PID 控制通用控制设计思路说明

作者：蒋庭佳

目录

背景

首先，抛出沿用多年的经典 PID 控制公式

$$\text{Output} = K_P e(t) + K_I \int e(t)\,dt + K_D \frac{d}{dt} e(t)$$

$$\text{Where}: \; e = \text{Setpoint} - \text{Input}$$

上述公式的具体原理，就不加以说明了，自控原理已经分析得很清楚。针对上述代码，为了做一个性能较强的工业控制器，还需要考虑一下几个问题：

1. 采样时间——改变采样时间会带来怎样的后果
2. 微分项的影响——突然改变设定值或者微分时间，如何避免冲击
3. PID 参数改变——PID 控制参数的突然改变，如何避免突变
4. 积分参数——突然改变 I 参数，如何避免冲击
5. 开关——在控制过程中，PID 调节开关突然的开启及关闭
6. 初始化——PID 运行一段时候后关闭，经过一段时间再次开启，如何避免突变
7. 调节的方向——这个不是大问题，仅仅是为了保证系统超预计的方向运行

下面针对上述每一个问题进行分析

采样时间的改变

问题定义

一般来说，PID 控制都是周期性调用（也就是意味着，每次计算的间隔都是固定的常量），但或多或少，由于各种需求会被非周期调用。如果非得修改采样时间，对 PID 控制进行非周期调用，那么这样会导致以下问题：

观察 PID 方程：

$$\text{Output} = K_P e(t) + K_I \int e(t)dt + K_D \frac{d}{dt} e(t)$$

$$\text{Where : } e = \text{Setpoint - Input}$$

如果采样时间变化了，那么对于积分项和微分项（也就是 KI 和 KD 对应的项），这两项是和采样时间间隔有关的，那么则需要进行额外的微积分运算(不能再按照原来写好的代码进行运算，必须对时间参数进行调整)。

解决方案

如果算法被固定的周期间隔调用，那么运算将会变得很简单，并且也能够周期性的获取到精确的运算结果。那么朝着这个思路。需要想办法将已经被改变的采样周期让 PID 控制器认为没有改变，依旧沿用原来的运算过程。

如果要解决这个问题，需要在采样间隔改变后，Ki、Kd 按照比例放大/缩小了与采样间隔改变相同的倍数。为什么只对 Ki 和 Kd 进行处理？

如果 Ki 变化了，那么和经典的 PID 控制公式结果不是会差很大吗

观察积分项，并改写为离散形式：

$$K_I \int e(t)dt = K_I(e(t) + e(t-1) + \cdots + e(1))$$

如果在调节过程中，Ki 是一个常量的话，那么可以进一步改写为：

$$K_I \int e(t)dt = K_{It}e(t) + K_{It-1}e(t-1) + \cdots + K_{I1}e(1)$$
$$= K_{It}e(t) + Inter$$

上式的第一项分别观察 Ki 及 e(t)，改变采样间隔后，e(t)受采样间隔影响会产生对应时间内的变化，而 Ki 等比例反向放大/缩小，效果相当，证必。

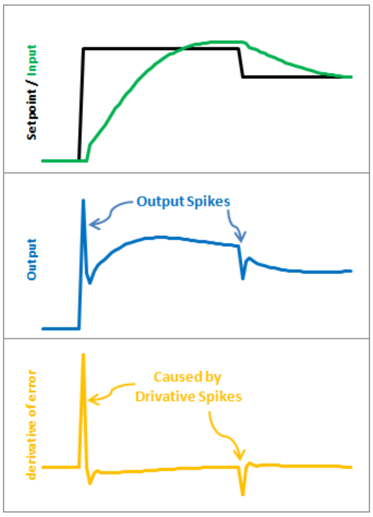
结论

无论调用 PID 算法多么频繁，此算法还是仅仅会周期性的计算。这样的好处是，PID 控制器可以按照它熟悉的路子走到底。

微分冲击

问题定义

既然叫做微分冲击，那么这个问题肯定和微分项有关，并且是微分项会受到一个突发状况的影响，产生一个较大的冲击，详见下图。



从第一张图看出，当设定值产生一个阶跃后，被控量 Input 随着时间慢慢向设定值靠近，第二张图反应的是设定值产生阶跃后控制量 Output 的变化，可以发现，Output 会突然产生一个较大的阶跃，这是由于 PID 经典控制方程计算式不可不免导致的。第三张图描述的是 Output 的梯度（也就是变化率：值变化：时间变化），同样可以发现一个脉冲，并且这个脉冲可能会非常的大（dt 非常小），远远超过 Output 变化量。同理，图中页描述了当设定值突然减小，Output 及其对应的梯度变化情况。对于一般的系统来说，不希望这样的突变发

