PID算法及应用

反馈

- 控制论的基本概念
- 反馈控制是指将系统的输出信息返送到输入端，与输入信息进行比较，并利用二者的偏差进行控制的过程
- 反馈控制其实是用过去的情况来指导现在和将来
- 在控制系统中，如果返回的信息的作用是抵消输入信息，称为负反馈，负反馈可以使系统趋于稳定；若其作用是增强输入信息，则称为正反馈，正反馈可以使信号得到加强。

有无反馈?

- 开环控制
- 闭环控制

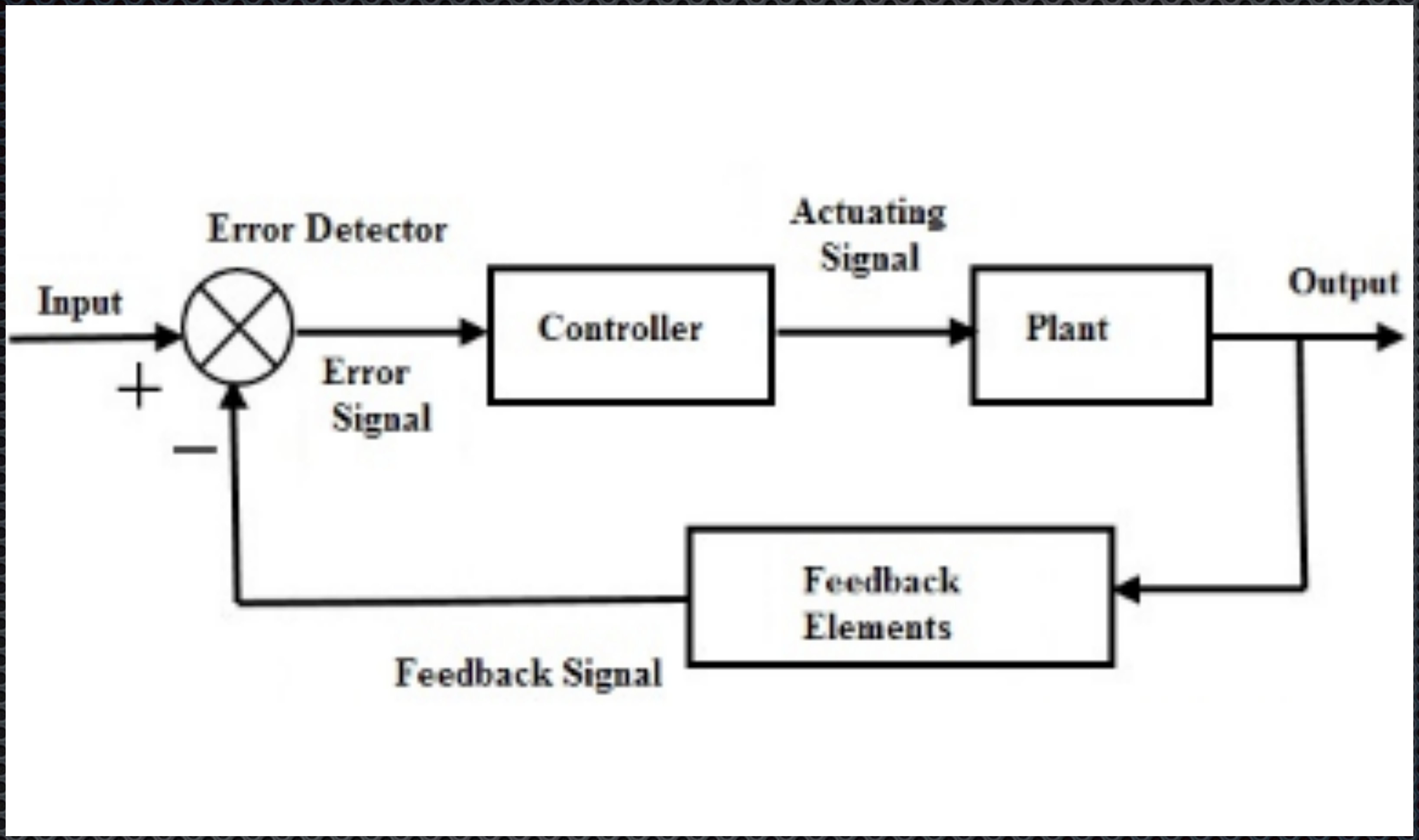
开环系统

- 无反馈系统
- open loop control，又称“开环控制系统”，是指在一个控制系统中系统的输入信号不受输出信号影响的控制系统, 也就是，不将控制的结果反馈回来影响当前控制的系统
- 典型应用:
    - 自动售货机、自动洗衣机、产品自动生产线、

