惩罚函数

总惩罚函数为

其中为：



主要由下面的不等式得到，其中



，





其中为：



主要由下面的等式得到



（ps: 惩罚函数前的系数川川可根据实际情况设置）

数据格式说明

1. **Input**
   1. Task

**说明：**

对车辆的起点和终点状态（即位置、车辆朝向、速度、车轮偏转角、加速度、角速率）进行限制

**格式：**

[[x0, xtf]; [y0, ytf]; [theta0, thetatf]; [v0, vtf]; [phi0, phitf]; [a0, atf]; [omega0, omegatf]];

**样例：**

[['4', '41'], ['4', '49'], ['0', '0.785398163397448'], ['0', '0'], ['0', '0'], ['0', '0'], ['0', '0']]

1.2 Map

**说明：**

为201\*201的0-1map,1为障碍物填充区域，0为安全区域。

**样例：**

[[ '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', ….], ['0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0', ….],….]

1. **Output**
   1. Traj

一共有60个轨迹点，每个轨迹点下有6个状态，即x，y，theta，v，phi，和tf。

（ps: tf均为车辆的总行驶时间, 其为相同的数，每个时刻下的相邻时间为delat\_t=tf/60）

**格式：**

[[x(1), y(1), theta(1), v(1), phi(1), tf],[ x(2), y(2), theta(2), v(1), phi(2), tf],[ x(3), y(3), theta(3), v(3), phi(3), tf],…, [x(60), y(60), theta(60), v(60), phi(60), tf]]

**样例：**

[['4', '4', '0', '0', '0', '133.957'], ['4', '4', '1.781e-16', '0.030', '0.020','133.957'], ['4.031', '4', '0.0002', '0.060', '0.041', '133.957'], ['4.094', '4.000', '0.001', '0.0911686047391166', '0.061', '133.957'], …]

程序说明

1. **Test.py**

**说明：**

Csv文件的读取，读取map,task,traj

1. **Visualize\_result.py**

**说明：**

通过可视化轨迹规划的结果，测试学习到的轨迹是否有效。其中可通过改变Globalvar.py中的vehicle\_TPBV\_Set来改变车辆的起止状态限制，通过改变Nobs来改变生成的障碍物，障碍物为多边形构成，costmap函数将生成障碍物转为了0-1map.

程序中需要川川补充trajectory。