PID控制算法其实指的就是**「比例、积分和微分控制」**，如下：

可以用下面的公式表示：

$$U(t) = kp(err(t) + \frac{1}{T_1}\int err(t)dt + \frac{T_D derr(t)}{dt})$$

比例控制算法

假设有一个水缸，初始时水缸里的水高度是0.2米，目的是要保证水位维持在1米的高度，那么现在的水位距离目标水位之间存在一个误差$error = 0.8$。假设现在旁边有个人通过往缸里加水的方式控制水位，则比例控制算法的运行方式如下：

- 加入水的量u与误差$error$成正比，即$u = kp * error$
- t=1时，$u = 0.5 * 0.8 = 0.4$，使水位在0.4的基础上上升0.4，达到0.6
- t=2时，$u = 0.5 * 0.4 = 0.2$，使水位在0.6的基础上上升0.2，达到0.8
- ……

按照上述方式循环下去，水位最终会达到1米

但上述算法也有问题。假设水缸存在漏水的情况，每次加水过程中会漏掉0.1米高度的水，那么在水位达到0.8时，将不再变化。因为此时$error = 0.2$，每次往缸中加水的量是0.1，漏掉的水高度也是0.1，加入的水和漏掉的水相抵消，水位不会变化。这个问题就是**「稳态误差」**

积分控制算法

基于现实中普遍存在稳态误差的情况，再引入一个分量，该分量与误差的积分是正比关系。因此，比例+积分控制算法为：

$$u = kp * error + ki * \int error$$

还是以水缸加水的例子来说明，第一次的误差error是0.8，第二次的误差是0.4，至此，误差的积分（离散情况下积分其实就是做累加）是1.2。这个时候的控制量，除了比例的那一部分，还有一部分就是一个系数ki乘以这个积分项。由于这个积分项会将前面若干次的误差进行累计，所以可以很好地消除稳态误差

微分控制算法

考虑刹车的场景。平稳的行驶车辆，当发现前面有红灯时，为了使得行车平稳，基本上提前几十米就放松油门并踩刹车了。当车辆距离停车线非常近的时候，则使劲踩刹车，使车辆停下来。整个过程可以看作一个加入微分的控制策略。在离散情况下，微分就是error的差值，即t时刻与t-1时刻的差，则：

$$u = kd * (error(t) - error(t-1))$$

其中，kd是一个系数。可以看到，在刹车过程中，因为error是越来越小的，所以这个微分控制项一定是负数，在控制中加入一个负数项，作用是为了防止汽车由于刹车不及时而闯过了线。从常识上可以理解，越是靠近停车线，越是应该注意踩刹车，不能让车过线，所以这个微分项的作用，可以理解为刹车。当车距离停车线很近并且车速还很快时，这个微分项的绝对值就会很大，从而表示应该用力踩刹车才能让车停下来

再来看开头的那个公式，已经很清楚了。在离散情况下，可以转换为：

$$u(k) = K_p e(k) + \frac{K_p T}{T_i}\sum_{n=1}^{k} e(n) + \frac{K_p T_d}{T}(e(k) - e(k-1))$$

为了方便起见，将这些系数进行统一：

$$u(k) = K_p e(k) + K_i \sum_{n=1}^{k} e(n) + K_d(e(k) - e(k-1))$$

到这里，PID算法的原理和方法就说完了，剩下的就是实践了。在真正的工程实践中，最难得是如何确定三个项的系数，这往往需要大量的实验和经验来确定