闭环控制

- 闭环控制是指作为被控的输出量以一定方式返回到作为控制的输入端，并对输入端施加控制影响的一种控制关系，即带有反馈信息的系统控制方式
- 优势:
  - 精确
  - 自适应调节
  - 稳定性高
- 应用
  - 温控、无人机飞控、电机控制。。。恒温加热器、太阳能系统、导弹发射器、汽车发动机、自动烤面包机、使用涡轮的水控制系统

闭环控制


Error Detector
Actuating
Signal
Input
Controller
Plant
Output
+
Error
Signal
−
Feedback
Elements
Feedback Signal

PID

- 在过程控制中，按偏差的比例（P）、积分（I）和微分（D）进行控制的PID控制器（亦称PID调节器）是应用最为广泛的一种自动控制器
- 优点： 原理简单，易于实现，适用面广，控制参数相互独立，参数整定方式简便，结构改变灵活（PI、PD、…）
- 当被控对象的结构和参数不能完全掌握，或得不到精确的数学模型时，控制理论的其它技术难以采用时，系统控制器的结构和参数必须依靠经验和现场调试来确定，这时应用PID控制技术最为方便。即当我们不完全了解一个系统和被控对象，或不能通过有效的测量手段来获得系统参数时，最适合用PID控制技术

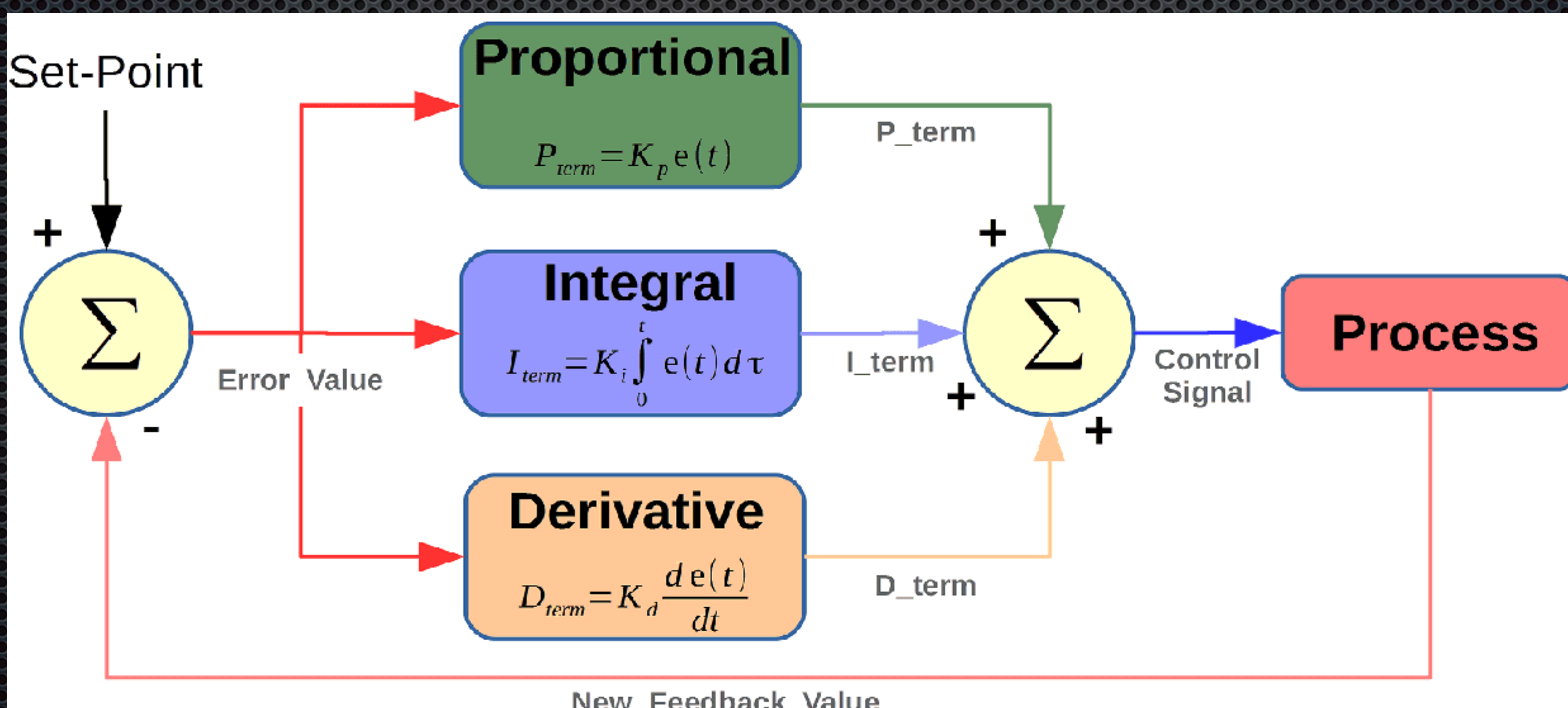
历史

- 最早提出PID控制理念的是瑞典裔美国人奈奎斯特， 在一篇论文当中写到了采用图形的方法来判断系统的稳定性