生（可以想象如果采样周期很长，那么这个冲击会持续很长时间，系统估计就失控了）。

解决方案

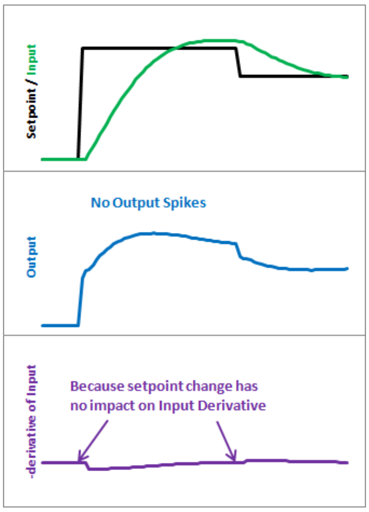
结合经典 PID 控制方程，来看一个公式：

$$\frac{d}{dt}\text{Error} = \frac{(\text{Setpoint}_i - \text{Setpoint}_{i-1})}{dt} - \frac{(\text{Input}_i - \text{Input}_{i-1})}{dt}$$

Setpoint 这一项就会产生一个巨大的冲击，而且仅会产生一次，在下一个计算周期就会消失。简单 Serpoint 这一项移除，也就是认为对于微分项来说不存在设定值的改变。如果这么处理，系统会不会失控呢？

上述问题的答案是不会，用 Input 变化量取反来代表 Error 的变化（其实就是忽略 Setpoint 的变化）。这里不讨论经典的 PID 控制方程，控制器的目标是使 Input 根据 Setpoint 往预计的方向靠拢，上述做法无可否认，与经典 PID 方程不一致，但核心思想是一致的。由于 Setpoint 变化仅对 Output 产生一次影响（也就是产生了那个尖峰），一旦步入一下个计算周期，Output 就不在收到 Setpoint 变化的影响，回归正常，不影响 PID 调节。

