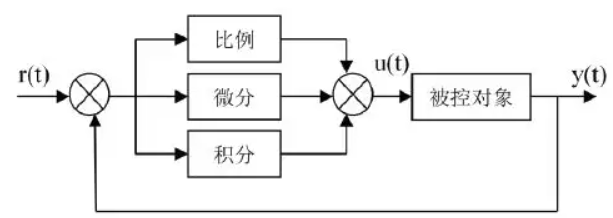
直流电机测速调速2

现在的你想必已经能灵活运用光电码盘和Arduino主板监测手中小车的速度了，但也许你已经发现串口通讯传来的速度好像与你期望的速度不尽相同，甚至相去甚远。通过合理的手段控制让实际速度与预期相仿，一直是研究者的重点关注对象。这里，我们分享一种最为常见的调速方式：PID控制。

说起PID控制，大家肯定不会陌生，作为一种原理简单、鲁棒性强且技术成熟的控制系统，它已经广泛地应用于各类产品上。无论是四轴飞行器，还是自平衡小车，都难免需要应用到PID控制。



PID控制框图

**PID控制简介**

PID控制，即指具有比例+积分+微分控制规律的控制。简单来说，即通过三者的配合来减少稳态误差，并使直流电机速度（在轮式机器人例子下）尽快、尽稳地达到期望值。

在形象地讲述PID方面，“前人之述备矣”，机协百科中有很生动的例子。这里我们不妨再简单复述一下：

任务目标：罗伯特需要不断为漏水水缸补水使水位稳定在100L。

采样周期T：如果10分钟才补一次水，水位则会大起大落；如果10秒补一次，罗伯特则腰酸背痛。因此，罗伯特发现，1分钟补充一次水最为合适，1分钟则为采样周期T。

偏差：每个采样周期后罗伯特所观察到的真实水位值与目标值（100L）之差。

**PID之P——比例控制：**

得到偏差后每次罗伯特究竟应该补充多少水呢？

比例系数Kp：通过总结发现，罗伯特每次补充偏差的1.5倍，能够使水位以较快速度接近目标值。这个1.5即为比例系数Kp。

仅用比例控制就可以了吗？并不尽然。

反例：假设每个采样周期漏10L水，Kp设置为0.5，而初始水量为80L。

T=1时：u[1]=0.5\*(100-80)=10L 水位变为90L=80L+u[1]

1-2时间内，因漏水，水位又会变为80L=90L-10L

T=2时：u[2]=0.5\*(100-80)=10L 水位变为90L=80L+u[2]

如此以来，水位将会长期低于目标值。

**PID之I——积分控制：**

为了消除水位的持续单向变化引起的累计误差，罗伯特统计了前几次的偏差量，并对补水量进行了微调，这即是PID的积分控制。

把前几次的偏差做和，再乘上采样周期，即为离散条件下求积分和的过程。

积分时间常数Ti：罗伯特发现，用这个积分和除以3个采样周期得到的“平均偏差”去修正比例控制中的偏差，再乘上1.5（比例系数Kp），水位就能在100L附近较稳定地波动了。这个3个采样周期（3T），即为积分时间常数。积分时间常数调节积分换节的敏感度：积分时间常数越大，积分环节就越不敏感，罗伯特在比例控制基础上的补水量就越少。

积分比例系数Ki：为表示方便，我们将Kp\*T/Ti用Ki表示。

**PID之D——微分控制：**

看到水位基本维持了平衡，罗伯特松了口气，靠在水缸旁休息。突然，他发现倚靠水缸产生的振动让水在短时间突然漏得更快了，而随着振动的消除，水位并没能快速回升。

怎么解决诸如振动的外界干扰呢？

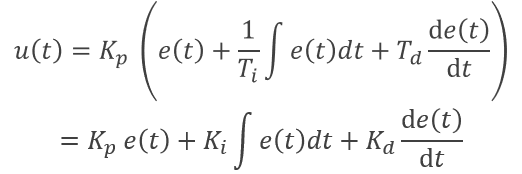
罗伯特认为，可以在水位突变的瞬间就及时制止振动的干扰，即通过计算水位下降加速度来调整，这就是微分控制。

微分控制的系数与积分控制类同，此处就不赘述了。

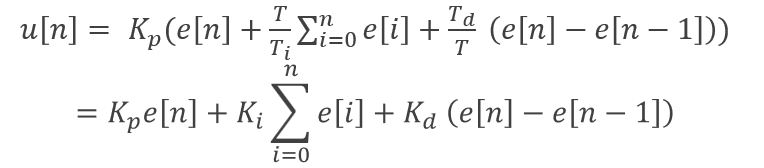
最终，罗伯特通过总结漏水水缸的控制经验，将PID的控制方法归纳成公式托机小协呈现给大家。

**PID连续和离散情况下的微分方程分别如下（位置式PID）：**

**连续：**



**离散：**



其中：

u(t)为控制器输出的控制量（上述例子为补水量，直流电机调速中为输出速度）；

e(t)为输入偏差信号（期望速度与编码器所测速度之差）；

T为采样周期（即得到相邻两次误差所需时间）；

Ti为积分时间常数，Ki为积分比例系数；

Td为微分时间常数，Kd为微分比例系数。

值得注意的是，PID的输入输出的物理量是由被控系统决定的，并不一定单位一致，PID输入是系统输出和参考输入的差值（误差，即期望速度与编码器速度之差），PID输出则是控制系统的输入。（建议结合控制流程图理解）