- 在他的基础上，伯德等人建立了一整套在频域范围设计反馈放大器的方法，后被用于自动控制系统的分析和设计
- 这也是PID算法最早从书面走向实践，实际上算法远比此有更长的历史，早在1917年就出现了论述PID控制方法的论文
- 于是在1936年，PID开始广泛的用于工业控制，虽然没有今天这样方便的可编程嵌入式芯片，但当时的人们也用了堪称登峰造极的模拟电路控制，实现了不错的PID效果，难以想象那个年代人的创造力有多强

连续控制系统的理想PID控制

$$
\begin{aligned}
u(t) &= K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{d}{dt} e(t) \\
&= K_p \left[e(t) + \frac{1}{T_i} \int_0^t e(\tau)d\tau + T_d \frac{d}{dt} e(t) \right]
\end{aligned}
$$

式中，K_p——比例增益；
K_i——积分增益；
K_d——微分增益；
T_i——积分时间常数；
T_d——微分时间常数；
$u(t)$——PID控制器的输出信号；
$e(t)$——给定值$r(t)$与测量值之差。

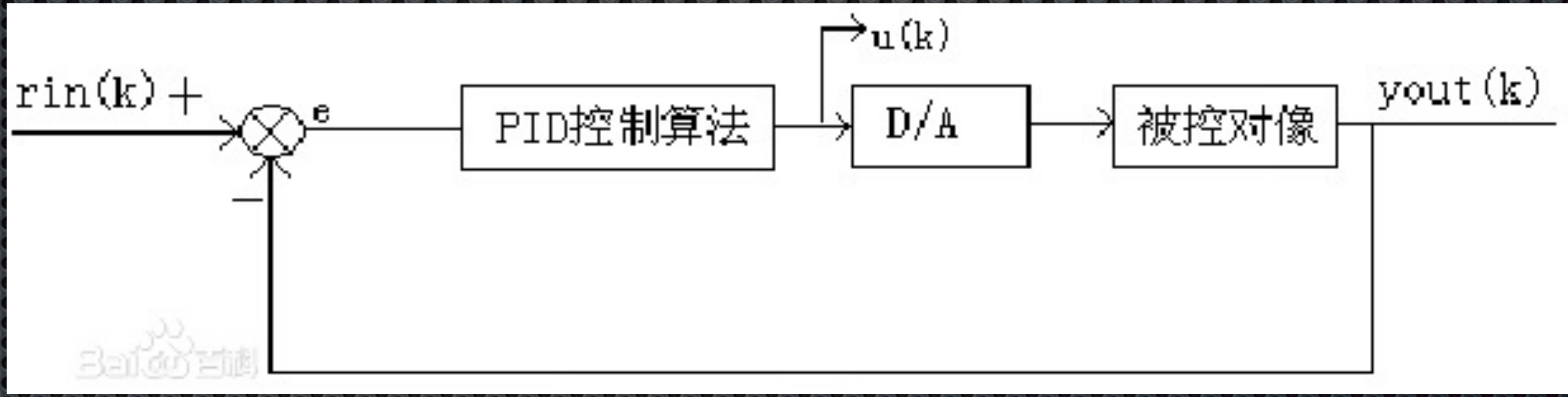


离散化PID算法

- 数字PID控制算法在实际应用中又可分为两种：
  - 位置式PID控制算法
  - 增量式PID控制算法
- 控制理论上两者是相同的，但在数字量化后的实现上会存在差别