结论

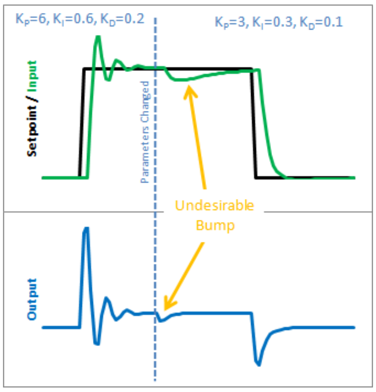


1. Output 梯度尖峰被去除
2. 控制量原来存在一个尖峰被消除
3. 被控量的微分变得较为平坦

控制参数突变

问题定义

在 PID 控制实际的应用过程中，可能会存在需要突然改变 PID 调谐参数 Kp、Ki、Kd 的情形，那么如果突然改变调谐参数，会有什么影响呢？首先先看一张图：



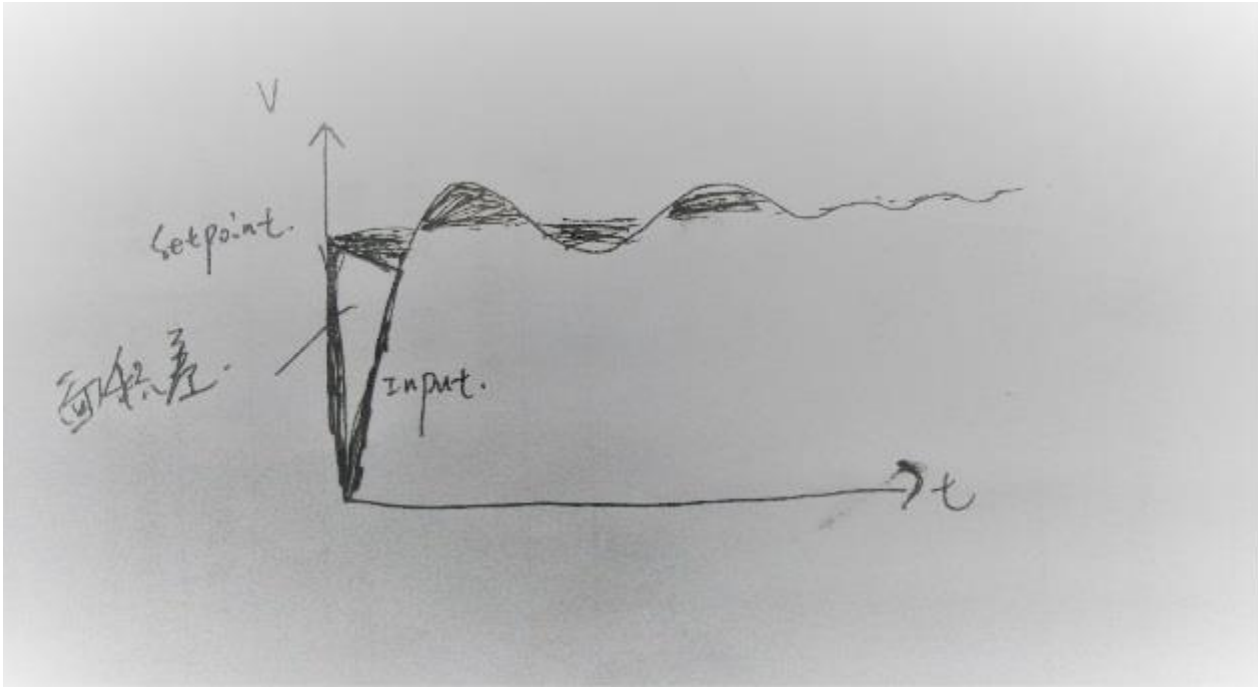
如果在系统运行的过程中，对调谐参数进行较大的改变，那么 Output 会产生一个突变，有点略微像“坑”。下图给出了一个改变 PID 参数对输出影响的量化分析：

Output is halved

	Output	= kp	* error	+ ki	* errSum	- kd	* dInput
Just Before	0.98	6	-0.01	0.6	1.73	-0.2	0.02
Just After	0.49	3	-0.01	0.3	1.72	-0.1	-0.01

Because the Integral Term suddenly gets halved

上述现象描述了在系统进入稳态后，突然改变 PID 参数导致的变化，起主要引起输出较大变化的因子为 I 参数，由于进入稳态 P 参数相乘因子 E 变化不大，D 参数同样不大，但是 I 参数相乘因子是关于时间的积分（可以想象如果一开始被控量与设定值相距较远，而积分表示的是带方向的面积和，所以必然存在一个方向会有较大面积，见下图），所以，会引起较大的变化。