**PID控制例程实现**

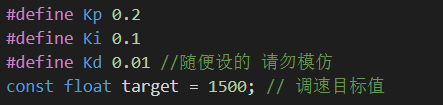
既然已经知道PID控制的原理，想要用程序实现也就并不那么困难了。

在例程中，我们使用编码器的采样周期作为PID控制的采样周期。

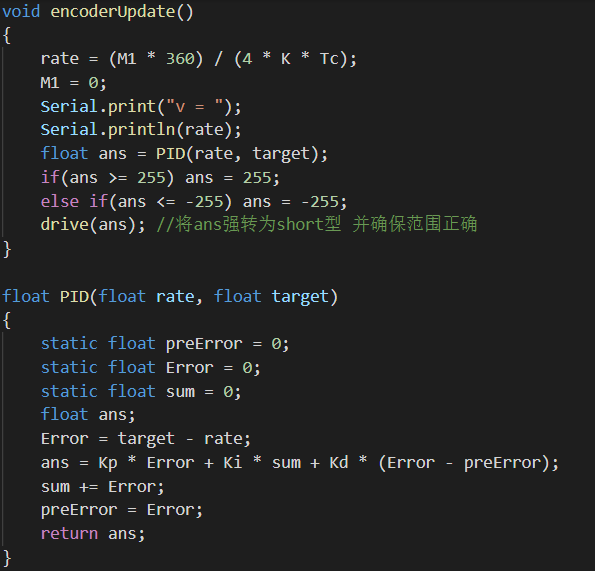
我们也并不需要数组来记录每次的偏差，只需使用sum来统计偏差和进行积分控制，并记录当前偏差及上一次偏差协助微分控制。其中，建立PID的控制模型，其实正是将直流电机输入的PWM波与输出的转速建立关系，寻找合适的参数，其实是让转速到PWM的映射尽快实现的过程。

本次例程在上一篇推文例程基础上写的，有遗忘可以关注上一篇推文。

初始化：



PID的主要部分：



**PID控制的调参**

兴许你已经发现了，PID控制中最难的，不是原理或代码，而是调参。

如今PID控制应用到的大多数场景，依然根据经验调参，参数随机化并不精准。

总结起来，有以下几个经验：

1. 先设定采样周期与Kp值，并调节到小车能大概稳定；
2. 再添加积分项，建议从小往上加；
3. 最后视情况考虑是否需要添加微分项。

具体总结下来，有如下一个口诀：

参数整定找最佳，从小到大顺序查。

先是比例后积分，最后再把微分加。

曲线振荡很频繁，比例度盘要放大。

曲线漂浮绕大湾，比例度盘往小扳。

曲线偏离回复慢，积分时间往下降。

曲线波动周期长，积分时间再加长。

曲线振荡频率快，先把微分降下来。

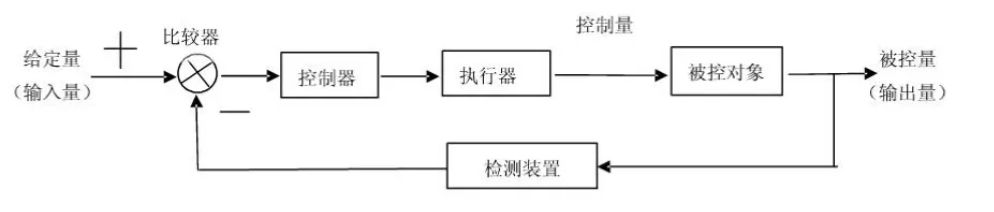
动差大来波动慢，微分时间应加长。

理想曲线两个波，前高后低四比一。

一看二调多分析，调节质量不会低。

到这里，读者也许就会不太满意了，难不成所有用到PID的控制系统都要逐个去试参数不成？这倒也不是。在工业实践中，最广泛应用的PID参数科学整定方法当属Z-N整定法，之后该方法又针对阶跃信号进行了改进。近些年，也有研究者利用蚁群算法等自适应算法进行PID参数的整定，在这里我们不做展开。

那么日常直流电机的调速中，有没有较为快捷的整定方法呢？机小协再教大家一招：**利用MATLAB工具箱的PID参数科学整定方法辅助PID调参。**



在MATLAB中科学整定PID参数主要分为两步：估计传递函数及整定PID参数。

估计传递函数：

MATLAB系统辨识的是执行器及被控对象部分的传递函数，该辨识是对线性系统的辨识过程，因此我们近似认为电机系统为线性。

将输入、输出数据以向量形式分别导入到MATLAB。其中输入向量为一个电机单位阶跃信号（如analogWrite(100)），输出向量为编码器测得的转速。

导入数据后，使用System Identification工具箱，选择Time domain data进行传递函数的估计。预测完毕的传递函数，可以通过给定阶跃输入判断输出是否吻合的方法来验证系统的辨识准确率。

整定PID参数：

PID控制属于控制器部分，在系统估计传递函数的基础上利用PID Tuner工具箱进行参数的整定即可。

详细步骤请参考：[(18条消息) 利用matlab工具箱的pid参数科学整定方法\_萧易风船长的博客-CSDN博客](https://blog.csdn.net/qq_35975855/article/details/125977247?app_version=5.7.0&code=app_1562916241&csdn_share_tail=%7B%22type%22%3A%22blog%22%2C%22rType%22%3A%22article%22%2C%22rId%22%3A%22125977247%22%2C%22source%22%3A%22qq_35975855%22%7D&uLinkId=usr1mkqgl919blen&utm_source=app)

博客原创：陈烨柯

文案：许子绎

排版：