目录

[1.为什么速度环输出能作为角度环输入？2. 怎么把速度转换成角度的（量纲是如何转换的）？ 2](#_Toc117202275)

[能不能先调整速度环？ 2](#_Toc117202276)

[为什么Encoder.c里面写了中断服务函数且没使用？这里中断服务函数必须加吗？ 3](#_Toc117202277)

[速度读取不合理 3](#_Toc117202278)

[为什么限幅函数的形参使用指针？为什么赋值加载函数不使用指针？ 3](#_Toc117202279)

[形参若不使用指针，那么执行流程就是这样的： 3](#_Toc117202280)

[形参如果使用指针，那么执行流程就是这样的： 4](#_Toc117202281)

[调参流程（通俗详细，与视频里略有不同之处，这个版本更方便与合理） 5](#_Toc117202282)

[一、角度环 5](#_Toc117202283)

[【角度环Kp】 5](#_Toc117202284)

[【角度环Kd】 6](#_Toc117202285)

[二、速度环 6](#_Toc117202286)

[Kp&Ki的极性调节 7](#_Toc117202287)

[Kp&Ki的大小调节 7](#_Toc117202288)

[三、转向环 9](#_Toc117202289)

[Kp极性： 9](#_Toc117202290)

[Kp大小： 9](#_Toc117202291)

# 1.为什么速度环输出能作为角度环输入？2. 怎么把速度转换成角度的（量纲是如何转换的）？

速度环的被控量是速度，角度环的被控量是角度。

**对于第一问，**因为我们的控制系统就是这么搭建的，控制框图就是这么画的，我们在按照控制框架去写控制程序。而这个控制框图，它综合了速度与角度两个被控量，满足使得车子直立在原地的标准。

事实上，你也可以将速度环与角度环分别计算，并把两者的输出线性叠加（也就是我们常说的并级PID），然后按照并级PID去编程。

视频没有选择双环并联，是因为在担心动力不足的情况发生（即双环的输出叠加后远大于PWM上限，此时就会由于动力不足，导致发散或意外情况发生），事实上串级同样会有动力不足的风险。

**对于第二问，**量纲转换，是靠内、外环的pid参数来进行转换的。

通俗一点就是说，通过调整pid参数，将速度环（外环）的速度控制量输出的变化范围，归一化到角度环（内环）的角度变化范围之间。

这个行为我可以称之为归一化，也可以称之为坐标系变换（将一个矢量从速度系旋转到角度系），怎么称呼都可以，重点在于理解其含义。

# 能不能先调整速度环？

不可以先速度环。要明白扰动更容易出现在哪个被控量里，以及哪个被控量更重要。

如果先调速度环，速度环只能约束车轮的速度，对车体的角度没有控制。相当于什么呢，相当于说，我只需要速度为0就好了，角度是不是0无所谓；那你角度不为0，车子必然马上倒；

而先调角度环的话，此时角度受控，速度没有去约束，这时候最多车子轮子一直跑罢了。最差的情况，也是会在维持几秒直立之后，由于执行机构反应慢或偏离平衡状态过大等多方面因素，导致系统发散车子倒下。

# 为什么Encoder.c里面写了中断服务函数且没使用？这里中断服务函数必须加吗？

Encoder.c的中断服务函数可以不加。视频里加了的理由是：因为初始化里开启了中断，所以就多写了一个中断服务函数以便于中断寻址。事实上，在这里中断也并不用开。

# 速度读取不合理

确实不合理，差分求导准确上应该是：

**S = [P(k)-P(k-1)]/dT**

其中，S是速度，P是位置，k表示第k个周期，P(k)表示第k时刻的位置量，dT表示采样周期。

比如视频中的采样周期是5ms，就有dT=0.005，而视频中将dT直接简化为了1。

将dT简化为1的原因是：结合编码器线数、采样周期等因素，并经过测试发现：若采用dT=0.005，采样出的速度数据的将会非常小（速度慢的情况下速度都是小于0的浮点数，即小数级），此时速度数据的量程太小不方便计算，并且我们这里秉承尽量不去做浮点运算的原则，所以需要解决该问题，此时的解决办法有三个：