解决方案

先看一组算式。

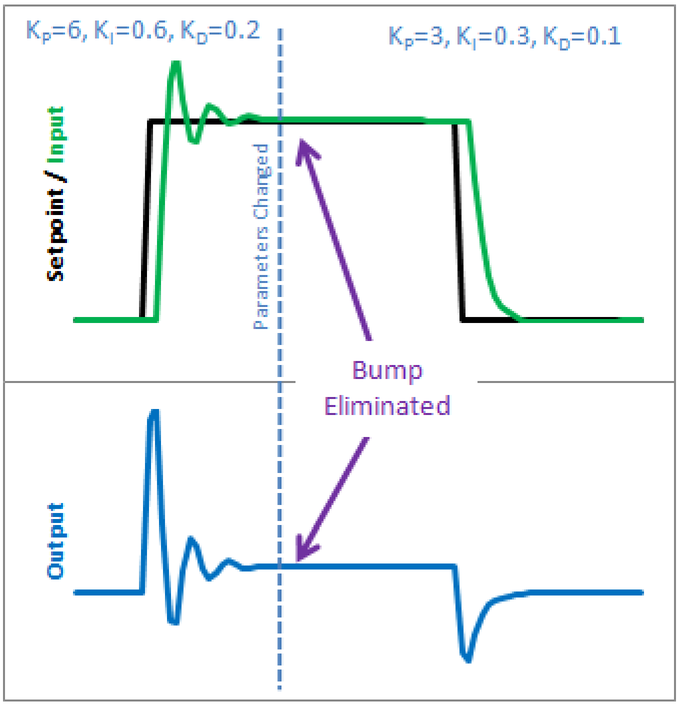
$$K_I \int e\,dt = \int K_I\, e\,dt$$

$$\int K_I\, e\,dt \approx K_{I_n} e_n + K_{I_{n-1}} e_{n-1} + \ldots$$

第一个等式在 KI 是常数的时候没问题，不是常数的时候需要评估，尽管不完全成立，但如果在稳态时 E 非常小，那么，也是可以接受。这似乎和经典的 PID 等式不一样。换个角度，经典的 PID 控制 I 项，也仅仅是为了消除静态误差而确定的，如果在这个大前提下，换一种方式消除静态误差也是可以的，仅仅是牺牲了“力度”。

结果

图示



量化结果

No Output Bump

	Output =	kp *	error +	ITerm -	kd *	dInput
Just Before	0.98	6	-0.01	1.04	-0.2	0.02
Just After	1.01	3	-0.01	1.04	-0.1	-0.01

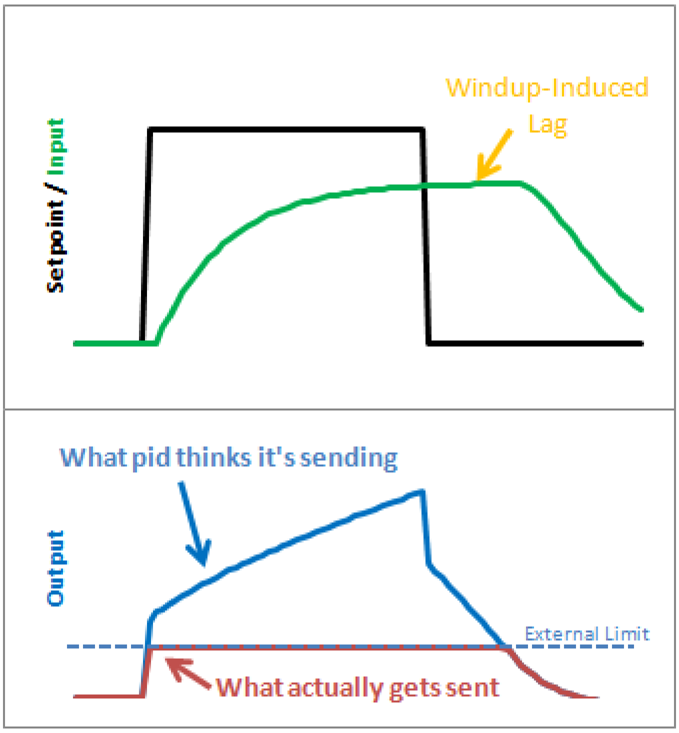
Because now Ki only affects us moving forward

从图表中可以看出，尽管 PID 参数发生了较大的改变，但输出仍然变得连续。在上述过程中，牺牲了系统的响应的“灵敏度”增加了控制的稳定性，系统的灵敏度往往可以通过增加计算频率来改善，大幅提高采样频率可以起到较快的更新积分项的作用。

