

自己动手写全套无人驾驶算法系列（一）机器人控制

一、概述

1.1 系列整体概述：

本文介绍无人驾驶的主要几大控制算法，PID, PID+Review, MPC,按理算还应该有 stanley control, LQR 这些，但是目前类似扫地机器人这些，用单纯 PID 就够了，而对于无人驾驶汽车，百度主要也是 PID 和 MPC，因为狭窄路段，倒车入库等 PID 就可以，而高速公路，跟车，车道保持等 MPC 就可以，所以关键不在多而在适用，stanley 是好算法，有时间会写，但目前人们大多用 MPC 而不用 LQR，因为 LQR 实际效果大都不如 MPC。另外本人自己主要就是做 MPC，能实现无人汽车下旋转停车场，倒车入库，上旋转坡，车道保持等等，我在 github 给出的是仿真模式，没有加入优化和逻辑，这个可以自己加，怎么加见下文和我的 github：
https://github.com/wujiazheng2020/WJZ_Controller

欢迎大家对我的 github 项目提出 BUG，或者需要我更新新的算法,或者需要技术合作，都可以联系我：

163 邮箱：wujiazheng2020@163.com
gmail:wujiazheng2020@gmail.com
QQ 群：710805413

1.2 控制算法实际运用问题

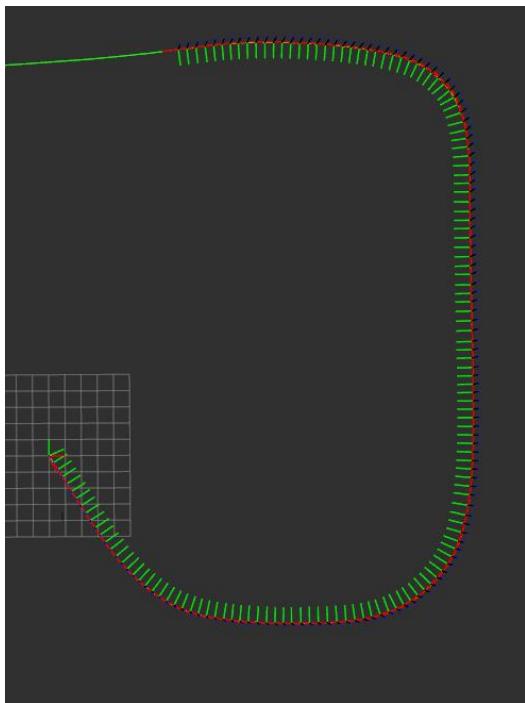
实际运用控制算法不像仿真那么简单，时间上还需要如下步骤：

- 1.时间同步：设定一个可信时间区间，对早到的时间，要按运动学补偿，迟到超过一定时间的数据，直接按运动学推算，见我的 github。
- 2.倒车逻辑：如航向角为负的 Trick, steer 问题
- 3.换档逻辑：如何停下来，如何判断停稳再换档，换档如何判断换档成功
- 4.底盘逻辑：如 USB 不允许方向盘一次转超过 25 度，加速度不能变化超过 5m/s² 等，要在 MPC 的凸优化里开盒函数，输出的时候也要限制。
- 5.轨迹处理逻辑：读取轨迹的时间同步，轨迹如果不适合三次曲线怎么办，终点前最后一段轨迹点不足怎么办，轨迹噪声太大怎么办。
- 6.临时停车逻辑：如何临时停车而不完全停下
- 7.紧急制动逻辑：突发情况处理
- 8.其他忘记说的逻辑。

一、算法介绍

注：本程序所用轨迹为从停车场-4 上升到-2 曾的轨迹，图片均来自我的 github。

2.1 PID_DIFF

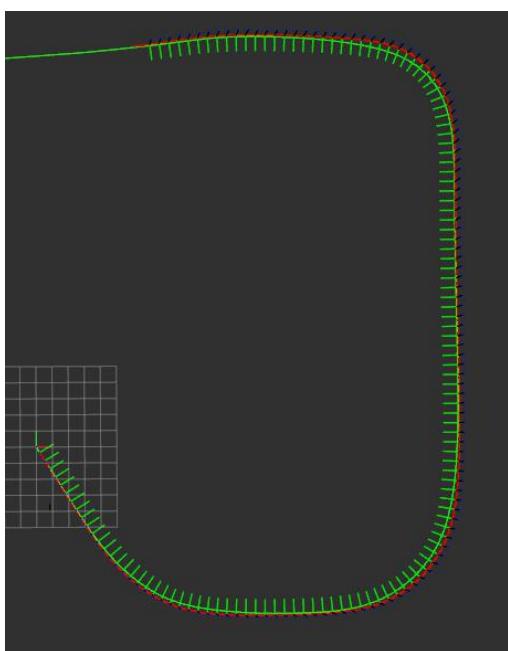


主要运用于如扫地机器人的差分驱动模型,实际上很多人说要双环 PID,但是就实际测试情况来看,单环这种对大部分机器人够了,而双环机器人,大部分厂家的生产标准不能到大公司的标准,换了机器人要重调试很麻烦,实际过程中对大型机器人,PID 需要一些 trick,比如停止时,轨迹处理这些,这些不属于算法内容,可以自己去探索。