1. 对速度数据不做处理，提高速度闭环的PID参数量程，比如将参数放大10倍、100倍等。
2. 对速度数据直接进行比例放大，比如放大100倍。

视频里将dT=1，相当于使用了第二种方法，将速度数据放大了200倍（1/0.005=200）。

# 为什么限幅函数的形参使用指针？为什么赋值加载函数不使用指针？

## 形参若不使用指针，那么执行流程就是这样的：

调用时传入**Limit(MOTO1,MOTO2);**

此时，限幅函数写为

文本

描述已自动生成

此时执行流程为：我先传入**MOTO1与MOTO2**进来，然后Limit()函数在栈区重新创建两个变量，分别叫motoA与motoB，然后在后续的Limit()函数中操作的，都是motoA与motoB这俩变量，完事Limit()函数执行完了后，motoA与motoB就又被释放了。你看，MOTO1与MOTO2全程都没有参与运算，但我们传入它两个的目的就是要操作MOTO1与MOTO2这俩啊！

所以我们选择形参使用指针变量。

## 形参如果使用指针，那么执行流程就是这样的：

调用时实参传入的是MOTO1与MOTO2地址：**Limit(&MOTO1,&MOTO2);**

文本

描述已自动生成

文本

低可信度描述已自动生成

此时执行流程是：Limit()函数在栈区创建两个指针变量，然后我传入MOTO1与MOTO2的地址进来，形参的指针motoA保存了MOTO1的地址，指针motoB保存了MOTO2的地址。此时在Limit()函数体内部，“\*motoA”与“\*motoB”操作叫解引用，并有：“\*motoA”等价于MOTO1，“\*motoB”等价于MOTO2。

也就是说，每个“\*motoA”你都能看成是MOTO1，每个“\*motoB”你都能看成是MOTO2。

这样，Limit()函数执行完后，虽然motoA与motoB一样会被释放，但MOTO1与MOTO2此时已经被操作过了。

所以，形参使用指针变量的原因是：**我们要修改所被传入的变量的值**。

那为什么赋值加载函数不需要使用指针呢？

我们再看一遍：形参使用指针变量的原因是：**我们要修改所被传入的变量的值**。

文本

描述已自动生成

你看，在赋值加载函数这里，我们不需要修改所被传入的变量的值啊！

形参moto1与moto2只负责把PWM传进来，然后就用不到他俩了，我们并没有希望去修改他俩的值，所以这里的形参我们不需要使用指针变量。OK。

# 调参流程（通俗详细，与视频里略有不同之处，这个版本更方便与合理）

## 一、角度环

**注意：**下文会出现类似**|Kp|**，表示绝对值，其中的参数可能为正，可能为负，但他们的绝对值一定是越来越大的。这里我无法确定你的为正还是负，需要你自己测试，所以使用了绝对值。所以说下文中出现的譬如“随着**|**Kd**|**增大”这句话就代表“Kd越来越正”或者“Kd越来越负”。

### 【角度环Kp】

#### 调Kp参数极性：

电机转向能使车子呈直立趋势，比如车子向前倒下时，车轮此时会向前旋转，使得车子有回正直立的趋势，就说明Kp极性正确。

那么反之，若车子向前倒下时，车轮此时会向后旋转，使得车子更趋向于倒下，就代表此时Kp极性反了。

#### 调Kp参数大小：

增大**|**Kp**|**，并用手扶着车子在平衡位置处，观察现象，直至车子在平衡位置处出现来回大幅低频振荡（出现过一次即可，不要求一直振荡，因为事实上单角度环是无法保持直立的），来回大幅低频振荡的曲线表示如图1。