设定值突然改变

问题定义

首先先看一下上述问题的图示说明：



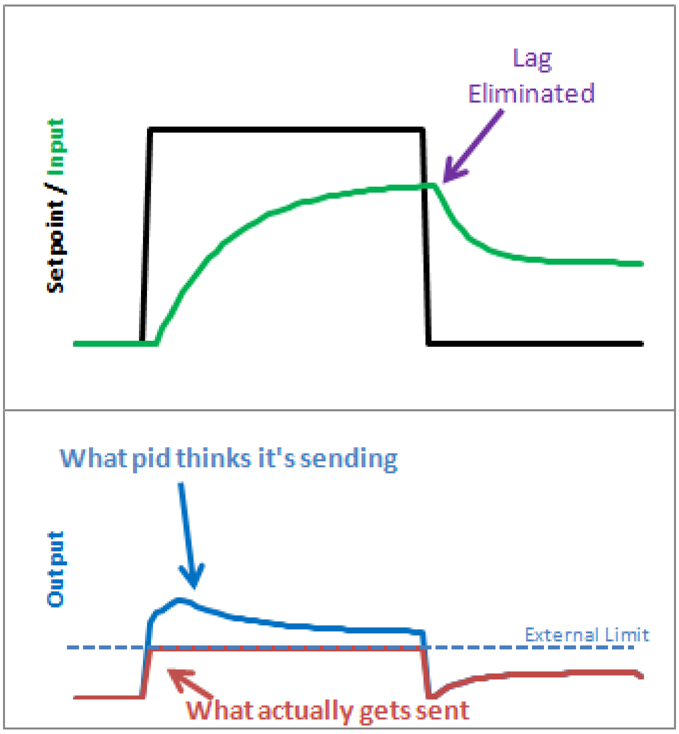
假设 PID 控制的输出的限制是 255，如果按照 PID 经典公式计算，PID 控制器不知道其输出的极限值是 255，由于设定值较大，那么输出会一直增长，超过 255，达到 300/400 或者更高。此时，突然降低设定值到一个较低的位置（比被控量更小），那么，由于积分项的作用，输出会需要一定的时间才会渐渐过度到极限值以下，然后再慢慢降到所需的设定值。从图中看，就是绿色线条的 Lag，这个是不希望看到的。

解决方案

有多种手段可以减小上述问题的影响。

最简单的方式：告诉 PID 软件输出限制是什么，一旦达到了限制，PID 参数将不再累加（积分），注意这里不是“生硬”的将输出限制在极限值，而是到达极限值后，要关闭积分项的继续累加，否则一旦 PID 恢复，积分项超限部分仍