位置式数字 PID 控制

- 对式1作离散化处理就可以得到位置式数字 PID 控制算法，即以一系列的采样时刻点 k代表连续时间 t，以矩形法数值积分近似代替积分，以一阶后向差分近似代替微分，可得到其 k 采样时刻的离散 PID 表达式



$$u(k)=K_p\left(e(k)+\frac{T}{T_i}\sum_{j=0}^{k}e(j)+\frac{T_d\left(e(k)-e(k-1)\right)}{T}\right)=K_p\times e(k)+K_iT\sum_{j=0}^{k}e(j)+K_d\frac{e(k)-e(k-1)}{T}$$

上式中，$K_i=K_p/T_i$，$K_d=K_p*T_d$，T 为采样周期，k 为采样序号，k=1,2,...，$e(k-1)$和$e(k)$分别为第（k-1）和第 k 时刻所得到的系统偏差信号。

增量式 PID 控制

- 指控制器的输出是控制量的增量 $\Delta u(k)$，当执行机构需要的是控制量的增量而不是位置量的绝对数值时，可以使用增量式 PID 控制算法进行控制。

$$u(k-1) = K_p \times e(k-1) + K_i T \sum_{j=0}^{k-1} e(j) + K_d \frac{e(k-1) - e(k-2)}{T}$$

$$\Delta u(k) = u(k) - u(k-1)$$

$$\Delta u(k) = K_p \left(e(k) - e(k-1)\right) + K_i T e(k) + \frac{K_d \left(e(k) - 2e(k-1) + e(k-2)\right)}{T}$$

上式中，$K_i=K_p/T_i$，$K_d=K_p*T_d$，T 为采样周期，k 为采样序号，k=1,2,...，$e(k-1)$和$e(k)$分别为第（$k-1$）和第 k 时刻所得到的系统偏差信号

PID控制器参数整定的一般方法

- PID控制器的参数整定是控制系统设计的核心内容。它是根据被控过程的特性确定PID控制器的比例系数、积分时间和微分时间的大小
- PID控制器参数整定的方法很多，概括起来有两大类：
  - 一是理论计算整定法,主要是依据系统的数学模型，经过理论计算确定控制器参数
    - 这种方法所得到的计算数据未必可以直接用，还必须通过工程实际进行调整和修改
  - 二是工程整定方法， 主要依赖工程经验，直接在控制系统的试验中进行，且方法简单、易于掌握，在工程实际中被广泛采用
    - PID控制器参数的工程整定方法，主要有临界比例法、反应曲线法和衰减法
      - 三种方法各有其特点，其共同点都是通过试验，然后按照工程经验公式对控制器参数进行整定
      - 但无论采用哪一种方法所得到的控制器参数，都需要在实际运行中进行最后调整与完善

