# **[内部交接]IDM+MOBIL算法模块提取**

# **0. About**

交付至@张健星，将Highway中的IDMVehicle对象中IDM算法部分和MOBIL算法部分提取出来，最终能够整合到oasis driver中。

临时云效仓库地址：[master · idm\_mobil\_agent · Codeup (aliyun.com)](https://codeup.aliyun.com/5f3f374f6207a1a8b17f933f/TayYim/idm_mobil_agent/tree/master)

已添加@张健星为管理员。

全部代码在idm\_mobil.py中，仅有numpy依赖。

注意：

* FakeCarlaActor类是用来测试此脚本的正确性的，在集成到Carla的时候用不到，可以删掉。相应的位置传入Carla Actor对象即可
* 计算前车距离的时候（lane\_distance\_between()）使用的是欧式距离，如果要更准确，需要结合Carla API计算沿着车道的距离
* 这里提供的是加速度决策和是否变道决策。车辆的动力学控制、循迹等，需要基于Carla已有的agent框架

# **1. 概况**

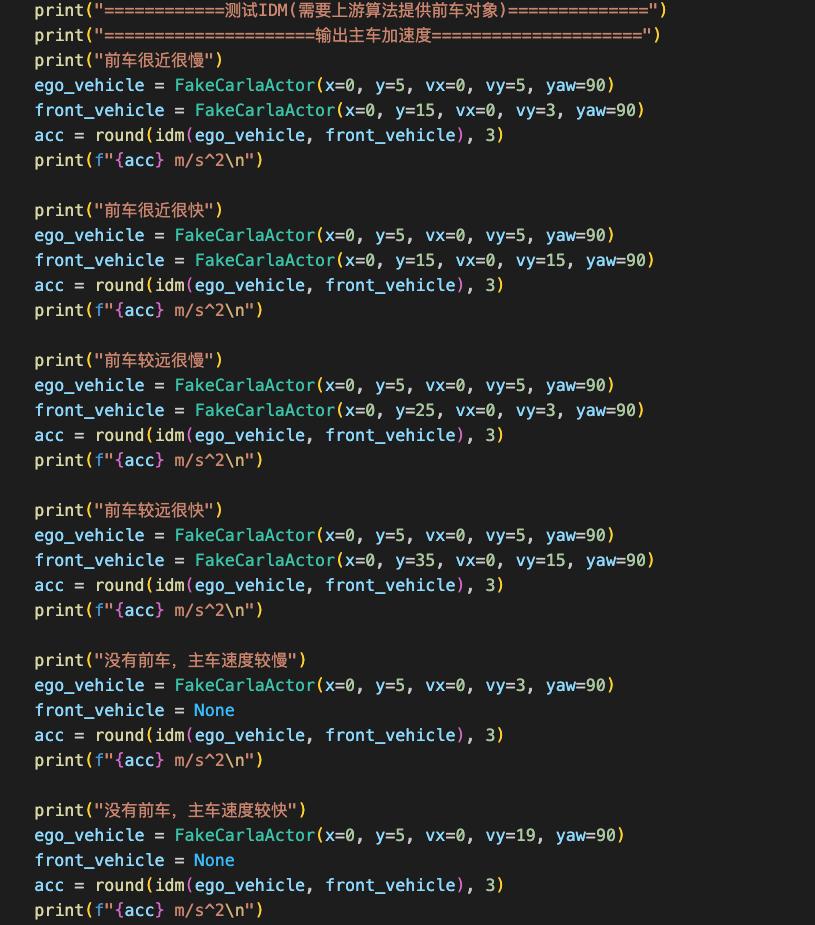
## 1.1 **IDM模块（加速度计算）**

* **输入**为主车对象和前车对象。前车对象需要由上游的算法提供。由于highway中的前车对象是直接取得的，所以此处不会涉及前车的搜索
* **输出**为主车的期望加速度