**调好的现象：**有短暂几秒的直立趋势，并且在平衡位置处可能会出现大幅低频振荡。

图表, 折线图

描述已自动生成

图1 大幅低频振荡

### 【角度环Kd】

#### 调Kd参数极性的方法：

1. 保持Kp大小（**无需将Kp=0**）。
2. 随意给Kd一个初值，比如我随便给个1，然后看现象。

##### Kd极性错误的现象：

随着Kd越来越正，车子直立的效果越来越差（控制变得越来越软，车子越来越立不住），甚至直立效果明显不如单kp。这就说明Kd=1是极性反了。

##### Kd极性正确的现象：

给Kd=-1，随着Kd越来越负，车子的控制越来越硬，甚至出现“在平衡位置高频抖动震荡”，说明Kd=-1极性正确。

（极性正确的判定方法二：能减弱或消除由单Kp造成的低频振荡，也说明极性正确。）

#### Kd参数大小的调节方法：

随着**|**Kd**|**增大，车子由单Kp造成的低频抖动消失，且明显控制越来越硬，增大至车子出现些许高频振荡即可。

**调好的现象：**能维持短暂几秒的直立，车子比单Kp硬一些（但车轮转速与位移没被约束，故只能短暂直立、最终还是会倒下），并且在平衡位置处可能会出现高频振荡。

## 二、速度环

注意：

1. 将角度环的参数乘以0.6。
2. 注意：**无需注释角度环。**

### Kp&Ki的极性调节

随意给一套Kp&Ki，刚开始使Kp：Ki比例呈1：200，比如我随意给个Kp=10，Ki=0.05，然后看现象。

#### Kp&Ki极性错误的现象：

随着**|**Kp&Ki**|**增大，直立效果越来越差（控制越来越软），甚至不如单直立环的直立效果。

#### Kp&Ki极性正确的现象：

随着**|**Kp&Ki**|**增大，直立效果会变好，明显比单直立环的效果好。（注意速度环的参数不宜过大）

注意：“随着**|**Kp&Ki**|**增大”表示“越来越正”或者表示“越来越负”，具体表示哪个取决于你的参数的极性正负。

### Kp&Ki的大小调节

#### 第一步：Kp&Ki以1：200的比例联合调节。

先将将Kp&Ki以1：200的比例，联合调参，小车调参至差不多能直立在原地，来回振荡没关系，只需要能直立在原地，不会倒下且不会向一边倒下，就可以了。

即：增加Kp&Ki直至小车保持平衡的同时速度接近于零，且回位效果较好。这一步很简单。

#### 第二步：Kp与Ki分别调节

其实在视频里，第一步就结束了。但为了获得更好的效果，可以继续进行第二步的细调。

注意，第二步这里，Kp与Ki不再是1：200的比例关系。下面会告诉你，将他俩分别单独调，如何做效果会更好。

因为公式是：

再结合实际的测试，我们可以得到如下现象：

##### 【速度环Kp】

**Kp越小**：回复**大幅**震荡越严重（对速度的约束力越小），这里要理解透，见图2（画的不好请担待）.

图示

描述已自动生成

图2 速度环Kp越小

**Kp越大**：回复答复振荡越轻（速度约束强），且回复到平衡位置后，微幅高频振荡越明显。见图3.

图示

描述已自动生成

图3 速度环Kp越大

所以，Kp越大性能指标越好，但太大会使得车子在平衡位置高频振荡越明显，注意把控。

##### 【速度环Ki】

**Ki越大**：推开同样的距离，车体倾斜角度越大，相对的回复到原地的越迅速，参考下图：

图示

描述已自动生成

图4 速度环Ki越大

**Ki越小：**推开后回复到原始位置的力越小，即下图中，Ki越小， T1时间会越短且T2时间会越长。如图。

信件

描述已自动生成

图4 速度环Ki越小

由此，Ki越大稳态指标越好，但Ki过大会使得系统发散，这里**提示Ki不要太大**。

**注意：**

Kp&Ki不一定是1：200的关系，不过极性肯定是相同的。说个例子，我记得自己调的最好的效果时，Kp=-270，Ki=-200（只是举例子，每个人的车子不同，参数也不同，所以该套参数没有任何参考性）。

## 三、转向环

### Kp极性：

**极性错误：**拿起小车，并将小车绕Z轴旋转，两车轮旋转的趋势与小车旋转趋势一致——典型正反馈。

**极性正确**：拿起小车，并将小车绕Z轴旋转，两车轮旋转的趋势与小车旋转趋势相反——典型负反馈。极性正确。

### Kp大小：

加大Kp，直至走直线效果较好，且无剧烈抖动。

**哪里看不懂，或者在经过深入思考后仍觉得不理解或不正确，可以群里和群友或艾特天下行走讨论。**

给笔记截个整体的图，想看的同学可以看看。

手机屏幕截图

描述已自动生成