2D.create_random(-0.5,0.5)
#ou retire le vecteur aleatoirement
v.random(-1,1)
#Operations usuelles
print(v+w, v-w, 2*v, v/2)
#Manipulation (+, -, *, /)
v += w
v.x += 1
v.norm += 1
v.angle += 1
print(v)
# norme, distance, produit scalaire
print(v.norm, v.distance(w), v.dot(w))
#Normalisation, norme max, mise a l echelle
v.scale(2)
print(v)
v.normalize()
print(v)
print(v.norm_max(0.2))
# Attention, normalize et scale change le vecteur
# norm_max renvoie un autre vecteur
```

1.3 Créer et manipuler un joueur et une équipe

Il suffit de créer les joueurs (nom, stratégie), puis l'équipe en donnant au constructeur le nom de l'équipe et la liste des joueurs.

```
joueur1 = Player("joueur_1", Strategy("base"))
joueur2 = Player("joueur_2", Strategy("base"))
print(joueur1.name, joueur2.strategy, joueur2.name, joueur2.strategy)
team1 = SoccerTeam("Equipe_1", [joueur1, joueur2])
# nombre de joueurs de l equipe
print(team1.nb_players)
# renvoie la liste des noms, la liste des strategies
print(team1.players_name, team1.strategies)
# nom et strategie du premier joueur
print(team1.player_name(0), team1.strategy(0))
```

1.4 Créer un match

```
#Créer un match entre 2 équipes et de durée 2000 pas
team2 = team1.copy()
match = Simulation(team1, team2, 2000)
#Jouer le match (sans le visualiser)
match.start()
#Jouer le match en le visualisant
show_simu(match)
#Attention !! une fois le match joué, la fonction start() permet de faire jouer le replay
# mais pas de relancer le match !!!
# Pour regarder le replay d'un match
show_simu(match)
# Pour réinitialiser un match
match.reset()
#sauver un match, charger un match
dump_jsonz(match)
match = load_jsonz(match)
```

1.5 Créer un tournoi

```
#Pour créer un tournoi d'équipes à 2 joueurs, de durée 2000, avec match retour
tournoi = SoccerTournament(nb_players=2, max_steps=2000, retour=True)
#ajouter une équipe
tournoi.add_team(team1)
tournoi.add_team(team2)
#Jouer un tournoi (lance tous les matchs à la suite)
tournoi.play()
#visualiser un tournoi ou replay d'un tournoi
#show(tournoi)
#afficher les scores
tournoi.format_scores()
tournoi.print_scores()
#Le match entre l'équipe i et j
tournoi.get_match(i,j)
#Tous les matchs de l'équipe i
tournoi.get_matches(i)
#Nombre de matchs, d'équipes
tournoi.nb_matches, tournoi.nb_teams
#Dico des équipes
tournoi.teams
#matchs joués, non joués
tournoi.played, tournoi.not_played
#Sauver un match
dump_jsonz(tournoi, "nom_fichier")
#Charger un match
tournoi = load_jsonz("nom_fichier")
#Réinitialiser un tournoi
tournoi.reset()
```

1.6 Créer une action

Il faut combiner un vecteur vitesse et un vecteur shoot.

```
vitesse = Vector2D(1,1)
shoot = Vector2D(1,1)
act1 = SoccerAction(vitesse, shoot)
act2 = SoccerAction(2*vitesse, 2*shoot)
print(act1.acceleration, act1.shoot)
# Possibilité de combiner les actions :
act_new = act1 + act2
#est équivalent à
act_new = SoccerAction(act1.acceleration+act2.acceleration,
                        act1.shoot+act2.shoot)
```

1.7 Créer une stratégie

Il faut faire hériter Strategy (ou implémenter une méthode compute_strategy)

```
class MaStrategie(Strategy):
    def __init__(self, name="ma_strategie"):
        Strategy.__init__(self, name)
    def compute_strategy(self, state, idteam, idplayer):
        #faire qqe chose d'intelligent
        return SoccerAction(vecteur_acceleration, vecteur_shoot)
```

1.8 Constantes utiles

```
GAME_WIDTH = 150           # Longueur du terrain
GAME_HEIGHT = 90           # Largeur du terrain
GAME_GOAL_HEIGHT = 10      # Largeur des buts
PLAYER_RADIUS = 1.         # Rayon d un joueur
BALL_RADIUS = 0.65         # Rayon de la balle
MAX_GAME_STEPS = 2000      # duree du jeu par defaut
maxPlayerSpeed = 1.         # Vitesse maximale d un joueur
maxPlayerAcceleration = 0.2 # Acceleration maximale
maxBallAcceleration = 5     # Acceleration maximale de la balle
```

Constantes dérivées :

- But de l'équipe 1 : $(0, GAME_HEIGHT/2.) \pm (0, GAME_GOAL_HEIGHT/2.)$
- But de l'équipe 2 : $(GAME_WIDTH, GAME_HEIGHT/2.) \pm (GAME_WIDTH, GAME_GOAL_HEIGHT/2.)$
- Centre du jeu : $(GAME_WIDTH/2., GAME_HEIGHT/2.)$
- Un joueur peut taper le ballon si $d(balle, joueur) < PLAYER_RADIUS + BALL_RADIUS$

2 Manipulation d'un état

2.1 Informations contenues dans un état

```
#Soit state un etat
state.ball           #MobileMixin de la balle
state.ball.position  #Vector2D de la position
state.ball.vitesse   #Vector2D de la vitesse
state.players        # liste : [(idteam, idplayer1), (idteam, idplayer2), ...]
state.player_state(1,0) #Config du 1er joueur de l equipe 1
state.player_state(1,0).position # position du joueur, equivalent a
state.player_state(1,0).position
state.get_score_team(1) # score de l equipe 1
#liste des joueurs en equipe 1
[ (it, ip) for (it, ip) in state.players if it ==1]
state.step           #numero du tour du jeu
```

2.2 Lancer un scénario spécifique

Indications pour pouvoir lancer un scénario, i.e. fixer les composants d'un état et faire dérouler le jeu.

```
#Creer un etat initial de 2 vs 2
state = SoccerState.create_initial_state(2,2)
#Changer la position du joueur (1,0)
state.player(1,0).position = Vector2D(0,0)
state.ball.position = Vector2D(0.5,0.5)
#Appliquer les actions contenues dans le dictionnaire en parametre, chaque
#joueur a pour action la valeur correspondant a sa cle.
state.apply_actions({(1,0) : SoccerAction(Vector2D(1,1),Vector2D()),
                    (1,1) : SoccerAction(Vector2D(),Vector2D())})
```