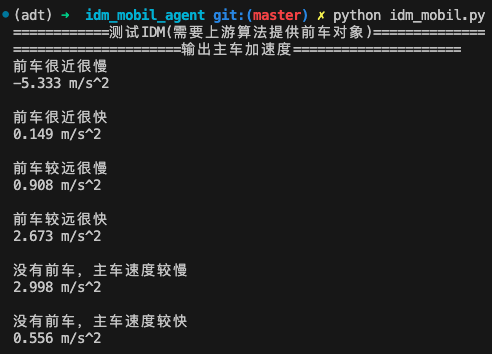
执行idm方法，返回加速度数值：

idm(ego\_vehicle, front\_vehicle)

代码文件中自带了测试用例，可以用来验证：



执行后输出：



## 1.2 **MOBIL模块（变道决策）**

* 输入为主车对象。上游算法需要提供获取主车前、后、左前、左后、右前、右后车辆的方法
* 输出为变道决策。LEFT:向左变道;RIGHT:向右变道;None:不变道

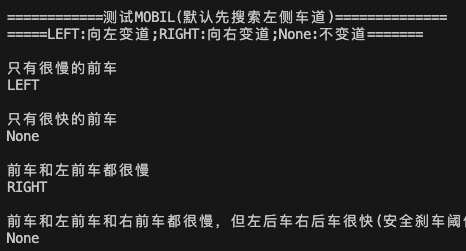
执行change\_lane\_policy方法，返回变道决策：

change\_lane\_policy(ego\_vehicle)

代码文件中自带了测试用例，可以用来验证：



输出：



# **2. 接入需要的后续工程任务**

有一些方法需要结合Carla工程进行实现，在代码中搜索“TODO”即可找到，一共有5个方法。

## 2.1 **找到车辆四周6辆车的方法**

def get\_near\_vehicles(vehicle) -> dict:  
 # TODO 需要结合Carla进行工程实现  
 # 获取车辆周围的6辆车  
 # 分别为左前，左后，左侧，右前，右后，右侧  
 # 如果没有车，则为None  
  
 # 格式如下  
 near\_vehicles = {  
 "P": None, # 前车  
 "F": None, # 后车  
 "P\_L": None, # 左前车  
 "F\_L": None, # 左后车  
 "P\_R": None, # 右前车  
 "F\_R": None, # 右后车  
 }  
  
 return vehicle.near\_vehicles # 实现后这里应该改为 return near\_vehicles

## 2.2 **确认车道存在的方法**

def has\_lane(vehicle, direction):  
 # TODO 需要结合Carla进行工程实现  
 # 判断是否存在这个方向的车道  
 return True

## 2.3 **获取车辆长度的方法**

# 传入Carla Actor对象，获取其对应信息。根据需要修改  
def get\_vehicle\_length(vehicle) -> float:  
 # TODO: 需要Carla工程实现，返回车辆长度  
 return 3

## 2.4 **获取车辆目标车速（仅会用于主车）的方法**

# 传入Carla Actor对象，获取其对应信息。根据需要修改  
def get\_vehicle\_target\_speed(vehicle) -> float:  
 # TODO: 需要Carla工程实现，返回车辆目标车速  
 return 20

## **2.5 获取主车和前车沿车道的距离（可选）**

# 传入Carla Actor对象，获取其对应信息。根据需要修改  
def lane\_distance\_between(ego\_vehicle, front\_vehicle) -> float:  
 # TODO  
 # 需要计算ego\_vehicle到front\_vehicle沿车道的距离，此处使用欧式距离近似代替  
 # 如果要准确，还需要考虑车道形状  
 dist = ego\_vehicle.get\_location().distance(front\_vehicle.get\_location())  
 return dist

## **2.6 油门量映射**

IDM算法输出的是主车加速度。如果要转化成油门量，需要结合Carla进行映射。这块不懂。

acc -> streering

**carla.vehicleackermanncontrol()**

1. **基于IDM优化跟车模型；比较效果;**