齐格勒－尼科尔斯方法参数控制
Ziegler–Nichols method

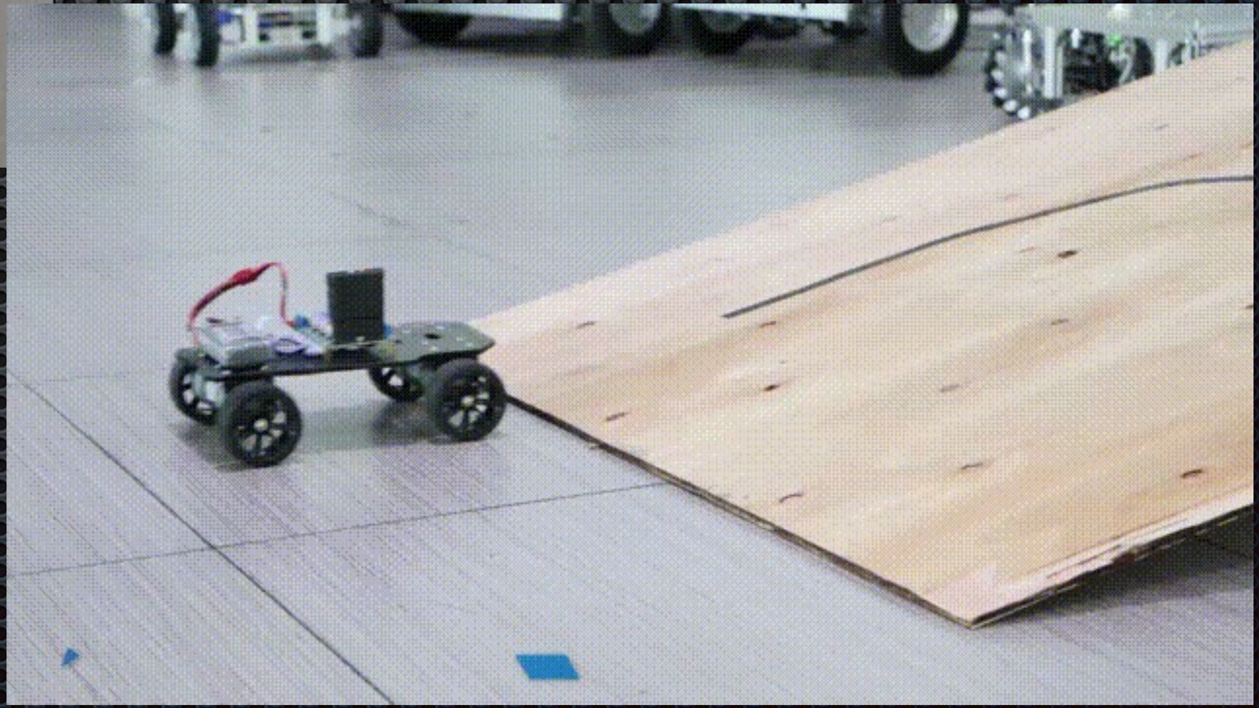
- 齐格勒－尼科尔斯方法的调试方式为，首先将积分和微分增益设置为0，然后比例增益从零开始逐渐增加，直到到达极限增益K_U，此时控制器输出值以恒定值振荡 。K_U和振荡周期T_U根据不同的类型，按下表中的方式来设置比例、积分和微分增益。

齐格勒－尼科尔斯方法			
控制类型	K_p	K_i	K_d
比例	$K_u/2$	-	-
比例-积分	$K_u/2.2$	$1.2K_p/T_u$	-
经典比例-积分-微分(PID)	$0.60K_u$	$2K_p/T_u$	$K_pT_u/8$
Pessen Integral Rule	$0.7K_u$	$2.5K_p/T_u$	$0.15K_pT_u$
部分过冲量	$0.33K_u$	$2K_p/T_u$	$K_pT_u/3$
无过冲量	$0.2K_u$	$2K_p/T_u$	$K_pT_u/3$