会进入控制流程，影响输出。至于微分项，在时间上仅和上一次的被控量值存在关联，所以影响是微乎其微的。



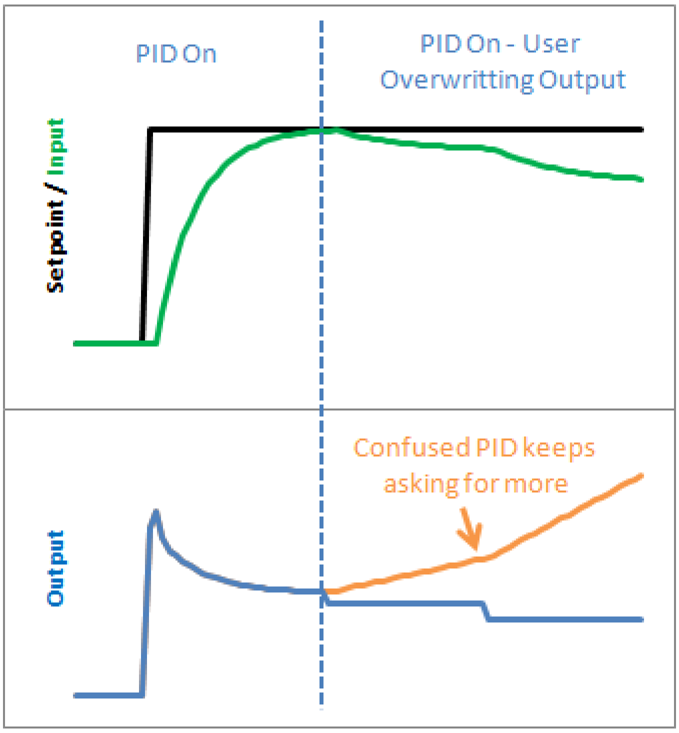
从图中看出：

一旦积分项控制住了，输出值则会被限制在一个合理的范围中，就算超限，也不会过大。而被控量在设定值降到合理范围后，也不会存在时间上的较大滞后。

PID 控制的突然开启和关闭

问题定义

先看问题的图示：

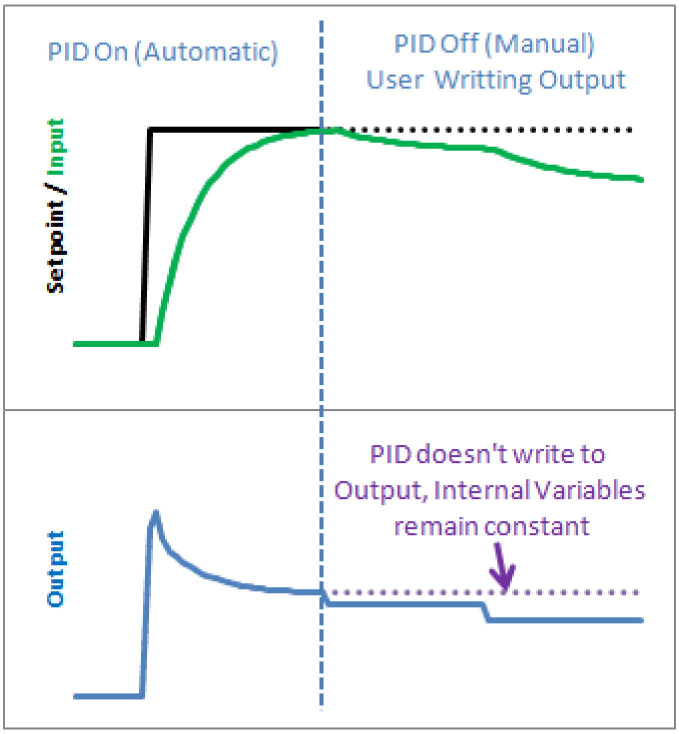


图中，蓝色的竖虚线表示在时间轴上，PID 控制由开启状态转为关闭的状态。如果对于关闭 PID 采取的操作是强制设定输出为“PID 由开启转关闭对应的输出值或更小或为 0”。由于输出被设为定值，由于输出的“力度”不够，那么被控量会缓慢减少。但对于 PID 控制器来说，他会不知道 PID 已经关闭了，由于被控量见笑了，他就会努力增加输出，以求改变这种态势。但现实是残酷的，为何我“放肆”增加输出，被控量却没有按照我设定的方向发展。此时，若突然又将 PID 打开了，将会有一个极大的输出对系统作用，稳定性降低。

解决方案

上述问题，并不是一个大问题，其实在关闭 PID 控制的时候，仅需要退出 PID 计算循环，让各控制参量仍保持原样即可。if(!inAuto) return; 和 setMode 函数完成了上述功能。

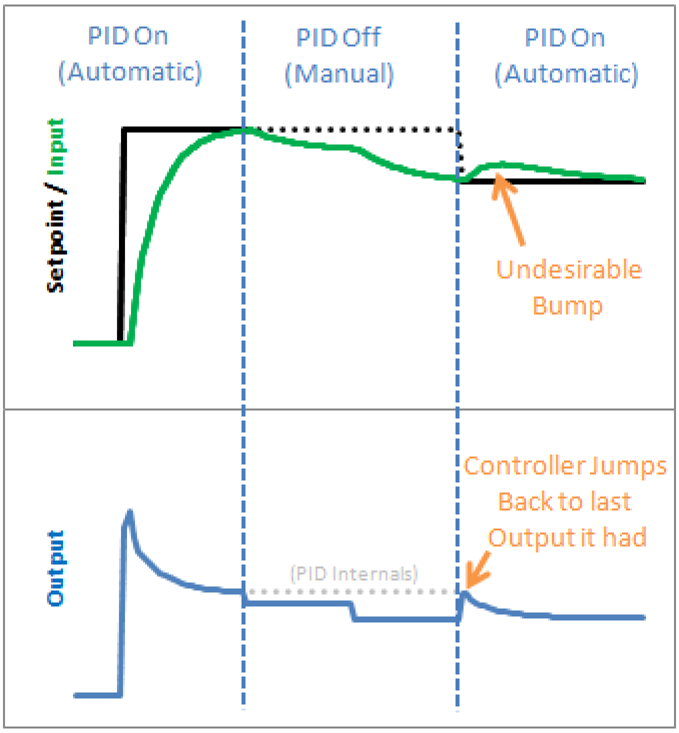
结论


PID On (Automatic)
PID Off (Manual)
User Writting Output
Setpoint / Input
Output
PID doesn't write to Output, Internal Variables remain constant

初始化

问题定义

先看问题的图示：



上一节，讨论的是 PID 控制由开转关的过程中存在的问题，那么紧接着上一节，如果在关闭后，突然再次开启，那么会产生什么问题呢？直观来看，对于被控量会出现图中，绿色线的一个 bump，这个 bump 是由于由关转开的一瞬间，输出突然放大，那么对于灵敏度高的系统，则会出现，滞后大的系统可能不会如此明显，但不管怎么说，为了杜绝一切不利因素，都应该想办法消除这个 bump。由于这个 bump 的根因是由于输出的突变而造成的，所以需要想办法控制这个输出的突变。