选取距离为当前最近点索引为 k 的点作为 target 点,以对其的距离,角度差为误差进行 PID 控制

控制 V , W , PID 是利用比例,积分,微分进行控制的算法,算法本身具体可以见一文读懂 PID 控制

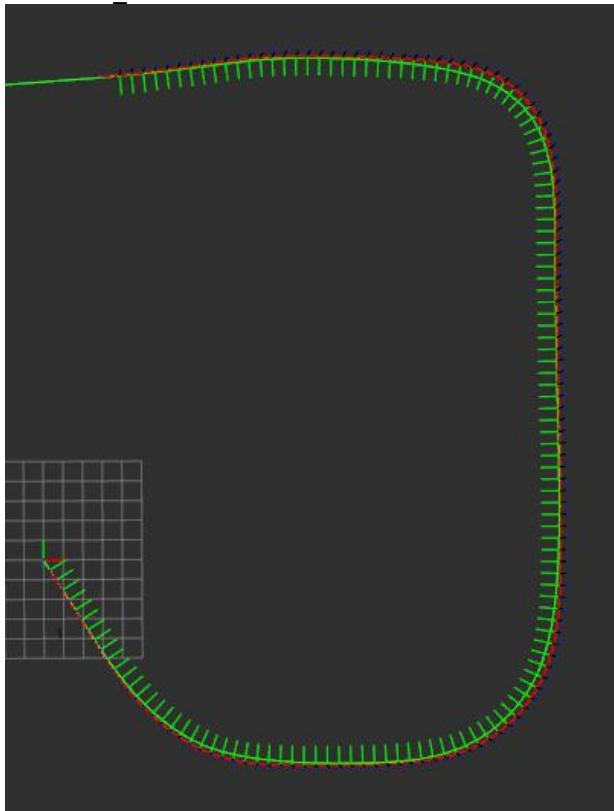
2.2 PID_ACM



主要运用于如无人车等阿克曼转向模型,阿克曼转向模型见我系列第二篇的介绍,
老实说实际过程中,做成双环效果似乎不如单环,但大家可以自行设置再加一环,会单环加环也很容易,对无人车,需要一些 trick 才能实际运行,比如时间不同,停车处理,各种逻辑的处理,这些不属于算法了,可以自行加上。

本算法利用了预瞄(preview)技巧,预瞄前面 N 个点的航向角,可以让无人车控制更稳定:
采用角度差和预瞄作为角度差,其他和上个 PID 一样。

2.3 MPC ACM



主要运用于如无人车等阿克曼转向模型,我曾经有一段时间就是专门做 MPC,可以实现下停车场并倒车入库,上旋转的坡,车道保持跟随各种,但是 MPC 运用时是要考虑时间同步的,时间对 MPC 特别重要,因为 MPC 是等时间间隔预测的,另外需要一些轨迹处理,很多停车,倒车逻辑,处理规划路线逻辑,这些根据不同车有不同处理方法,不属于算法本身,可以自己探索。

全称是模型预测控制,就是给定车前方轨迹 N 个点,时间间隔为 dt ,故总时间是 $N \times dt$,那么我们按运动学方程,以不同的 $a[N]$ 和 $steer[N]$ (表示 a 和 $steer$ 的数组, $a[6]$ 就是表示 $t=6 \times dt$ 时刻的给的加速度) 推算出到相应时间的 x , y , 横向误差, 纵向误差, a , 航向角等状态,那么计算 x, y , 横向等与轨迹相应点的差,得到一个 cost function,其中 $a[N]$ 和 $steer[N]$ 都是待优化量,之后我们便可以按 IPOPT 进行凸优化迭代即可,当然了,我们要对 a 的变化率, $steer$ 变化率, 速度大小, 加速度大小等进行约束。

最后我们就得到一个最优的 $a, steer$ 序列,取 $a[0], steer[0]$ 作为当前输出即可。正常行驶效果是各大控制算法最好。