书上的常用口诀

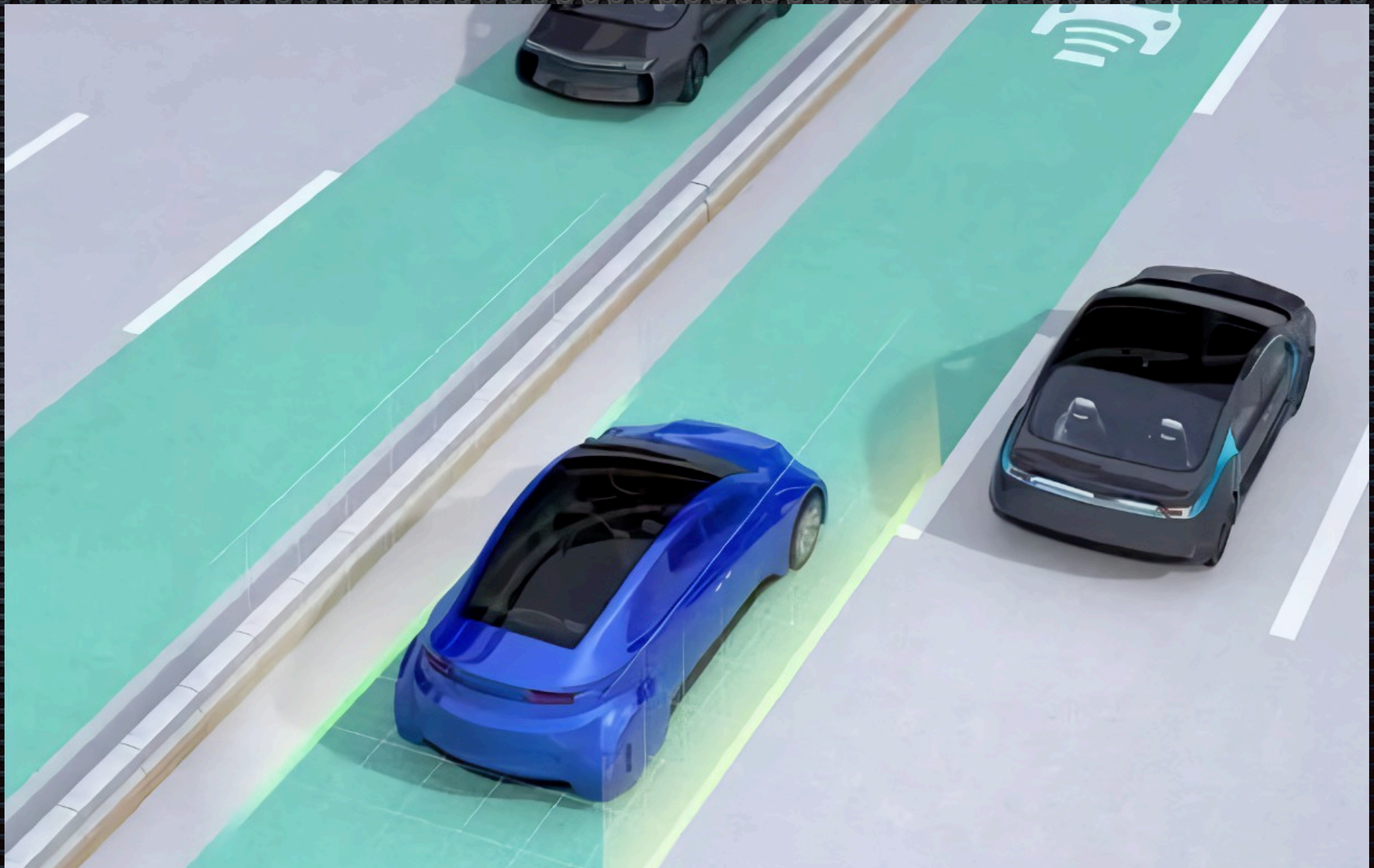
参数整定找最佳，从小到大顺序查
先是比例后积分，最后再把微分加
曲线振荡很频繁，比例度盘要放大
曲线漂浮绕大湾，比例度盘往小扳
曲线偏离回复慢，积分时间往下降
曲线波动周期长，积分时间再加长
曲线振荡频率快，先把微分降下来
动差大来波动慢，微分时间应加长
理想曲线两个波，前高后低4比1
一看二调多分析，调节质量不会低


循迹黑线功能演示



车道保持

- 车道保持辅助系统属于智能驾驶辅助系统中的一种，它可以在车道偏离预警系统（LDWS）的基础上对转向系统进行控制辅助车辆保持在本车道内行驶。



作业 – 沿墙移动

- 在本任务中，将实现无人车自主运行的感知和控制ROS节点
- 这个作业的目的是实现一个简单的沿着墙运动的算法，在走廊里，维持车与墙平行。利用激光雷达的传感器数据和实现一个PID控制器来跟踪墙壁