解决方案

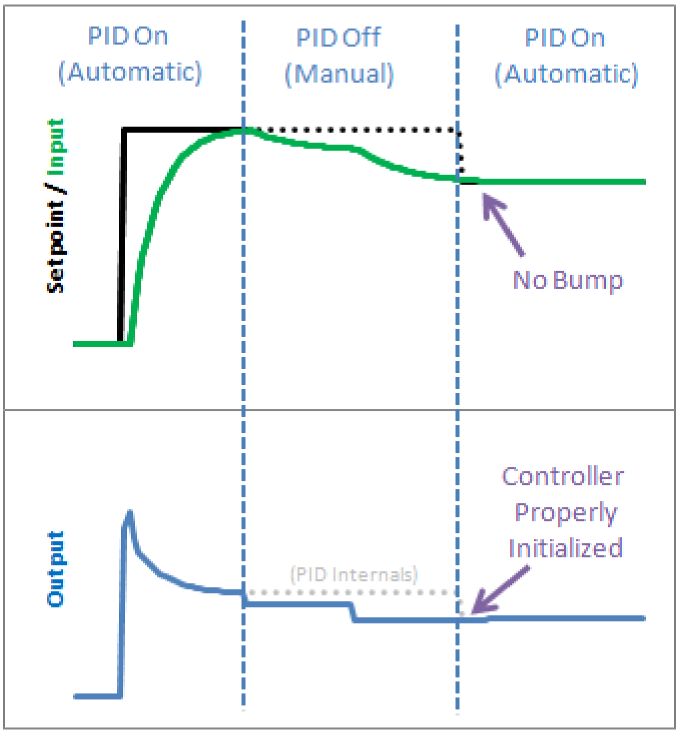
想一想，所有此类问题都是发生在时间轴上的，那么 PID 控制中时间轴会影响的项只有积分项和微分项（可以想象，比例向为 Kp * （设定值 - 被控量），这是一个连续量，不存在突变的可能（除非是采样时间特别长，在改变输出前，被控量飞上天了），所以只需要从这两项上想办法，控制住这两项的突变，即可控制住输出的突变。首先问题发生在 PID 关闭转开启的过程中，由于

PID 开启关闭控制的函数是 SetMode，所以在此函数中，增加一个 initial 函数用于控制积分项和微分项即可。具体做法是：

1、更新微分项上一次采样值为 PID 开启一瞬间的采样值，这样可以保持微分项维持在上一次开启结束的状态不变。

2、将积分项设置为当前的输出，为什么要这么做呢？积分项由于 PID 关闭长时间维持关闭前的状态，一旦开启，如果不改变积分项，输出会瞬间被拉回到上一次开启结束的状态，突变就这样产生了

结论



控制方向

问题定义

一个完善的PID控制器的调节方式，应该有两种:

1、正向调节，也就是输出增大，那么被控量也会增大

2、反向调节，与正向调节相对应，输出增大，被控量却减小

上述两种方式很容易理解，举个例子，例如在温度控制的加热控制中，加热输出增大，那么温度随之上升（正向调节），而在对应的制冷控制中，如果制冷输出增大，那么温度应该是随之下降的（方向调节）。

解决方案

这个问题很容易解决，在代码中，仅需要将Kp、Ki、Kd设定为负的即可，在代码中，提供了一个函数SetControllerDirection和两个宏DIRECT和REVERSE用于控制输出的方向，如果需要正向输出，在构造PID之后，调用SetControllerDirection（DIRECT）即可将输出设为正向（正向是默认的，构造PID的时候，默认是正向调节），反之亦然。

结论

此PID控制算法具备了PID控制的基本要素，并且考虑了一切特殊的情况，大大增加了环境适应性。

结语

上述PID控制器设计思路并非仅使用与温度控制，在其他自动化控制过程中都可以使用，并且都应当考虑到。整个设计思路考虑了许多抗干扰的问题，在目前我司温度控制中，可能体现不出来，但如果在新研发的设备中有新的需求，需要PID控制，可以直接使用，无需二次开发，同样，此控制思路可复制性非常强，简单的转换即可进行二次开发。