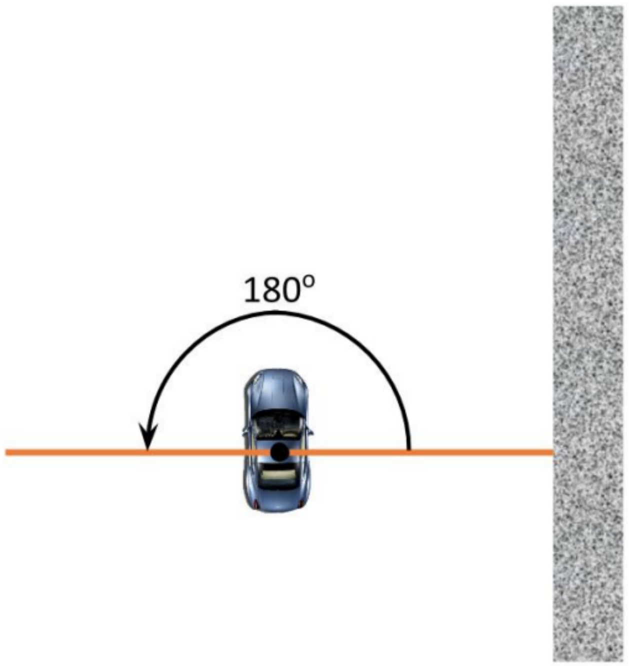


Figure 1: Lidar scan angles

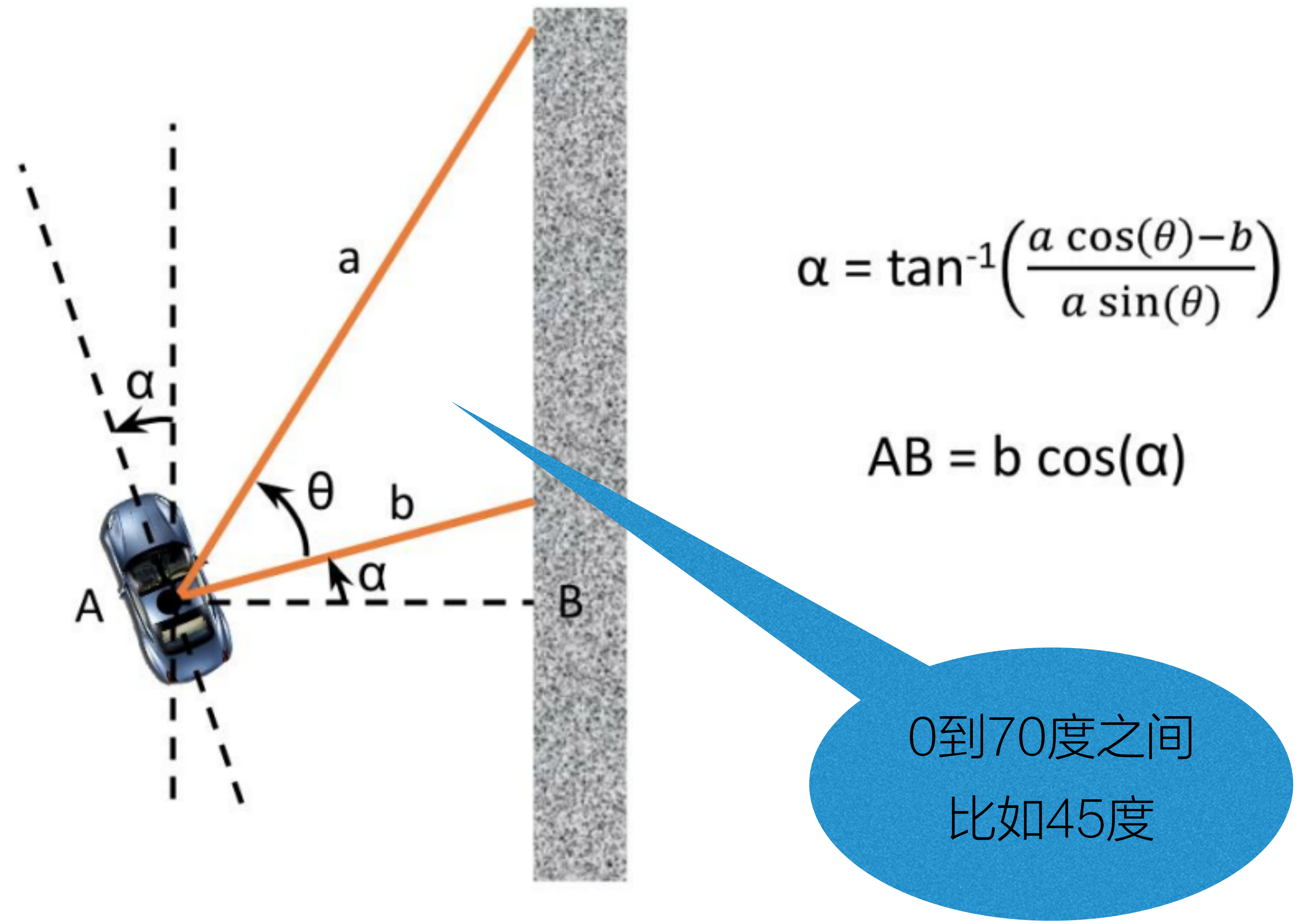
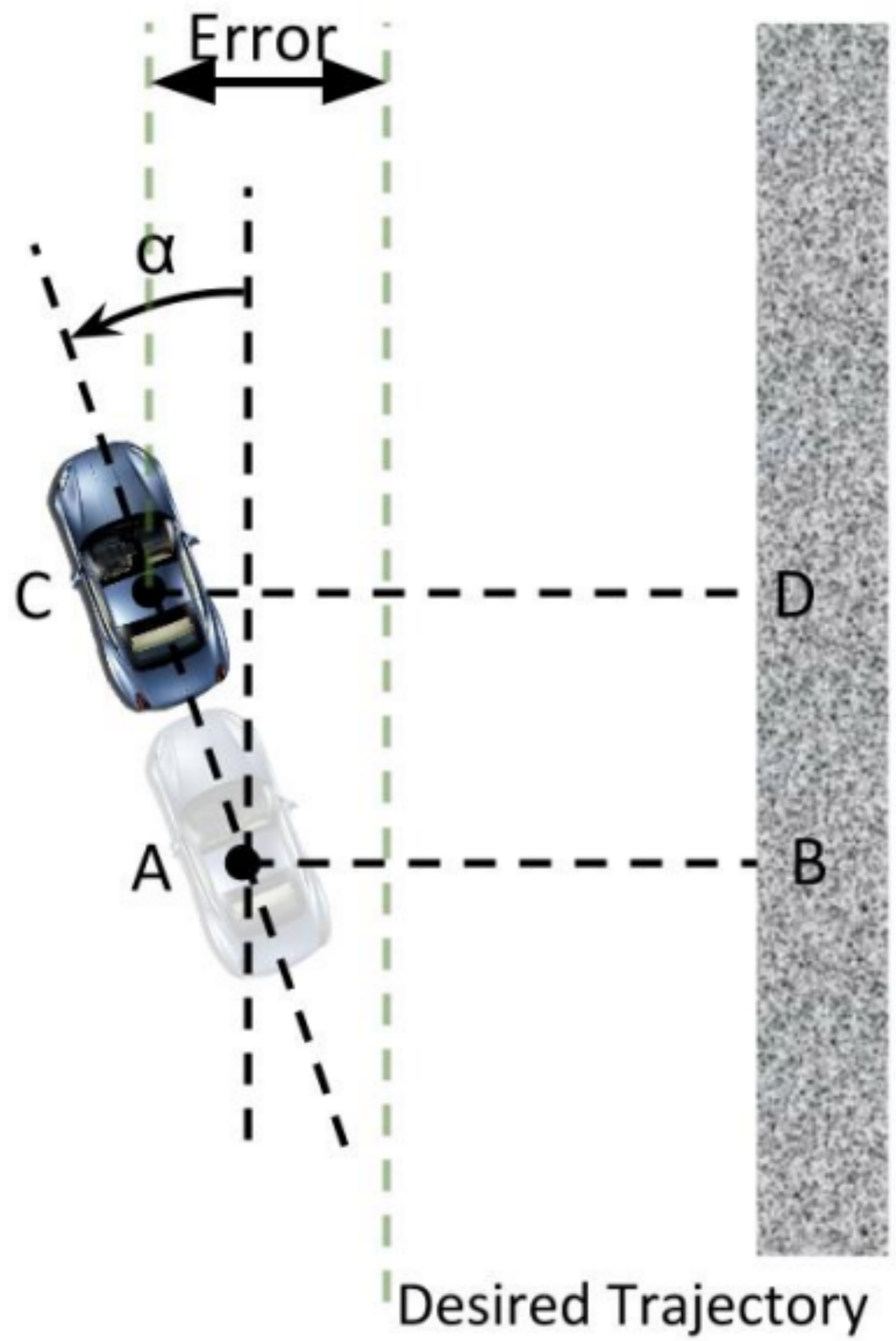


Figure 2: Calculating the orientation and distance from the wall

如果期望的轨迹距离右壁0.5米，那么通常PID必须控制的误差将是
[desired_distance – AB] = 0.5– AB

Figure 3: Projecting the car future in time



$$\alpha = \tan^{-1}\left(\frac{a\cos(\theta)-b}{a\sin(\theta)}\right)$$

$$AB = b\cos(\alpha)$$

$$CD = AB + AC\sin(\alpha)$$

[desired_distance - CD]

PID steering controller

$$V_\theta = K_p \times e(t) + K_d \frac{de(t)}{dt}$$

$$V_\theta = K_p \times error + K_d \times previous\ error\ \text{-}\ current\ error$$

参考

- https://cloud.tencent.com/developer/article/1107411

- https://en.wikipedia.org/wiki/Ziegler%E2%80%93Nichols_method
- https://baike.baidu.com/item/%E9%BD%90%E6%A0%BC%E5%8B%92%EF%BC%8D%E5%B0%BC%E7%A7%91%E5%B0%94%E6%96%AF%E6%96%B9%E6%B3%95/22694516

- https://zhuanlan.zhihu.com/p/509164571